Normative Systems

Mehdi Dastani

Utrecht University
The Netherlands

Joint works with Nils Bulling and Max Knobbout

Outline

Programming Normative Systems

Monitoring Norms

Analysing Normative Systems

Background (1)

Multi-agent systems as promising candidate to construct distributed software systems that:

- are open and consist of individual autonomous and heterogenous agents
  - Open: agents dynamically enter and exit the system
  - Autonomy: each agent pursues its own objective.
  - Heterogeneity: internal state and operations of an agent may not be known to external entities.
- The overall objective of such systems can be achieved by coordinating the behavior of the involved agents.

Background (2)

Norms (e.g., obligations & prohibitions) are a popular candidate for coordination. Norms are standards of behaviour which prescribe certain behavior.

- Norms can be about states, actions, or behaviors, e.g.,
  - state norms: *Maximum length of papers is 15 pages*.
  - action norms: *PC members should not review own papers*.
  - behavior norms: *Reviews should be delivered in time.*
- Norm can be enforced by means of rewards/sanctions or regimented.
  - Enforcement: *sanction page limit violation by additional fees*.
  - Regimentation: *prevent reviewing conflicting papers*.
  - Enforcement: *blacklist PC members with late reviews*.

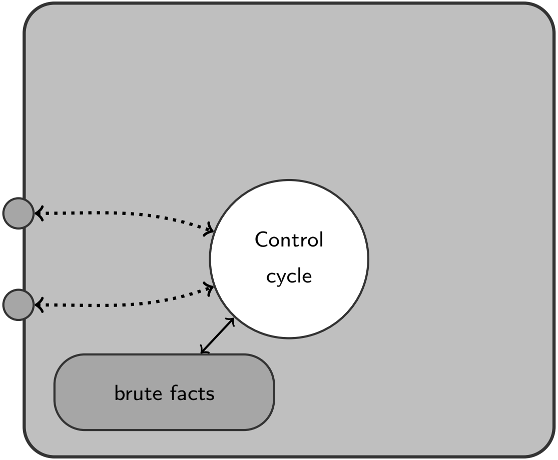
Background (3)

The development of normative systems requires implementation, verification, and analysis of norms.

- Norm implementation: design and develop a programming language that support the implementation of norms and rewards/sanctions.
- Norm Enforcement/Regimentation: monitors to observe norm violations.
- Norm analysis: use game theoretic tools to determine if a norm program can ensure the designer's objectives given some information about the involved agents.

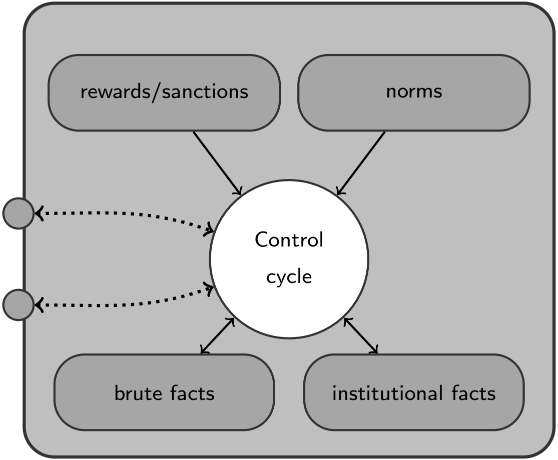
Programming Normative Systems

Coordination Component (IAT 2009, JLC 2011)



- Brute facts model the domain specific (environment) state, including action specifications;
- Agents modify brute facts by performing actions;
- Control cycle monitors agents actions and realizes their effects.

Coordination Component with Norms



- Brute facts model the domain specific (environment) state, including action specifications;
- Agents modify brute facts by performing actions;
- Ideal brute state described by norm;
- Active norms and norm violations stored by institutional facts;
- Norm obedience/violation might lead to rewards/sanctions;
- Control cycle monitors agents actions and realizes their effects in the context of the norms and rewards/sanctions.

Norms

- State-based norms (JLC 2011):

$F(\phi)$	Prohibited to achieve ϕ states
$O(\phi)$	Obliged to achieve ϕ states

- Action-based norms (IJCAI 2011):

$F(\phi, \alpha)$	Prohibited to perform action α in ϕ states
$O(\phi, \alpha)$	Obliged to perform action α in ϕ states

- Behaviour-based norms (IJCAI 2013):

$(cond, F(\phi), d)$	if $cond$, then prohibited to achieve ϕ before d
$(cond, O(\phi), d)$	if $cond$, then obliged to achieve ϕ before d

Examples of Behaviour-based Norms

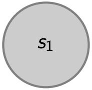
Norms:

```
reviewdue(R):                                        } label
   < phase(review) and assigned(R,P)                 } condition
     , O(review(R,P))                                } obligation
     , phase(collect)>                               } deadline

minreviews(P):
   < phase(submission) and paper(P)
     , O( nrReviews(P) >= 2 )
     , phase(collect)>

pagelimit(P):
   < phase(submission) and paper(P),
     , F(pages(P) > 15)
     , phase(review)>
```

Evolution of Obligations

Recall norm:

```
reviewdue(R):
< phase(review) and assigned(R,P), O(review(R,P)), phase(collect) >
```

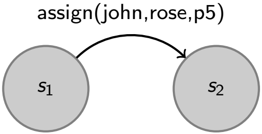


brute facts	:	`{ phase(review), paper(p5) }`
inst. facts	:	`{ rea(john,chair) }`

Evolution of Obligations

Recall norm:

```
reviewdue(R):
< phase(review) and assigned(R,P), O(review(R,P)), phase(collect) >
```



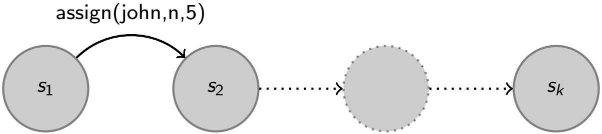
brute facts	:	{ `phase(review)`, `paper(p5)`, **assigned(rose,p5)** }
inst. facts	:	{ `rea(john,chair)` ,
		(reviewdue(rose) , **O(review(rose,p5))** , **phase(collect))** }

Evolution of Obligations

Recall norm:

