Model Checking Norms and Sanctions in Institutions*

Francesco Viganò[1] and Marco Colombetti[1,2]

[1] Università della Svizzera italiana, via G. Buffi 13, 6900 Lugano, Switzerland
{francesco.vigano,marco.colombetti}@lu.unisi.ch
[2] Politecnico di Milano, piazza Leonardo Da Vinci 32, Milano, Italy
marco.colombetti@polimi.it

Abstract. In this paper we enrich FIEVeL (a modelling language for institutions amenable to model checking) with new constructs to describe norms and sanctions. Moreover, we present a specification language to reason about the effectiveness of norms and sanctions in shaping agent interactions. Finally we show that when properties of artificial institutions reflect certain interpretations of norms of human institutions, it is not always possible to satisfy them. As a consequence, regimentation of norms is not always a viable solution.

1 Introduction

Rules defined by *artificial institutions* and enforced by their software implementations, named *electronic institutions* [5], have been put forward as means to regulate open multiagent systems. Institutions define two kinds of rules [17]: *norms* (also named *regulative rules* [17]), which regulate existing activities, and *constitutive rules*, which create the very possibility of certain institutional actions.

Artificial institutions are often designed to reflect constitutive and regulative rules defined by human institutions in artificial systems [10,9,7], and model checking can play an important role to evaluate the compliance of artificial institutions with rules of human institutions and to compare design alternatives arising from different interpretations of such rules.

In general, when we map human rules only onto constitutive rules of artificial institutions, we obtain systems where violations cannot occur (they are regimented [10,9]). Instead, when we introduce regulative rules into artificial institutions, we obtain systems where violations may occur due, for instance, to the agents' autonomy. As a consequence, when we analyze results obtained by a model checker, it is important to consider how rules of human institutions have been mapped onto rules of artificial institutions: if a norm of a human institution has been mapped onto a set of constitutive rules of an artificial institution and a property that reflects it does not hold, then the artificial institution is incorrect. Instead, when a norm n has been mapped onto regulative rules of the artificial institution, we have to analyze whether: (i) norms of the artificial institution are correct, that is, a property reflecting expected effects of norm n holds over paths compliant with norms, and (ii) sanctions applied when norms are violated enforce desirable effects of norm n over all other possible evolutions.

* Supported by the Swiss National Science Foundation project 200020-109525, "Artificial Institutions: specification and verification of open distributed interaction frameworks."

J.S. Sichman et al. (Eds.): COIN 2007 Workshops, LNAI 4870, pp. 316–329, 2008.

The main contributions of this paper are threefold: first, we extend FIEVeL [19], a modelling language for institutions amenable to model checking, with new constructs to describe norms and sanctions, exemplifying how norms can be defined and enforced with our language; second, we present a flexible specification language which provides temporal operators that select paths compliant with certain sets of norms, showing that existing proposals (e.g. [12,16,1]) can be reduced to particular patterns of specification of our language; finally, we contribute to the ongoing debate about *regimentation* and *enforcement* of norms [10,9,6,8], showing that when human institutions impose a specific interpretation of norms, it may be the case that properties that reflect them cannot be satisfied by artificial institutions under the assumption that agents are autonomous. As a consequence, *regimentation* of norms is not always a viable solution.

The remainder of this paper is structured as follows: Section 2 introduces the OMSFOTL logic which is used to define the semantics of FIEVeL and to state properties of institutions in Section 3, where we provide an overview of our framework by resuming results discussed in our previous works. Section 4 presents how norms can be described with FIEVeL, while Section 5 introduces a language to define properties which consider only evolutions of institutions that comply with certain sets of norms. Section 6 explains how to formalize sanction mechanisms with FIEVeL and finally Section 7 provides a comparison of our approach with related works and presents some conclusions.

2 Ordered Many-Sorted First-Order Temporal Logic

An ordered many-sorted first-order temporal logic (OMSFOTL) is a many-sorted first-order logic [13] enriched with temporal operators and hierarchies of sorts. The signature of an OMSFOTL logic consists of a finite nonempty set of *sort symbols* Σ, a *hierarchy of sorts* $\leq_\Sigma$ (where $\sigma_1 \leq_\Sigma \sigma_2$ means that sort σ_1 is a *subsort* of sort σ_2), finite sets of *constants* (C), *function symbols* (F), and *predicate symbols* (P), and a denumerable set of *variables* (V). Moreover, an OMSFOTL signature defines function ξ which assigns a sort to every variable and every constant, and a signature (i.e. a sequence of sorts) to every function and predicate symbol. Given sorts Σ, the set T_σ of *terms of sorts* σ is the smallest set such that:

- $v \in \mathrm{T}_\sigma$ if $v \in \mathrm{V}$ and $\xi(v) \leq_\Sigma \sigma$;
- $c \in \mathrm{T}_\sigma$ if $c \in \mathrm{C}$ and $\xi(c) \leq_\Sigma \sigma$;
- $f(t_1, ..., t_n) \in \mathrm{T}_\sigma$ if $f \in \mathrm{F}$, $\xi(t_i) \leq_\Sigma [\xi(f)]_i$ for $1 \leq i \leq n$ and $[\xi(f)]_0 \leq_\Sigma \sigma$

where $[\xi(q)]_i$ refers to the i-th sort of the signature of a predicate or function symbol q. The set T of *terms* is the union of the sets T_σ for all $\sigma \in \Sigma$ and the set A of *atomic formulae* is the smallest set such that:

- $(t_1 = t_2) \in \mathrm{A}$ if there exists sort σ such that $\xi(t_1) \leq_\Sigma \sigma$ and $\xi(t_2) \leq_\Sigma \sigma$;
- $p(t_1, ..., t_n) \in \mathrm{A}$ if $p \in \mathrm{P}$ and $\xi(t_i) \leq_\Sigma [\xi(p)]_i$ for $1 \leq i \leq n$.