`reviewdue(R):`

`< phase(review) and assigned(R,P), O(review(R,P)), phase(collect) >`



brute facts : `{ phase(review), paper(p5), assigned(rose,p5) }`

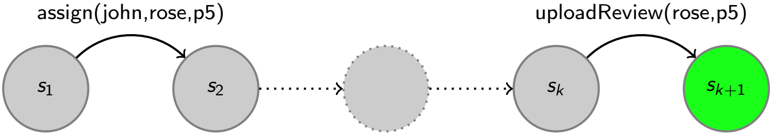
inst. facts : `{ rea(john,chair) ) ,`

(reviewdue(rose) , O(review(rose,p5)) , phase(collect)) }

Evolution of Obligations

Recall norm:

```
reviewdue(R):
< phase(review) and assigned(R,P), O(review(R,P)), phase(collect) >
```



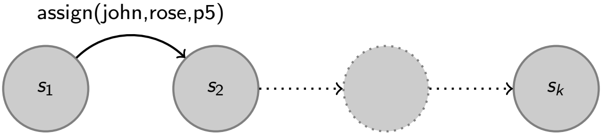
brute facts : { `phase(review)`, `paper(p5)`, `assigned(rose,p5)`, **review(rose,p5)** }

inst. facts : { `rea(john,chair)`, **obey(reviewdue(rose))** }

Evolution of Obligations

Recall norm:

```
reviewdue(R):
< phase(review) and assigned(R,P), O(review(R,P)), phase(collect) >
```

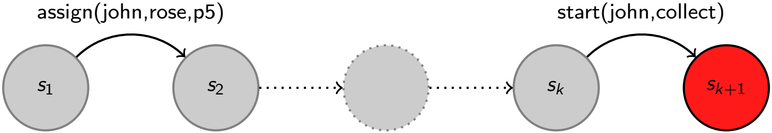


brute facts	:	`{ phase(review), paper(p5), assigned(rose,p5) }`
inst. facts	:	`{ rea(john,chair) ) ,`
		(reviewdue(rose) , **O(review(rose,p5))** , **phase(collect))** }

Evolution of Obligations

Recall norm:

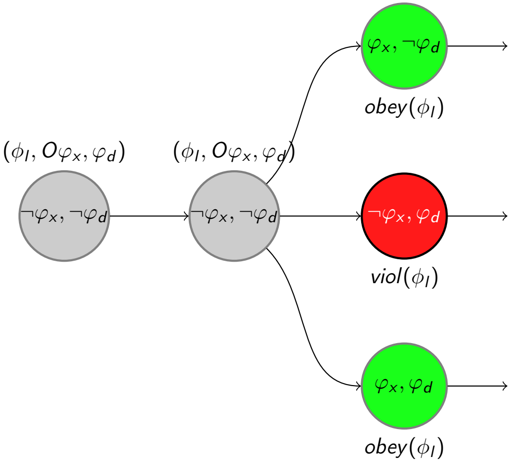
```
reviewdue(R):
< phase(review) and assigned(R,P), O(review(R,P)), phase(collect) >
```



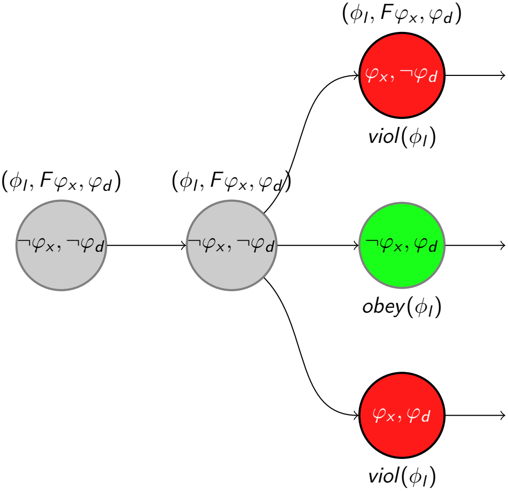
brute facts : { **phase(collect)**, `paper(p5)`, `assigned(rose,p5)` }

inst. facts : { `rea(john,chair)`, **viol(reviewdue(rose))** }

Behavior of an Obligation Summarized



Behavior of a Prohibition Summarized



Normative System Configuration

Definition (Configuration)

The state of a coordination component is a tuple $\langle \sigma_b, \sigma_i, \Delta \rangle$ with:

- σ_b a set of ground first-order atoms, the brute state;
- σ_i a set of ground first-order atoms, the institutional state;
- Δ a set of norms (Static);

For simplicity we ignore static Δ and present a configuration as:

$$\langle \sigma_b, \sigma_i \rangle$$

Triggering Norms

Recall norm:

```
reviewdue(R):
< phase(review) and assigned(R,P), O(review(R,P)), phase(collect) >
```

Definition (Norm Instantiation)

Given:

- $ns = \phi_l(\overline{v_1}) : \langle \varphi_c(\overline{v_2}), \mathbb{P}(\varphi_x(\overline{v_3})), \varphi_d(\overline{v_4}) \rangle$
- $\mathbb{P}$ either O or F
- $\overline{v_1}, \ldots, \overline{v_4}$ the sets of variables occurring in the formulae
- ground substitution θ

The function *inst* for instantiating *ns* with θ is defined as:

$$inst(ns, \theta) = (\ \phi_l(\overline{v_1})\theta\ ,\ \mathbb{P}(\varphi_x(\overline{v_3})\theta)\ ,\ \varphi_d(\overline{v_4})\theta\ )$$

Triggering Norms

Transition Rule

Given ground substitution θ, the rule for triggering of norms is defined as:

$$\frac{ns = (\phi_l : \langle \varphi_c, \mathbb{P}(\varphi_x), \varphi_d \rangle) \in \Delta \quad \sigma_b \models \varphi_c\theta \quad ni = inst(ns, \theta)}{\langle \sigma_b, \sigma_i \rangle \longrightarrow \langle \sigma_b, \sigma_i \cup \{ni\} \rangle}$$

Triggering Norms

Transition Rule

Given ground substitution θ, the rule for triggering of norms is defined as:

$$\frac{ns = (\phi_l : \langle \varphi_c, \mathbb{P}(\varphi_x), \varphi_d \rangle) \in \Delta \quad \sigma_b \models \varphi_c \theta \quad ni = inst(ns, \theta)}{\langle \sigma_b, \sigma_i \rangle \longrightarrow \langle \sigma_b, \sigma_i \cup \{ni\} \rangle}$$

- When the condition of a norm is satisfied

Triggering Norms

Transition Rule

Given ground substitution θ, the rule for triggering of norms is defined as:

$$\frac{ns = (\phi_l : \langle \varphi_c, \mathbb{P}(\varphi_x), \varphi_d \rangle) \in \Delta \quad \sigma_b \models \varphi_c\theta \quad ni = inst(ns, \theta)}{\langle \sigma_b, \sigma_i \rangle \longrightarrow \langle \sigma_b, \sigma_i \cup \{ni\} \rangle}$$

- We instantiate it and add it to the institutional facts

Monitoring Obligations

Transition Rule

The rule for violation of an obligation is defined as follows:

$$\frac{(\phi_l, O(\varphi_x), \varphi_d) \in \sigma_i \quad \sigma_b \not\models \varphi_x \quad \sigma_b \models \varphi_d}{\langle \sigma_b, \sigma_i \rangle \longrightarrow \langle \sigma_b, \ (\sigma_i \ \setminus \{ni\}) \ \cup \{viol(\phi_l) \ \}\rangle}$$

Monitoring Obligations

Transition Rule

The rule for violation of an obligation is defined as follows:

$$\frac{(\phi_l, O(\varphi_x), \varphi_d) \in \sigma_i \quad \sigma_b \not\models \varphi_x \quad \sigma_b \models \varphi_d}{\langle \sigma_b, \sigma_i \rangle \longrightarrow \langle \sigma_b,\ (\sigma_i \ \setminus \{ni\})\ \cup \{viol(\phi_l)\ \}\rangle}$$

- When an obligation is violated

Monitoring Obligations

Transition Rule

The rule for violation of an obligation is defined as follows:

$$\frac{(\phi_l, O(\varphi_x), \varphi_d) \in \sigma_i \quad \sigma_b \not\models \varphi_x \quad \sigma_b \models \varphi_d}{\langle \sigma_b, \sigma_i \rangle \longrightarrow \langle \sigma_b, \ (\sigma_i \ \setminus \{ni\}) \ \cup \{viol(\phi_l) \ \}\rangle}$$

- We remove it from the institutional facts

Monitoring Obligations

Transition Rule

The rule for violation of an obligation is defined as follows:

$$\frac{(\phi_l, O(\varphi_x), \varphi_d) \in \sigma_i \quad \sigma_b \not\models \varphi_x \quad \sigma_b \models \varphi_d}{\langle \sigma_b, \sigma_i \rangle \longrightarrow \langle \sigma_b,\ (\sigma_i \ \setminus \{ni\}) \cup \{viol(\phi_l)\ \} \rangle}$$

- We record its violation to the institutional facts

Monitoring Obligations

Transition Rule

The rule for fulfillment of an obligation is defined as follows:

$$\frac{(\phi_l, O(\varphi_x), \varphi_d) \in \sigma_i \quad \sigma_b \models \varphi_x}{\langle \sigma_b, \sigma_i \rangle \longrightarrow \langle \sigma_b,\ (\sigma_i \setminus \{ni\})\ \cup \{obey(\phi_l)\} \rangle}$$

Monitoring Obligations

Transition Rule

The rule for fulfillment of an obligation is defined as follows:

$$\frac{(\phi_l, O(\varphi_x), \varphi_d) \in \sigma_i \quad \sigma_b \models \varphi_x}{\langle \sigma_b, \sigma_i \rangle \longrightarrow \langle \sigma_b,\ (\sigma_i \setminus \{ni\})\ \cup \{obey(\phi_l)\} \rangle}$$

- When an obligation is fulfilled

Monitoring Obligations

Transition Rule

The rule for fulfillment of an obligation is defined as follows:

$$\frac{(\phi_l, O(\varphi_x), \varphi_d) \in \sigma_i \quad \sigma_b \models \varphi_x}{\langle \sigma_b, \sigma_i \rangle \longrightarrow \langle \sigma_b,\ (\sigma_i \setminus \{ni\})\ \cup \{obey(\phi_l)\} \rangle}$$

- We remove it from the institutional facts

Monitoring Obligations

Transition Rule

The rule for fulfillment of an obligation is defined as follows:

$$\frac{(\phi_l, O(\varphi_x), \varphi_d) \in \sigma_i \quad \sigma_b \models \varphi_x}{\langle \sigma_b, \sigma_i \rangle \longrightarrow \langle \sigma_b,\ (\sigma_i \setminus \{ni\}) \cup \{obey(\phi_l)\} \rangle}$$

- We record its fulfilment to the institutional facts

Properties

Proposition

Detached obligations remain in force until obeyed or violated.

I.e., for every trace $\langle \sigma_b^0, \sigma_i^0 \rangle \rightarrow^* \langle \sigma_b^n, \sigma_i^n \rangle$ *with* $\sigma_b^j \not\models \phi_x$ *and* $\sigma_b^j \not\models \phi_d$ *for* $0 \leq j \leq n$*, if* $(\phi_l, O(\phi_x), \phi_d) \in \sigma_i^0$*, then* $(\phi_l, O(\phi_x), \phi_d) \in \sigma_i^n$*.*

Proposition

Violation of detached obligations are recorded in deadline states.

Proposition

Violation is inevitable in case of conflicting norms.

Properties

Proposition

Detached obligations remain in force until obeyed or violated.

Proposition

Violation of detached obligations are recorded in deadline states.

I.e., for every trace $\langle \sigma_b^0, \sigma_i^0 \rangle \rightarrow^* \langle \sigma_b^n, \sigma_i^n \rangle$ *with* $\sigma_b^j \not\models \phi_x$ *and* $\sigma_b^j \not\models \phi_d$ *for* $0 \leq j < n$ *and* $\sigma_b^n \not\models \phi_x$ *and* $\sigma_b^n \models \phi_d$*, if* $(\phi_l, O(\phi_x), \phi_d) \in \sigma_i^0$*, then* $\sigma_i^n \models viol(\phi_l)$ *and* $(\phi_l, O(\phi_x), \phi_d) \notin \sigma_i^n$.