The set of *formulae* is defined according to the following grammar:

$$\varphi ::= \alpha \mid \neg\varphi \mid \varphi \wedge \varphi \mid \exists\varphi \mid \mathbf{X}\varphi \mid \varphi\mathbf{U}\varphi \mid \mathbf{E}\varphi$$

where α is an atomic formula.

The semantics of OMSFOTL is given with respect to a Kripke structure M, a path π (i.e., a sequence of states $\pi = s_0, s_1, s_2, \ldots$ of M), and an interpretation function I which, given a state s and an atomic formula $\alpha \in A$, returns a value in $\{0, 1\}$. In the sequel we use π_k to denote the k-th state of path π and π^k for the suffix of π starting at state π_k. A formula φ is true in a model M over a path π in M ($M, \pi \models \varphi$) when:

$M, \pi \models \alpha$ iff $I(\alpha, \pi_0) = 1$;
$M, \pi \models \neg\varphi$ iff $M, \pi \not\models \varphi$;
$M, \pi \models \varphi \wedge \psi$ iff $M, \pi \models \varphi$ and $M, \pi \models \psi$;
$M, \pi \models \exists x\varphi$ iff there exists a constant c of sort $\xi(x)$ such that $M, \pi \models \varphi_c$, where φ_c is obtained from φ by replacing all unbounded occurrences of variable x with constant c;
$M, \pi \models \mathbf{X}\varphi$ iff $M, \pi^1 \models \varphi$;
$M, \pi \models \varphi \mathbf{U} \psi$ iff exists an $i \geq 0$ such that $M, \pi^i \models \psi$ and for all $0 \leq j < i$ $M, \pi^j \models \varphi$;
$M, \pi \models \mathbf{E}\varphi$ iff there exists a path π' such that $\pi'_0 = \pi_0$ and $M, \pi' \models \varphi$

assuming, for the sake of presentation, that for each state s of M: (i) for each constant c of sort σ there exists an individual i such that $I(c, s) = i$, and (ii) that for each individual i there exists a constant c such that $I(c, s) = i$. Expressions *true*, *false*, $(\varphi \vee \psi)$, $(\psi \rightarrow \varphi)$, $(\varphi \leftrightarrow \psi)$, and $\forall x\varphi$ are defined in terms of $\neg$, $\wedge$, and $\exists$ in the conventional manner, and temporal operators $\mathbf{F}$, $\mathbf{G}$, and the path quantifier $\mathbf{A}$ are introduced as abbreviations as usual [3] to state that eventually φ holds ($\mathbf{F}\varphi \equiv true\mathbf{U}\varphi$), φ is satisfied by all states of a path ($\mathbf{G}\varphi \equiv \neg\mathbf{F}\neg\varphi$), and that all paths satisfy φ ($\mathbf{A}\varphi \equiv \neg\mathbf{E}\neg\varphi$).

In [20] we have shown that if we assume that each sort σ is associated to a finite domain D_σ, then OMSFOTL is as expressive as CTL* [4,3] and its models can be encoded with a finite number of atomic propositions. Despite it, we adopt OMSFOTL for two main reasons: (i), it represents an abbreviated form for long and complex formulae and (ii), institutions describe rules that typically are independent of the cardinality of domains and which can be naturally expressed by allowing quantification over sorts.

3 Modelling, Specifying, and Verifying Institutions

In [19] we proposed a metamodel of institutions based on the notion of an agent status function, which can be interpreted as a position involving a (possibly empty) set of institutionalized powers [11], obligations, prohibitions, etc. To formalize status functions and related concepts, we map them onto sorts, functions, and predicates of an OMSFOTL signature and define a set of axioms to capture their interrelations and temporal evolution. For instance, common aspects of status functions are represented by introducing sort σ_{sf}, which also defines the function $subject$ denoting the agent (σ_{aid}) the status function has been assigned to. Sort σ_{sf} also induces the two predicates $assigned$ and $modified$, which respectively represent if a status function is currently assigned (or revoked) and if it has been modified by the occurrence of an institutional event. Finally, the metamodel defines a set of axioms based on such symbols, for instance requiring that if a status function is not affected, then its subject does not change:

$$\mathbf{AG}\forall f(\neg\mathbf{X}modified(f) \rightarrow \exists a(subject(f) = a \wedge \mathbf{X}subject(f) = a)) \qquad \text{(A.1)}$$

```
1  basic-sorts:
2    σ_resources;
3    σ_reqState = {answ,notAnsw};
4  base-events:
5    message giveResource(rec:σ_aid,res:σ_resources);
6      ...
7  institution resourceManagement {
8    status-function member() {...}
9    status-function requested(reqRes:σ_resources,ag:σ_aid,
10     sta:σ_reqState){...}
11   status-function holder(resource:σ_resources){
12   key resource;
13   powers give ← (∃ r:σ_requested (assigned(r)∧ag(r)=rec∧
14        reqRes(r)=resource(f)∧sta(r)=answ)∧res=resource(f));
15   }
16     ...
17   institutional-events:
18      institutional-action give(rec:σ_aid,res:σ_resources)
19      pre ∃ x:σ_member(assigned(x)∧subject(x)=rec
20          ∧¬subject(x)=actor);
21      eff r:σ_requested revoke (reqRes(r)=res),
22          k:σ_holder assign (subject(k)=rec,resource(k)=res);
23    ...
24   conventions
25      exch-Msg(giveResource) [true]=c=> give
26          [rec=c=>rec res=c=>res]
27     ...
28 }
```