Proposition

Violation is inevitable in case of conflicting norms.

Properties

Proposition

Detached obligations remain in force until obeyed or violated.

Proposition

Violation of detached obligations are recorded in deadline states.

Proposition

Violation is inevitable in case of conflicting norms.

I.e., for every trace $\langle \sigma_b^0, \sigma_i^0 \rangle \rightarrow^* \langle \sigma_b^n, \sigma_i^n \rangle$ *with* $(\phi_l, O(\phi_x), \phi_d) \in \sigma_i^0$ *and* $(\phi_{l'}, F(\phi_x), \phi_{d'}) \in \sigma_i^0$, *if* $\sigma_b^n \models \phi_d$ *and* $\sigma_b^j \not\models \phi_{d'}$ *for* $0 \leq j \leq n$ *then* $\exists 0 \leq k \leq n : \sigma_i^k \models viol(\phi_l)$ *or* $\sigma_i^k \models viol(\phi_{l'})$.

Monitoring Norms

Motivation

Existing work on normative multi-agent systems typically assume perfect monitoring.

We want to develop a very general framework in order to . . .

- . . . characterize monitors.
- . . . reason about monitors.
- . . . study the relation between monitors and norms.

Runs, Norms, Monitors

Given an (interpreted) transition system $(Q, \rightarrow, q_0, v)$, we define:

Definition

$\mathcal{R}$ as the set of runs, where $\mathcal{R} = \{q_0 q_1 \ldots \in Q^\omega \mid \forall n, q_n \rightarrow q_{n+1}\}$

Runs, Norms, Monitors

Given an (interpreted) transition system $(Q, \rightarrow, q_0, v)$, we define:

Definition

$\mathcal{R}$ as the set of runs, where $\mathcal{R} = \{q_0 q_1 \ldots \in Q^\omega \mid \forall n, q_n \rightarrow q_{n+1}\}$

Definition

$\mathcal{N} \subseteq \mathcal{R}$ as a norm, which denotes a set of desired runs. Moreover, we say that a run r is a $\mathcal{N}$-violation iff $r \notin \mathcal{N}$.

Runs, Norms, Monitors

Given an (interpreted) transition system $(Q, \rightarrow, q_0, v)$, we define:

Definition

$\mathcal{R}$ as the set of runs, where $\mathcal{R} = \{q_0 q_1 \ldots \in Q^\omega \mid \forall n, q_n \rightarrow q_{n+1}\}$

Definition

$\mathcal{N} \subseteq \mathcal{R}$ as a norm, which denotes a set of desired runs. Moreover, we say that a run r is a $\mathcal{N}$-violation iff $r \notin \mathcal{N}$.

Definition

Given set $\mathcal{R}$, a monitor m is a function from $\mathcal{R}$ to $\mathcal{P}(\mathcal{R})$.

Monitor Types

Some extreme cases:

- For all $r \in \mathcal{R}$ $m(r) = \{r\}$ (perfect observation).
- For all $r \in \mathcal{R}$ $m(r) = \mathcal{R}$ (no observation).

Monitor Types

Some extreme cases:

- For all $r \in \mathcal{R}$ $m(r) = \{r\}$ (perfect observation).
- For all $r \in \mathcal{R}$ $m(r) = \mathcal{R}$ (no observation).

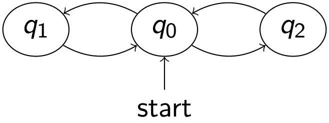
Definition

Let m be a monitor over $\mathcal{R}$. We say that m is

- broken iff there exists a $r \in \mathcal{R}$ such that $m(r) = \emptyset$.
- correct iff for all $r \in \mathcal{R}$ we have $r \in m(r)$.
- ideal iff for all $r \in \mathcal{R}$ we have $m(r) = \{r\}$.

Example

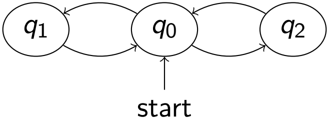
An example:



- $\mathcal{N} = \{(q_0\, q_1)^\omega\}$
- $m(r) = \{r' \in \mathcal{R} \mid r[1] = r'[1]\}$

Example

An example:



- $\mathcal{N} = \{(q_0\, q_1)^\omega\}$
- $m(r) = \{r' \in \mathcal{R} \mid r[1] = r'[1]\}$

Note that this monitor is correct, but not ideal.

Logical setting

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \varphi \mathcal{U} \varphi \mid \bigcirc\varphi$$

Semantics:

- $\mathfrak{I}, r \models p$ iff $p \in v(r[0])$
- $\mathfrak{I}, r \models \neg\varphi$ iff $\mathfrak{I}, r \not\models \varphi$
- $\mathfrak{I}, r \models \varphi \vee \psi$ iff $\mathfrak{I}, r \models \varphi$ or $\mathfrak{I}, r \models \psi$
- $\mathfrak{I}, r \models \bigcirc\varphi$ iff $\mathfrak{I}, r[1, \infty] \models \varphi$
- $\mathfrak{I}, r \models \varphi \mathcal{U} \psi$ iff $i \geq 0$ such that $\mathfrak{I}, r[i, \infty] \models \psi$ and for all $0 \leq k < i$, $\mathfrak{I}, r[k, \infty] \models \varphi$

Moreover, $\mathfrak{I}, R \models \varphi$ iff $\forall r \in R : \mathfrak{I}, r \models \varphi$.

Definition

Let χ be an *LTL*-formula and $\mathfrak{I}$ be a transition system. The χ-norm in $\mathfrak{I}$ is the set $\{r \in \mathcal{R} \mid \mathfrak{I}, r \models \chi\}$. We simply write χ to refer to this norm.

Monitoring LTL-Norms

Definition

Let χ be an *LTL*-formula and $\mathfrak{I}$ be a transition system. The χ-norm in $\mathfrak{I}$ is the set $\{r \in \mathcal{R} \mid \mathfrak{I}, r \models \chi\}$. We simply write χ to refer to this norm.

Definition

We say that monitor m on input r detects a

- χ-violation iff $\mathfrak{I}, m(r) \models \neg\chi$;
- χ-compliance iff $\mathfrak{I}, m(r) \models \chi$; and
- χ-indifference iff both $\mathfrak{I}, m(r) \not\models \chi$ and $\mathfrak{I}, m(r) \not\models \neg\chi$.

Characterizing Monitors based on LTL-Norms

Definition

Let $\mathfrak{I}$ be a transition system, χ a norm, and m a monitor. We say that m makes a χ-classification error on r iff

- m detects a χ-violation on r and r is not a χ-violation (false negative); or
- m detects a χ-compliance on r and r is a χ-violation (false positive).

Characterizing Monitors based on LTL-Norms

Definition

Let $\mathfrak{I}$ be a transition system, χ a norm, and m a monitor. We say that m makes a χ-classification error on r iff

- m detects a χ-violation on r and r is not a χ-violation (false negative); or
- m detects a χ-compliance on r and r is a χ-violation (false positive).

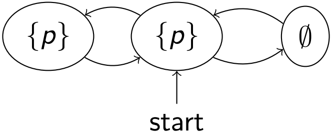
Definition

Let $\mathfrak{I}$ be a transition system, χ a norm and m a monitor. We say that m is

- χ-sound in $\mathfrak{I}$ iff for all $r \in \mathcal{R}$: $\mathfrak{I}, m(r) \models \neg\chi \Rightarrow \mathfrak{I}, r \models \neg\chi$.
- χ-complete in $\mathfrak{I}$ iff for all $r \in \mathcal{R}$: $\mathfrak{I}, r \models \neg\chi \Rightarrow \mathfrak{I}, m(r) \models \neg\chi$.
- χ-sufficient in $\mathfrak{I}$ iff m is χ-sound and χ-complete.

Example

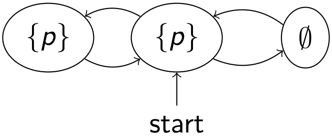
Back to the example:



- $\chi = \Box p$
- $m(r) = \{r' \in \mathcal{R} \mid r[1] = r'[1]\}$

Example

Back to the example:



- $\chi = \Box p$
- $m(r) = \{r' \in \mathcal{R} \mid r[1] = r'[1]\}$

m is χ-sound: for all r: $\mathfrak{I}, m(r) \models \neg\Box p \Rightarrow \mathfrak{I}, r \models \neg\Box p$.
m is not χ-complete: exists r: $\mathfrak{I}, r \models \neg\Box p$ and $\mathfrak{I}, m(r) \not\models \neg\Box p$

LTL-Monitors & Composed Monitors

Definition

Let φ be an LTL-formula and $\mathfrak{I}$ be a transition system. The φ-monitor over $\mathfrak{I}$ is the function $m_\varphi : \mathcal{R} \to \mathcal{P}(\mathcal{R})$ defined as follows: $m_\varphi(r) := \{r' \in \mathcal{R} \mid \mathfrak{I}, r \models \varphi \text{ iff } \mathfrak{I}, r' \models \varphi\}$.

LTL-Monitors & Composed Monitors

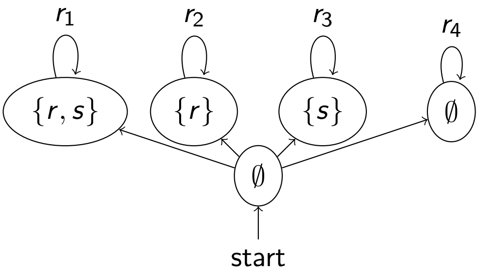
Definition

Let φ be an LTL-formula and $\mathfrak{I}$ be a transition system. The φ-monitor over $\mathfrak{I}$ is the function $m_\varphi : \mathcal{R} \to \mathcal{P}(\mathcal{R})$ defined as follows: $m_\varphi(r) := \{r' \in \mathcal{R} \mid \mathfrak{I}, r \models \varphi \text{ iff } \mathfrak{I}, r' \models \varphi\}$.

Definition

Let $m_1, m_2 : \mathcal{R} \to \mathcal{P}(\mathcal{R})$ be two monitors. We define the monitor $m \oplus m' : \mathcal{R} \to \mathcal{P}(\mathcal{R})$ as follows: $m \oplus m'(r) := m(r) \cap m'(r)$.

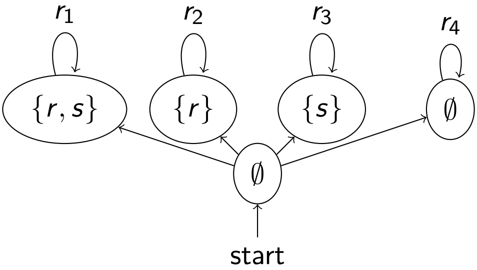
Example



Let $\chi = \bigcirc((r \wedge \neg s) \vee (\neg r \wedge s))$. Let $m_{\bigcirc r}$ and $m_{\bigcirc s}$ be two LTL-monitors. We have the following:

- $(m_{\bigcirc r} \oplus m_{\bigcirc s})(r_2) = \{r_2\}$
- $m_{(\bigcirc r \wedge \bigcirc s)}(r_2) = \{r_2, r_3, r_4\}$

Example



Let $\chi = \bigcirc((r \wedge \neg s) \vee (\neg r \wedge s))$. Let $m_{\bigcirc r}$ and $m_{\bigcirc s}$ be two LTL-monitors. We have the following:

- $(m_{\bigcirc r} \oplus m_{\bigcirc s})(r_2) = \{r_2\}$
- $m_{(\bigcirc r \wedge \bigcirc s)}(r_2) = \{r_2, r_3, r_4\}$

Only composed monitor $(m_{\bigcirc r} \oplus m_{\bigcirc s})$ is χ-sufficient.