Fig. 1. Fragments of the Resource Management institution

An institution evolves because events (σ_{ev}) occur or agents perform actions ($\sigma_{act} \leq_{\Sigma} \sigma_{ev}$). Each event-type e induces a sort σ_e and three predicates, $happens_e$, $prec_e$, and eff_e, which express when an event of type e happens and what conditions must be satisfied before and after its occurrence. In contrast with *base-level events* (e.g., *exchange-message* events), the occurrence of an *institutional event* (σ_{ie}) requires that another event conventionally associated to it occurs and that, in the case of institutional actions, the actor must be empowered to perform it:

$$\mathbf{AG}\forall\overline{x}((prec_{ia}(\overline{x}) \wedge \exists f(subject(f) = x_1 \wedge empowered_{ia}(f,\overline{x}) \wedge assigned(f) \wedge \bigvee_{a\in\sigma_{act}} \mathbf{X}(conv_{a-ia}(\overline{x}) \wedge happens_a(\overline{x}'))) \leftrightarrow \mathbf{X}happens_{ia}(\overline{x})) \tag{A.2}$$

where: $\overline{x}$ is a set of variables determined by predicate $happens_{ia}$; the first variable of $\overline{x}$ refers to the actor of action ia; predicate $empowered_{ia}$ states when status functions are empowered to perform institutional action ia; predicate $conv_{a-ia}$ represents the existence of a convention among action a and institutional action ia; and $\overline{x}'$ reflects how arguments of ia are mapped over arguments of action a.

To model institutions in terms of the concepts described by our metamodel, in [19] we introduced FIEVeL, a modelling language for institutions, whose syntax is exemplified in Figure 1 and whose semantics is given by providing a translation of its constructs into a set of symbols and formulae of an OMSFOTL logic. According to Figure 1, in the Resource Management institution a $member$ can request a $holder$ to $give$ the control of one of its resources. When an agent accepts to satisfy the request, it is empowered to give a resource to the agent that has requested it, which becomes its new $holder$. More precisely, line 2 of Figure 1 induces sort $\sigma_{resources}$, which represents a set of resources, while lines 8-10 introduce status functions $member$ (σ_{member}) and $requested$ ($\sigma_{requested}$), which represent respectively the status that an agent should have to request the control over a resource and the status acquired after having successfully performed a request to an $holder$. Resources are hold by agents through status function $holder$ (declared at line 11 of Figure 1), which defines sort $\sigma_{holder} \leq_{\Sigma} \sigma_{sf}$ and function $resource$ of signature $\xi(resource) = \langle \sigma_{resources}, \sigma_{holder} \rangle$. According to lines 13 and 14, an $holder$ is empowered to $give$ a resource res when an agent has requested it and the $holder$ has already acknowledged to transfer the control over the requested resource as required by the following axiom:

$$\begin{aligned}\mathbf{AG}\forall s \forall actor \forall rec \forall res(empowered_{give}(s, actor, rec, res) \leftrightarrow \exists f(f = s \wedge \\ (\exists \mathtt{r}(\mathtt{assigned(r)} \wedge \mathtt{ag(r)} = \mathtt{rec} \wedge \mathtt{reqRes(r)} = \mathtt{resource(f)} \wedge \\ \mathtt{sta(r)} = \mathtt{answ}) \wedge \mathtt{res} = \mathtt{resource(f)}))))\end{aligned} \quad \text{(A.3)}$$

where $\xi(actor) = \xi(rec) = \sigma_{aid}$, $\xi(res) = \sigma_{resources}$, $\xi(s) = \sigma_{sf}$, $\xi(f) = \sigma_{holder}$, and $\xi(r) = \sigma_{requested}$.

According to FIEVeL semantics, lines 18-22 define institutional action $give$ such that: (i) it can be performed only if the $receiver$ is a $member$ and if it is not the $actor$, and (ii) it revokes status function $requested$ to the receiver and assigns status $holder$ to it. More precisely, institutional action $give$ induces predicates $happens_{give}$, $prec_{give}$, and eff_{give} ($\xi(happens_{eff}) = \langle \sigma_{aid}, \sigma_{aid}, \sigma_{resources} \rangle$) such that predicates $prec_{give}$ and eff_{give} satisfy the following axioms:

$$\begin{aligned}\mathbf{AG}\forall actor \forall rec \forall res(prec_{give}(actor, rec, res) \leftrightarrow (\exists \mathtt{x}(\mathtt{assigned(x)} \wedge \\ \mathtt{subject(x)} = \mathtt{rec} \wedge \neg \mathtt{subject(x)} = \mathtt{actor})))\end{aligned} \quad \text{(A.4)}$$

$$\begin{aligned}\mathbf{AG}\forall actor \forall rec \forall res(eff_{give}(actor, rec, res) \leftrightarrow \mathbf{X}(\forall r(reqRes(r) = res \rightarrow \\ (\neg assigned(r) \wedge modified(r))) \wedge \forall k(resource(k) = res \rightarrow \\ (assigned(k) \wedge modified(k) \wedge subject(k) = rec)))\end{aligned} \quad \text{(A.5)}$$

where $\xi(actor) = \xi(rec) = \sigma_{aid}$, $\xi(res) = \sigma_{resources}$, $\xi(x) = \sigma_{member}$, $\xi(r) = \sigma_{requested}$, and $\xi(k) = \sigma_{holder}$.

Finally, lines 25 and 26 define a convention such that the exchange of a message of type $giveResource$ counts-as the performance of action $give$ when the sender is empowered and preconditions of action $give$ are satisfied. As a consequence, axiom (A.2) is instantiated as follows:

$$\mathbf{AG}\forall actor\forall rec\forall res((prec_{give}(actor, rec, res) \wedge \exists f(subject(f) = actor \wedge assigned(f) \wedge empowered_{give}(f, actor, rec, res) \wedge \mathbf{X}(happens_{giveResource}(actor, rec, res) \wedge conv_{giveResource-give}(actor, rec, res))) \leftrightarrow \mathbf{X}happens_{give}(actor, rec, res)) \quad \text{(A.6)}$$

where $\xi(actor) = \xi(rec) = \sigma_{aid}$, $\xi(res) = \sigma_{resources}$, and $\xi(f) = \sigma_{sf}$.