Theorem

Let m_φ and m_ψ be two (or more) LTL-monitors, χ be an LTL-norm, and $\mathfrak{I}$ be a transition system. Let $\Sigma = \{\varphi \wedge \psi, \varphi \wedge \neg\psi, \neg\varphi \wedge \psi, \neg\varphi \wedge \neg\psi\}$. Then, the following statements are equivalent:

(a) *$m_\varphi \oplus m_\psi$ is χ-sufficient over $\mathfrak{I}$.*

(b) *for all $\xi \in \Sigma$, if $\neg\chi \wedge \xi$ is satisfiable on $\mathfrak{I}$ then $\mathfrak{I} \models \xi \rightarrow \neg\chi$.*

(c) *exists $\Sigma' \subseteq \Sigma$ such that $\mathfrak{I} \models (\bigvee_{\xi \in \Sigma'} \xi) \leftrightarrow \neg\chi$.*

Analysing Normative Systems

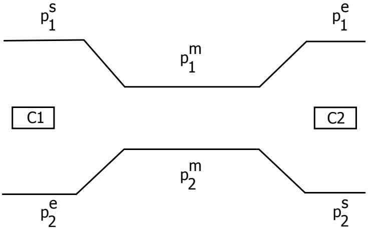
Norms as Mechanism Design (AAMAS 2011, IJCAI 2011)

Can specific behaviours be enforced by a normative environment program if agents follow their subjective preferences?

I.e.,

Does a set of norms and sanctions/rewards implements specific social choice functions (designer's objectives) in specific equilibria?

Example: A Road Scenario



- $s0$: the cars are at their starting positions, i.e., $p_1^s \wedge p_2^s$
- $s1$: car $C1$ is at ending and car $C2$ at starting position,i.e., $p_1^e \wedge p_2^s$
- $s2$: car $C1$ is at starting and car $C2$ at ending position,i.e., $p_1^s \wedge p_2^e$
- $s3$: the cars are jammed in the middle of the road,i.e., $p_1^m \wedge p_2^m$
- $s4$: the cars are at their ending positions, i.e., $p_1^e \wedge p_2^e$

Action Specification

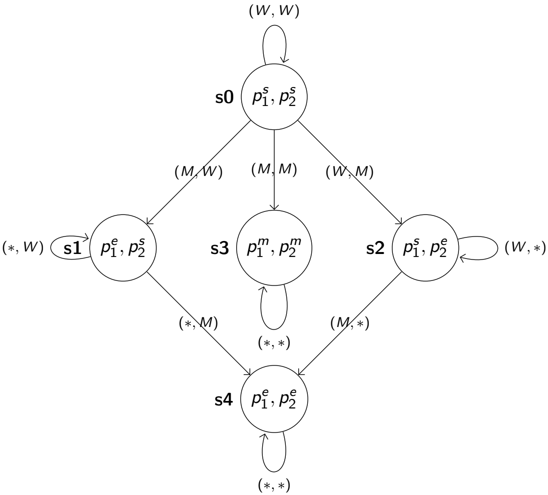
- Each car has two actions.
  - Move (M)
  - Wait (W), and
  - $* \in \{M, W\}$.
- Joint actions are specified in terms of their pre- and post-conditions.

$$
\begin{array}{lll}
\{p_1^s, p_2^s\} & (M, W) & \{p_1^e, p_2^s\} \\
\{p_1^s, p_2^s\} & (W, M) & \{p_1^s, p_2^e\} \\
\{p_1^s, p_2^s\} & (M, M) & \{p_1^m, p_2^m\} \\
\{p_1^s, p_2^e\} & (M, *) & \{p_1^e, p_2^e\} \\
\{p_1^e, p_2^s\} & (*, M) & \{p_1^e, p_2^e\}
\end{array}
$$

Other actions do not cause state change.

Transition System without Norms

Possible behaviours of the cars can be described by the following transition system.



Preferences: Cars & System Designer

Preferences profiles, λ_1 (egoistic) and λ_2 (social), are represented as lists of LTL formulae.

$$\begin{aligned}
\lambda_1 = \{ \quad & [(Xp_1^e \wedge \Box\neg s_1, 3), (\Diamond p_1^e \wedge \Box\neg s_1, 2), (\Box\neg s_1, 1), (\top, 0)] \ , \\
& [(Xp_2^e \wedge \Box\neg s_2, 3), (\Diamond p_2^e \wedge \Box\neg s_2, 2), (\Box\neg s_2, 1), (\top, 0)] \ \}
\end{aligned}$$

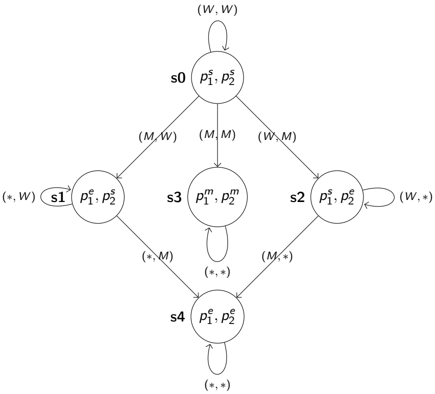
$$\begin{aligned}
\lambda_2 = \{ \quad & [(X(p_1^e \wedge p_2^e) \wedge \Box\neg s_1, 3), (Xp_1^e \wedge \Box\neg s_1, 2), (\Box\neg s_1, 1), (\top, 0)] \ , \\
& [(X(p_1^e \wedge p_2^e) \wedge \Box\neg s_2, 3), (Xp_2^e \wedge \Box\neg s_2, 2), (\Box\neg s_2, 1), (\top, 0)] \ \}
\end{aligned}$$

The preference of the system designer is represented by the following social choice function:

$$SCF(\lambda_i) = Xp_1^e \vee (Xp_1^e \wedge \Diamond p_2^e)$$

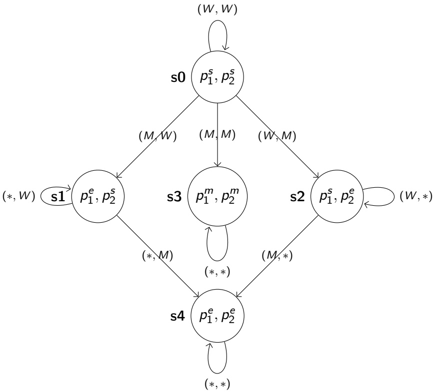
The designer prefers that first car reaches its end position directly.