In our framework, also properties are specified in terms of OMSFOTL formulae such that temporal operators ($\mathbf{X}$, $\mathbf{G}$, $\mathbf{F}$, and $\mathbf{U}$) are always preceded by a path quantifier ($\mathbf{E}$ or $\mathbf{A}$). One of the main advantages of our approach resides in the fact that any symbol introduced by our metamodel or by an institution can appear in a property. Furthermore, to increase the flexibility of the language, occurrences of events are referenced with a generic predicate $happens$ and we write "$x : \sigma$" to say that variable x is of sort σ. For instance, the following property requires that whenever an agent receives a positive answer to its requests, it will eventually become the *holder*:

$$\mathbf{AG}\forall act : \sigma_{aid}\forall rec : \sigma_{aid}\forall res : \sigma_{resources}(happens(accept, act, rec, res) \rightarrow \mathbf{AF}\exists h : \sigma_{holder}(subject(h) = rec \wedge resource(h) = res)) \quad \text{(P.1)}$$

Analogously, we can also check if whenever a holder accepts to give a resource, it will eventually do so:

$$\mathbf{AG}\forall act : \sigma_{aid}\forall rec : \sigma_{aid}\forall res : \sigma_{resources}(happens(accept, act, rec, res) \rightarrow \mathbf{AF}happens(give, act, rec, res)) \quad \text{(P.2)}$$

In [20] we presented a symbolic model checker specifically developed to verify FIEVeL institutions. Given an institution and a set of properties, our tool proceeds as follows: (i) it converts the institution into a set Φ of OMSFOTL formulae by considering the semantics of FIEVeL constructs and axioms determined by our metamodel (see axioms (A.3), (A.4), and (A.5)); (ii) formulae Φ are translated into propositional logic and subsequently converted into a formula in conjunctive normal form (CNF); (iii) given the set of assignments satisfying the CNF (whose disjunction constitutes the transition relation of a Kripke structure) and a formula φ_0, representing a set of initial states, a symbolic representation of an institution is built and is exploited to verify properties by applying standard symbolic algorithms [3]. According to our model checker, properties (P.1) and (P.2) do not hold: since constitutive rules reported in Figure 1 define possible actions that agents can carry out, but do not ensure that empowered agents will necessarily perform them, it may be the case that agents accept to give their resources but do not perform action $give$.

4 Norms

To define the semantics of norms, our metamodel assumes the existence of sort σ_o, whose individuals reify norms of institutions. Sort σ_o is used to express prohibitions and obligations characterized by certain deadlines (not necessarily a time expression),

and we consider that a state of affairs is permitted if it is reached without violating any norm. In particular, for the sake of conciseness, in this paper we focus only on norms which are considered fulfilled or violated only *once* after a given status function is imposed on an agent and certain conditions are met. Given sort σ_{state}, which introduces constants $unfired$, $activated$, and $inactive$, sort σ_o is characterized by function $state$ ($\xi(state) = \langle \sigma_{state}, \sigma_o \rangle$), which keeps trace of the temporal evolution of a norm, a set of timers (e.g., function $activation$ which counts how many time events have occurred since a norm has been activated), and by a set of predicates ($start$, $fulfillment$, and $violation$ of signature $\xi(violation) = \langle \sigma_{sf}, \sigma_o \rangle$). Agents are subject to norms when certain status functions are imposed on them: to model the interdependency among norms and status functions, we introduce function $ofStatus$ ($\xi(ofStatus) = \langle \sigma_{sf}, \sigma_o \rangle$) which denotes the status function an obligation is associated to. When a status function is not assigned, then its norms are considered to be $inactive$ and cannot be violated: we represent this fact by the following axiom, which states that norms of a revoked status function are always $inactive$:

$$\mathbf{AG} \forall o \forall f ((ofStatus(o) = f \wedge \neg assigned(f)) \rightarrow state(o) = inactive) \quad \text{(A.7)}$$

where $\xi(o) = \sigma_o$ and $\xi(f) = \sigma_{sf}$. Similarly, Axiom (A.8) requires that when a status function is imposed on an agent, then the state of a norm is set to $unfired$ if predicate $start$ is not satisfied, otherwise it is set to $activated$:

$$\mathbf{AG} \forall o \forall f ((ofStatus(o) = f \wedge \mathbf{X}(assigned(f) \wedge modified(f))) \rightarrow ((\neg start(o, f) \wedge \mathbf{X} state(o) = unfired) \vee (start(o, f) \wedge \mathbf{X} state(o) = activated))) \quad \text{(A.8)}$$

Axioms (A.7) and (A.8), as well as other axioms omitted here for the sake of brevity, describe the temporal evolution of functions $state$ and $activation$, which in combination with predicates $fulfillment$ and $violation$, determine when an obligation should be considered to be infringed. In particular, given predicate $violated$ of signature $\xi(violated) = \langle \sigma_o \rangle$, a norm is violated if and only if it was $activated$, the associated status function is not modified, $violation$ holds while $fulfillment$ is false:

$$\mathbf{AG} \forall o \forall f (ofStatus(o) = f \rightarrow (\mathbf{X} violated(o) \leftrightarrow (state(o) = active \wedge (violation(o) \wedge \neg fulfillment(o) \wedge \neg \mathbf{X} modified(f))))) \quad \text{(A.9)}$$

Norms are described in FIEVeL according to the following syntax:

```
norm ::= symbol start fulfillment violation ;
start ::= "start" "<->" expression ";" ;
fulfillment ::= "fulfillment" "<->" expression ";" ;
violation ::= "violation" "<->" expression ";" ;
```

where `expression` is an OMSFOTL formula which does not contains **U**, **E**, **G**, or nested occurrences of **X**. Moreover, given that a norm is described within a status function σ_s, free occurrences of a variable f of sort σ_s may appear in any formula used to describe a norm's condition. A norm $symbol$ induces sort $\sigma_{symbol} \leq_\Sigma \sigma_o$ and

defines under what conditions predicates $fulfillment$, $violation$, and $start$ hold when are evaluated over an obligation of sort σ_{symbol}, as exemplified by the following axiom schema:

$$\mathbf{AG}\forall o \forall f(fulfillment(o,f) \leftrightarrow (ofStatus(o) = f \wedge \texttt{expression})) \qquad \text{(A.10)}$$

where $\xi(o) = \sigma_{symbol}$ and $\xi(f) = \sigma_s$. Combining instances of Axiom Schema (A.10) (and similarly for predicates $violation$ and $start$) with Axiom(A.9), it is possible to automatically classify states with respect to each norm defined by an institution. In contrast with other approaches (e.g., [16] and [1]), in our framework designers can describe norms at a high-level in terms of institutional concepts, ignoring the actual number of states and transitions admitted by an institution. For instance, the following norm, named $h1$ and associated to the $holder$ status function, states that once a holder accepts to give the control of a resource, then it ought to do so before a certain time interval elapses:

```
h1  start<->X ∃ ag:σ_aid ∃ rec:σ_aid ∃ res:σ_resources (subject(f)=ag ∧
            resource(f)=res ∧ happens(accept,ag,rec,res));
    fulfillment<->∃ ag:σ_aid ∃ rec:σ_aid ∃ res:σ_resources (subject(f)=ag
            ∧ res=resource(f) ∧ X happens(give,ag,rec,res));
    violation<->(activation(o)=1 ∧ X happens(time));
```

Without proper sanction mechanisms, the introduction of norms typically does not change the set of properties satisfied by an institution, given that autonomous agents may not comply with such norms [5,2,9,18,7]: as a consequence certain properties may not hold in an institution even if its rules are correctly stated. For instance, properties (P.1) and (P.2) do not hold in the new model of the Resource Management institution obtained by adding norm $h1$, despite this correctly requires that a $holder$ gives a resource after it has positively answered to an agent. This is due to the fact that norms regulate existing activities, describing what evolutions of an institution should be considered as legal, but do not change the temporal evolution admitted by an institution.

5 Normed Temporal Operators

To analyze whether an institution may lead a system into certain states when its norms are respected, we can exploit predicate $violated$ and the fact that in our framework norms are reified as norm individuals. Therefore, it is possible to quantify over sort σ_o (and its subsorts induced by each norm), investigating how norms condition the evolution of an institution. In particular, in this paper we define operators that allow designers to reason about what properties are satisfied by an institution when a set of norm individuals are not violated. More precisely, given a set of norms which constitute the extension of formula φ_o (an open formula in which variable o of sort σ_o occurs free), *normed temporal operators* are defined as follows:

- $\mathbf{EG}^{\varphi_o}\varphi =_{def} \mathbf{EG}(\forall o : \sigma_o(\varphi_o \rightarrow \neg violated(o)) \wedge \varphi)$;
- $\mathbf{EX}^{\varphi_o}\varphi =_{def} \mathbf{EX}(\forall o : \sigma_o(\varphi_o \rightarrow \neg violated(o)) \wedge \varphi)$;
- $\mathbf{E}\psi\mathbf{U}^{\varphi_o}\varphi =_{def} \mathbf{E}(\forall o : \sigma_o(\varphi_o \rightarrow \neg violated(o)) \wedge \psi)\mathbf{U}(\forall o : \sigma_o(\varphi_o \rightarrow \neg violated(o)) \wedge \varphi)$;

Since the satisfaction of CTL temporal operators (with the exception of $\mathbf{EX}$) refers to the initial state π_0 of a path π [4,3], then also their normed counterparts refer to state π_0. As a consequence, if state π_0 violates norms φ_o, then the normed operators $\mathbf{EG}^{\varphi_o}$ and $\mathbf{EU}^{\varphi_o}$ are trivially falsified. This may occur when the system is inconsistent or because normed temporal operators are nested and external operators do not ensure compliance with norms considered by internal operators. While in the former case we would conclude that our system is irrational, in the latter case we may get counter-intuitive results. To avoid this, we can prefix internal operators with $\mathbf{EX}^{\varphi_o}$, ensuring that the initial state is not considered and only paths compliant with norms of internal operators are taken into account. Despite this problem may be avoided by different definitions of normed temporal operators, we consider more relevant the fact that normed and unnormed operators are evaluated over the same set of states and are expressed in terms of the standard semantics of CTL [4,3]. In doing so, if formula φ_o refers to an empty set of obligations, then normed temporal operators are equivalent to their temporal counterpart (e.g., $\mathbf{EG}^{false}\varphi \equiv \mathbf{EG}\varphi$), and $\mathbf{EG}^{\varphi_o}$, $\mathbf{EX}^{\varphi_o}$, and $\mathbf{EU}^{\varphi_o}$ constitute an adequate set of operators, since we have the following equivalences:

- $\mathbf{EF}^{\varphi_o}\varphi \equiv \mathbf{E}true\mathbf{U}^{\varphi_o}\varphi$;
- $\mathbf{AG}^{\varphi_o}\varphi \equiv \neg\mathbf{EF}^{\varphi_o}\neg\varphi \wedge \mathbf{EG}^{\varphi_o}true$;
- $\mathbf{AX}^{\varphi_o}\varphi \equiv \neg\mathbf{EX}^{\varphi_o}\neg\varphi \wedge \mathbf{EX}^{\varphi_o}true$;
- $\mathbf{A}\psi\mathbf{U}^{\varphi_o}\varphi \equiv \neg(\mathbf{E}\neg\varphi\mathbf{U}^{\varphi_o}(\neg\varphi \wedge \neg\psi)) \wedge \neg\mathbf{EG}^{\varphi_o}\neg\varphi \wedge \mathbf{EF}^{\varphi_o}\varphi$;
- $\mathbf{AF}^{\varphi_o}\varphi \equiv \neg\mathbf{EG}^{\varphi_o}\neg\varphi \wedge \mathbf{EF}^{\varphi_o}\varphi$;