Equilibrium Analysis



$1 \backslash 2$	$M**$	$WM*$	$WW*$
$M**$	03^*	014^*	011^*
$W*M$	024^*	00^*	00^*
$W*W$	022^*	00^*	00^*

Equilibrium Analysis

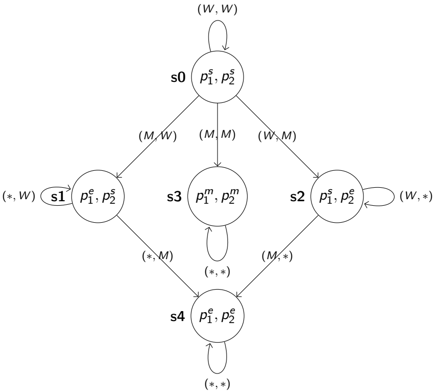


$1\backslash 2$	$M**$	$WM*$	$WW*$
$M**$	03^*	014^*	011^*
$W*M$	024^*	00^*	00^*
$W*W$	022^*	00^*	00^*

$1\backslash 2$	$M**$	$WM*$	$WW*$
$M**$	$1\backslash 1$	$3\backslash 2$	$3\backslash 1$
$W*M$	$2\backslash 3$	$1\backslash 1$	$1\backslash 1$
$W*W$	$1\backslash 3$	$1\backslash 1$	$1\backslash 1$

$$\lambda_1 = \{ \quad \begin{array}{l} [(Xp_1^e \wedge \Box\neg s_1, 3), (\Diamond p_1^e \wedge \Box\neg s_1, 2), (\Box\neg s_1, 1), (\top, 0)] \ , \\ [(Xp_2^e \wedge \Box\neg s_2, 3), (\Diamond p_2^e \wedge \Box\neg s_2, 2), (\Box\neg s_2, 1), (\top, 0)] \ \} \end{array}$$

Equilibrium Analysis

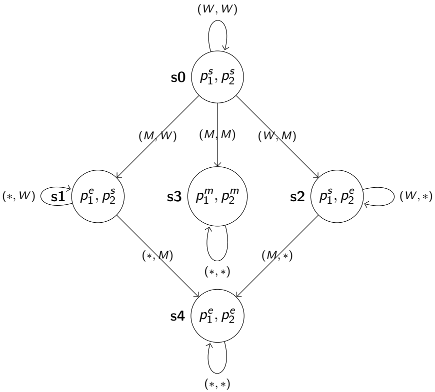


$1\backslash 2$	$M**$	$WM*$	$WW*$
$M**$	03^*	014^*	011^*
$W*M$	024^*	00^*	00^*
$W*W$	022^*	00^*	00^*

$1\backslash 2$	$M**$	$WM*$	$WW*$
$M**$	$1\backslash 1$	$3\backslash 2$	$3\backslash 1$
$W*M$	$\cancel{2\backslash 3}$	$1\backslash 1$	$1\backslash 1$
$W*W$	$1\backslash 3$	$1\backslash 1$	$1\backslash 1$

$$SCF(\lambda_i) = Xp_1^e \vee (Xp_1^e \wedge \Diamond p_2^e)$$

Equilibrium Analysis

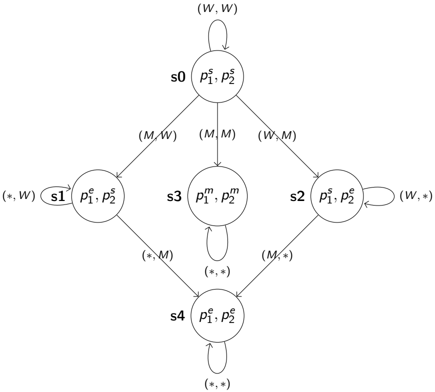


$1\backslash 2$	$M**$	$WM*$	$WW*$
$M**$	03^*	014^*	011^*
$W*M$	024^*	00^*	00^*
$W*W$	022^*	00^*	00^*

$1\backslash 2$	$M**$	$WM*$	$WW*$
$M**$	$1\backslash 1$	$2\backslash 1$	$2\backslash 1$
$W*M$	$1\backslash 2$	$1\backslash 1$	$1\backslash 1$
$W*W$	$1\backslash 2$	$1\backslash 1$	$1\backslash 1$

$$\lambda_2 = \{ \quad [(X(p_1^e \wedge p_2^e) \wedge \Box\neg s_1, 3), (Xp_1^e \wedge \Box\neg s_1, 2), (\Box\neg s_1, 1), (\top, 0)]\ ,$$
$$[(X(p_1^e \wedge p_2^e) \wedge \Box\neg s_2, 3), (Xp_2^e \wedge \Box\neg s_2, 2), (\Box\neg s_2, 1), (\top, 0)]\ \}$$

Equilibrium Analysis

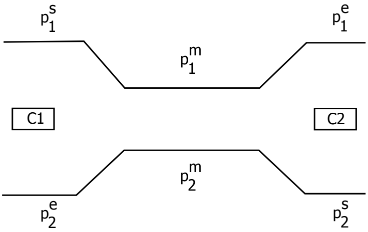


$1\backslash 2$	$M**$	$WM*$	$WW*$
$M**$	03^*	014^*	011^*
$W*M$	024^*	00^*	00^*
$W*W$	022^*	00^*	00^*

$1\backslash 2$	$M**$	$WM*$	$WW*$
$M**$	$1\backslash 1$	$2\backslash 1$	$2\backslash 1$
$W*M$	$\cancel{1\backslash 2}$	$1\backslash 1$	$1\backslash 1$
$W*W$	$\cancel{1\backslash 2}$	$1\backslash 1$	$1\backslash 1$

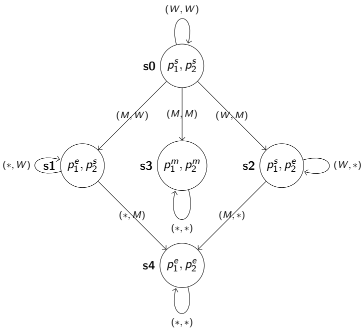
$$SCF(\lambda_i) = Xp_1^e \vee (Xp_1^e \wedge \Diamond p_2^e)$$