It is worth observing that by definition, the consistency of norms represents a necessary condition for the satisfaction of normed temporal operators universally quantified over paths, otherwise they would be trivially satisfied by an inconsistent normative system. In contrast with other specification languages characterized by a normative flavor (e.g. [14,16,1]), which assume that the normative system is consistent (i.e., there exists a legal outward transition for every state) either by assuming axiom D [14] or as an explicit hypothesis on the transition system [16,1], in our approach the absence of contradictory norms represents a desirable property that a rational institution ought to satisfy and that can be verified by our model checker. To exemplify the use of normed temporal operators, we modify Property (P.2) such that if holders respect all norms of the institution and they perform action $accept$, then they will $give$ their resources:

$$\mathbf{AG}\forall act : \sigma_{aid}\forall rec : \sigma_{aid}\forall res : \sigma_{resources}(happens(accept, act, rec, res) \rightarrow \mathbf{AF}^{\exists h:\sigma_{holder}\exists f:\sigma_{sf}(subject(h)=subject(f)\wedge ofStatus(o)=f)} happens(give, act, rec, res)) \quad \text{(P.3)}$$

We can also rewrite property (P.1) to investigate whether norm $h1$ is capable of directing the behavior of holders in such a way that when an agent has requested a good and has received a positive answer, it will eventually become the holder of the good:

$$\mathbf{AG}\forall act : \sigma_{aid}\forall rec : \sigma_{aid}\forall res : \sigma_{resources}((happens(accept, act, rec, res) \rightarrow \mathbf{AF}^{\exists w:h1(w=o)}\exists h : holder(subject(h) = rec \wedge resource(h) = res))) \quad \text{(P.4)}$$

To conclude this section we compare the expressiveness and the flexibility of our approach to the specification languages proposed in [1] and [12]. In [1] the authors

proposed *Normative Temporal Logic* (NTL), a language similar to CTL with the exception that operators $\mathbf{A}$ and $\mathbf{E}$ are replaced by O_η and P_η, which intuitively can be read as "for all paths compliant with the normative system η" and "there exists a path compliant with the normative system η". Given the semantics provided in [1] and assuming that η represents a set of norms, NTL operators are equivalent to normed temporal operators characterized by a formula φ_η representing all individuals of sorts belonging to η. For instance, formula $\mathrm{O}\Box_\eta\varphi$ of NTL corresponds to $\mathbf{AX}^{\varphi_\eta}\mathbf{AG}^{\varphi_\eta}\varphi$, where φ_η is defined as follows: $\varphi_\eta \equiv \bigwedge_{\sigma_n \in \eta} \exists k : \sigma_n(k = o)$.

In [12] Lomuscio and Sergot presented a modal operator $\mathrm{O}_a\varphi$ which expresses the fact that φ holds over reachable states where agent a complies with its protocol. Assuming that a is an agent, $\mathrm{O}_a\varphi$ is equivalent to $\mathbf{AX}^{\exists f(ofStatus(o)=f \wedge subject(f)=a)}\varphi$. While NTL does not provide any construct to reason about agents, in [12] it is possible to investigate only the compliance of agents with the whole set of norms (described as a protocol): instead, normed temporal operators allow us to reason about subsets of norms and agents, and to express complex interdependencies among them as exemplified by Property (P.3).

6 Sanction Mechanisms

To guarantee that those agents that follow norms are not damaged by those who do not, institutions should provide rules that describe what kind of sanctions are applied when agents violate norms. According to [17], the imposition of status functions constitutes a necessary condition for the application of sanctions, since "with that new status come the appropriate punishment" [17, pag. 50]. Such status functions not only may provide new powers and new obligations (prohibitions), but may also revoke or change existing powers or norms: for instance, the exclusion of an agent from an interaction ruled by an institution (e.g., an auction) means that powers and norms defined by such institution have been revoked. Analogously, officials can apply sanctions only if they have the necessary powers, and certain obligations (prohibitions) may further regulate how such powers ought to be exerted. Therefore, given that sanctions modify the powers and norms of agents, we propose to model sanction mechanisms as rules that impose or revoke status functions when a norm is violated.

In our framework sanction mechanisms are defined according to the following grammar:

```
sanction ::= "sanction" symbol "pre" expression ";" "eff" post
    ("," post)* ";" ;
precondition ::= expression;
post ::= (selection "-X->")? effects
selection ::= var ("," var)* "(" expression ")"
effects ::= var ("assign"|"revoke") "(" term "=" term
    ("," term"=" term)* ")";
```

where `expression` is an OMSFOTL formula which does not contain temporal operators or path quantifiers, and `post` is constituted by (i) an (optional) *selection expression* and (ii) an expression describing what statuses are assigned or revoked when the sanction mechanism is activated. As we will see, effects must hold when a violation

is detected, while the selection expression is evaluated in the previous state. For this reason, we separate the selection expression from the effects through symbol `-X->`.

For instance, the following sanction mechanism describes that when a norm $h1$ is violated, then the resource is assigned to the agent that has requested the good and powers and obligations associated to status function $requested$ are revoked:

```
sanction h1
pre true;
eff r2:σ_requested revoke (reqRes(r2)=resource(f)),
    r1:σ_requested res:σ_resources a:σ_aid(res=resource(f)∧reqRes(r1)=res
        ∧ a=requester(r1)) -X->
        r2:σ_holder assign(resource(r2)=res,subject(r2)=a)
```

Before continuing with our presentation, it is worth remarking that in our approach sanction mechanisms reflect what powers, obligations, and prohibitions are assigned to agents when violations are observed, which does not necessarily means that sanctions (like fines) are automatically enforced by the system. Despite designers may decide to enforce norms through automatic reactions of the system, FIEVeL allows to model scenarios where sanction mechanisms confer powers to certain agents to punish violations: for instance, when an agent violates a norm, an officer may be empowered to impose a fine and obliged to do so before a certain time instant.

Sanction mechanisms do not induce any new sort: instead, each of them introduces two predicates, pre_{san_i} and eff_{san_i}, which respectively represent a condition that must be satisfied before a violated obligation activates the i-th sanction mechanism, and its effects. Predicates pre_{san_i} (and analogously predicates eff_{san_i}) are determined by the obligation sort that must be sanctioned (σ_{symbol}) and the status function that defines it (σ_s). Furthermore, predicate pre_{san_i} must satisfy the following axiom schema:

$$\mathbf{AG}\forall o \forall f (pre_{san_i}(o, f) \leftrightarrow \texttt{precondition}_i) \tag{A.11}$$

where $\xi(o) = \sigma_{symbol}$ and $\xi(f) = \sigma_s$. Similarly, each sanction mechanism instantiates the following axiom schema which defines what status functions are imposed or revoked when a sanction mechanism is activated:

$$\begin{aligned}\mathbf{AG}\forall o \forall f (eff_{san_i}(o, f) \leftrightarrow (\bigwedge_{k=0}^{K_i} \forall \overline{s}_{k_i} (\texttt{expression}_{k_i} \rightarrow \mathbf{X}\exists t_{k_i} \\ ([\neg] assigned(t_{k_i}) \wedge \bigwedge_{l=1}^{N_{k_i}} term_{k_i,l,1} = term_{k_i,l,2})))))\end{aligned} \tag{A.12}$$

where variables $\overline{s}_{k_i}$ is a set of variables defined by the k-th effect expression of the i-th sanction mechanism and t_{k_i} represents status functions that will be assigned or revoked. Finally, the following axiom schema states that the i-th sanction mechanism brings about its effects when it is activated by the violation of an obligation and its preconditions are met:

$$\mathbf{AG}\forall o \forall f ((ofStatus(o) = f \wedge pre_{san_i}(o, f) \wedge \mathbf{X} violated(o)) \rightarrow eff_{san_i}(o, f)) \tag{A.13}$$

Axiom Schema (A.13) suggests that, as institutional events, also sanction mechanisms concur to the definition of predicate $modified$, which ensures that a status is not assigned (revoked) when no institutional event or sanction mechanism affects it (see Section 3). Moreover, Axiom Schema (A.13) describes the main difference among institutional events and sanction mechanisms: while the former happen because other events occur and certain conditions are satisfied (see Axiom (A.2)), the latter are fired only by violations. To some extend, we can interpret Axiom Schema (A.13) as defining a single convention for the activation of any sanction mechanism.

Properties (P.1) and (P.2) can be regarded as two different interpretations of the human norm "when agents accept to give a resource, then requesters ought to become the new holders", where the latter property explicitly refers to the actor and the action that ought to be performed. Norm $h1$ introduced in Section 4 reflects such rule and the introduction of a sanction mechanism for norm $h1$ changes the set of constitutive rules in such a way that Property (P.1) is satisfied by the Resource Management institution. Observing Figure 1, we can notice that the violation of norm $h1$ forces the effects of action $give$, but not the performance of the action itself: therefore, we can expect that Property (P.2) still does not hold, which is confirmed by our model checker. As it has been formulated and unless we introduce a convention such that $accept$ counts as $give$ (which may be incompatible with the rules of a human institution), we think that it is impossible to devise a mechanism to satisfy Property (P.2), since it would mean that we are capable of forcing an autonomous agent to act.

7 Discussion and Conclusions

In this paper we have extended FIEVeL with new constructs to model normative aspects of institutions and we have introduced a flexible specification language to define properties regarding paths that are compliant with norms. We have also exemplified how an institution can be developed by using our approach, verifying that it satisfies certain requirements and modifying its constitutive and regulative rules to comply with them. We have also shown that when properties stem from norms of human institutions that artificial institutions should reflect, it is not always possible to satisfy them, at least under certain interpretations of the human institutions.

In [9] Grossi et al. presented an overview of the role of norms and sanctions in institutions. According to [9] it seems that every norm can be either regimented or enforced, while we think that the viability of such mechanisms depends on the meaning attributed by designers to norms. As we have seen, certain interpretations may exclude the possibility of regimenting them and, generally speaking, regimentation of norms regarding institutional aspects can be achieved only by converting regulative rules into constitutive rules. More precisely, prohibitions can be regimented by revoking powers [6,7] while obligations can be enforced by changing the interpretation of certain terms. For instance, norm "all yes/not questions should be answered" can be trivially regimented by assuming that silence counts as a positive (negative) answer. Instead, assuming that only a message sent by an agent counts as a communicative act (like in [7]) it is impossible to regiment such norm.

In [6] sanctions are considered only as rules which *create* new obligations (commitments) and powers, while in this paper we have claimed that sanctions may also *delete* obligations and powers by revoking status functions. Moreover, the approach discussed in [6] is based on an intuitive semantics, which does not allow the development of a framework to verify properties guaranteed by institutions. Analogously, the correctness of protocols modelled in terms of institutional concepts by Artikis et al. [2,15] is only guaranteed by systematic executions. Despite the terminologies used in this paper and in [2] are quite similar, in [2] *physical actions* can be performed only by agents playing a specific role, suggesting that such actions are actually institutional. Furthermore, the formalism used in [2,15] does not provide any abstraction to describe that every institutional action must be empowered in order to be successfully executed. Instead, the authors have to specify this fact for every single action and for every role.

In [8] a rule language is introduced to model norms and to represent the effects of concurrent events. The author proposed the notion of *enforcing events*, which means that obligatory events are considered as if they were executed even when agents do not perform them. In our opinion, events' enforcement transforms regulative rules into constitutive rules, by defining when time events count as obligatory events, and represents an effective mechanism to describe automatic updates of institutions. In general, we believe that it is not possible to enforce all kinds of events, especially those (like actions) that can only be performed by autonomous agents.

The constructs presented in Section 4 constitute a high-level description of norms, and our tool automatically classifies transitions and states as compliant with each norm of the system. In this respect, our approach is similar to the one presented in [18]. Instead, the input language of the model checker described in [16] requires designers to explicitly list the set of states that each agent may reach, and to classify them as *red* (an agent violates the protocol) or *green*. Although red states are such only because they violate a protocol [12,16], such classification is not inferred from the protocol but must be manually provided independently from it: therefore designers may introduce discrepancies among the protocol and the classification of states. Similarly, in [1] systems are described with a low-level language which requires to associate a name to each transition, and norms can be defined only by listing under what conditions a set of transitions is considered legal.

In the future we plan to define a translation of axioms stemming from our metamodel and from FIEVeL models into Prolog, providing a single framework for the definition, verification, and monitoring of institutions.

References

1. Ågotnes, T., van der Hoek, W., Rodríguez-Aguilar, J.A., Sierra, C., Wooldridge, M.: On the logic of normative systems. In: Proceedings of the 20th International Joint Conference on Artificial Intelligence, pp. 1175–1180 (2007)
2. Artikis, A., Kamara, L., Pitt, J., Sergot, M.J.: A Protocol for Resource Sharing in Norm-Governed Ad Hoc Networks. In: Leite, J.A., Omicini, A., Torroni, P., Yolum, p. (eds.) DALT 2004. LNCS (LNAI), vol. 3476, pp. 221–238. Springer, Heidelberg (2005)
3. Clarke, E.M., Grumberg, O., Peled, D.: Model Checking. MIT Press, Cambridge (1999)

4. Emerson, E.A., Halpern, J.Y.: "Sometimes" and "not never" revisited: on branching versus linear time temporal logic. Journal of the ACM 33(1), 151–178 (1986)
5. Esteva, M., Rodríguez-Aguilar, J.A., Sierra, C., Garcia, P., Arcos, J.L.: On the Formal Specification of Electronic Institutions. In: Sierra, C., Dignum, F.P.M. (eds.) AgentLink 2000. LNCS (LNAI), vol. 1991, pp. 126–147. Springer, Heidelberg (2001)
6. Fornara, N., Colombetti, M.: Specifying and Enforcing Norms in Artificial Institutions. In: Omicini, A., Dunin-Keplicz, B., Padget, J. (eds.) Proceedings of the 4th European Workshop on Multi-Agent Systems (2006)
7. Fornara, N., Viganò, F., Colombetti, M.: Agent Communication and Artificial Institutions. Autonomous Agents and Multi-Agent Systems 14(2), 121–142 (2007)
8. García-Camino, A.: Ignoring, Forcing and Expecting Concurrent Events in Electronic Institutions. In: Sichman, J.S., et al. (eds.) COIN 2007 Workshops. LNCS (LNAI), vol. 4870, pp. 316–329. Springer, Heidelberg (2008)
9. Grossi, D., Aldewereld, H., Dignum, F.: Ubi lex, ibi poena: Designing norm enforcement in e-institutions. In: Noriega, P., Vázquez-Salceda, J., Boella, G., Boissier, O., Dignum, V., Fornara, N., Matson, E. (eds.) COIN 2006, vol. 4386, pp. 110–124. Springer, Heidelberg (2007)
10. Jones, A., Sergot, M.J.: On the characterization of law and computer systems: The normative systems perspectives. In: Deontic Logic in Computer Science: Normative Systems Specification, pp. 275–307 (1993)
11. Jones, A., Sergot, M.J.: A formal characterisation of institutionalised power. Journal of the IGPL 4(3), 429–445 (1996)
12. Lomuscio, A., Sergot, M.: A formulation of violation, error recovery, and enforcement in the bit transmission problem. Journal of Applied Logic 1(2), 93–116 (2002)
13. Manzano, M.: Introduction to many-sorted logic. In: Many-sorted logic and its applications, pp. 3–86. John Wiley, Chichester (1993)
14. Meyer, J.-J., Wieringa, R.J.: Deontic Logic: A Concise Overview. In: Deontic Logic in Computer Science: Normative Systems Specification, pp. 3–16. John Wiley, Chichester (1993)
15. Pitt, J., Kamara, L., Sergot, M., Artikis, A.: Formalization of a voting protocol for virtual organizations. In: Proceedings of the 4th Conference on Autonomous agents and Multi-Agent Systems, pp. 373–380 (2005)
16. Raimondi, F., Lomuscio, A.: Automatic Verification of Deontic Interpreted Systems by Model Checking via OBDD's. In: Proceedings of the 16th European Conference on Artificial Intelligence, pp. 53–57 (2004)
17. Searle, J.R.: The construction of social reality. Free Press, New York (1995)
18. Sergot, M.J., Craven, R.: The Deontic Component of Action Language nC+. In: Goble, L., Meyer, J.-J.C. (eds.) DEON 2006. LNCS (LNAI), vol. 4048, pp. 222–237. Springer, Heidelberg (2006)
19. Viganò, F., Colombetti, M.: Specification and Verification of Institutions through Status Functions. In: Noriega, P., et al. (eds.) COIN 2006. LNCS (LNAI), vol. 4386, pp. 125–141. Springer, Heidelberg (2007)
20. Viganò, F., Colombetti, M.: Symbolic Model Checking of Institutions. In: Proceedings of the 9th International Conference on Electronic Commerce (ICEC 2007), pp. 35–44. ACM Press (2007)