Example: Introducing Norms in the Road Scenario



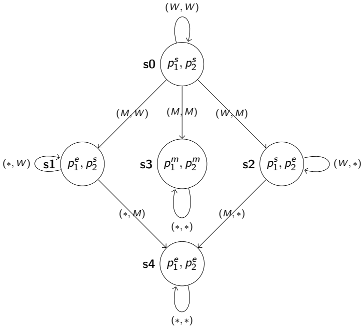
- The second car is prohibited to move in the start position, otherwise sanction s_2 will be imposed, i.e.,
$$F(p_1^s \wedge p_2^s \ , \ (*, W) \ , \ s_2)$$
- It is prohibited that cars wait on each other in the start position, otherwise sanction s_1 will be imposed, i.e.,
$$F(p_1^s \wedge p_2^s \ , \ (W, W) \ , \ s_1)$$

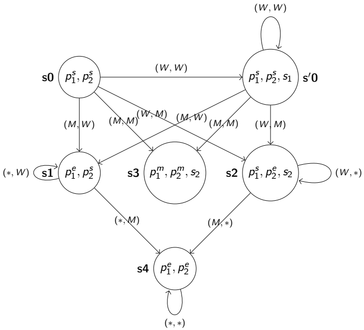
Norms and Norm Updates



- The second car is prohibited to move in the start position, otherwise sanction s_2 will be imposed, i.e.,
$$F(p_1^s \wedge p_2^s \ , \ (*, W) \ , \ s_2)$$
- It is prohibited that cars wait on each other in the start position, otherwise sanction s_1 will be imposed, i.e.,
$$F(p_1^s \wedge p_2^s \ , \ (W, W) \ , \ s_1)$$

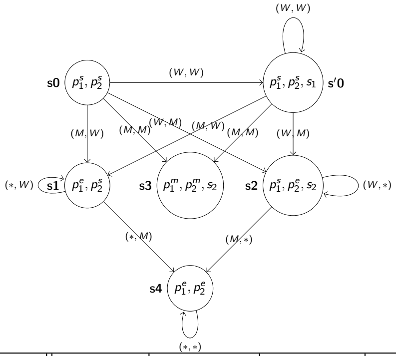
Norms and Norm Updates





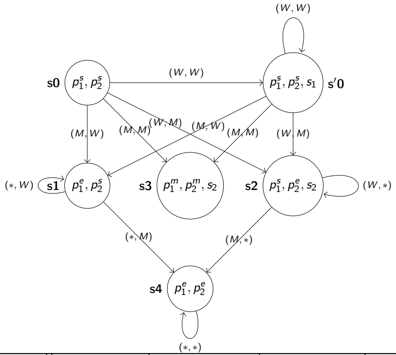
- The second car is prohibited to move in the start position, otherwise sanction s_2 will be imposed, i.e.,
$$F(p_1^s \wedge p_2^s\ ,\ (*, W)\ ,\ s_2)$$
- It is prohibited that cars wait on each other in the start position, otherwise sanction s_1 will be imposed, i.e.,
$$F(p_1^s \wedge p_2^s\ ,\ (W, W)\ ,\ s_1)$$

Equilibrium Analysis



$1\backslash 2$	$M***$	$WWW*$	$WMM*$	$WMW*$
$M***$	03^*	011^*	014^*	011^*
$WW*M$	024^*	$00'0'^*$	$00'24^*$	$00'24^*$
$WW*W$	022^*	$00'0'^*$	$00'22^*$	$00'22^*$
$WM*M$	024^*	$00'11^*$	$00'3^*$	$00'3^*$
$WM*W$	022^*	$00'11^*$	$00'3^*$	$00'3^*$

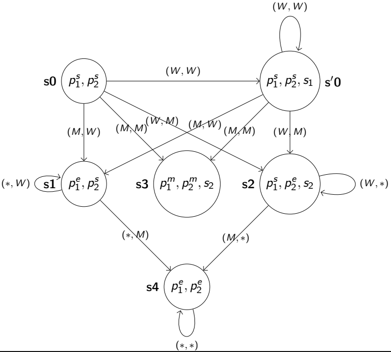
Equilibrium Analysis



$1\backslash 2$	$M{*}{*}{*}$	$WWW{*}$	$WMM{*}$	$WMW{*}$
$M{*}{*}{*}$	$1\backslash 0$	$3\backslash 1$	$3\backslash 2$	$3\backslash 1$
$WW{*}M$	$2\backslash 0$	$0\backslash 1$	$0\backslash 0$	$0\backslash 0$
$WW{*}W$	$1\backslash 0$	$0\backslash 1$	$0\backslash 0$	$0\backslash 0$
$WM{*}M$	$2\backslash 0$	$0\backslash 1$	$0\backslash 0$	$0\backslash 0$
$WM{*}W$	$1\backslash 0$	$0\backslash 1$	$0\backslash 0$	$0\backslash 0$

$$\lambda_1 = \{ \quad [(Xp_1^e \wedge \Box\neg s_1, 3), (\Diamond p_1^e \wedge \Box\neg s_1, 2), (\Box\neg s_1, 1), (\top, 0)] \;,$$
$$[(Xp_2^e \wedge \Box\neg s_2, 3), (\Diamond p_2^e \wedge \Box\neg s_2, 2), (\Box\neg s_2, 1), (\top, 0)] \; \}$$

Equilibrium Analysis



$1\backslash 2$	$M***$	$WWW*$	$WMM*$	$WMW*$
$M***$	$1\backslash 0$	$2\backslash 1$	$2\backslash 1$	$2\backslash 1$
$WW*M$	$1\backslash 0$	$0\backslash 1$	$0\backslash 0$	$0\backslash 0$
$WW*W$	$1\backslash 0$	$0\backslash 1$	$0\backslash 0$	$0\backslash 0$
$WM*M$	$1\backslash 0$	$0\backslash 1$	$0\backslash 0$	$0\backslash 0$
$WM*W$	$1\backslash 0$	$0\backslash 1$	$0\backslash 0$	$0\backslash 0$

$$\lambda_2 = \{ \quad [(X(p_1^e \wedge p_2^e) \wedge \Box\neg s_1, 3), (Xp_1^e \wedge \Box\neg s_1, 2), (\Box\neg s_1, 1), (\top, 0)] \ ,$$
$$[(X(p_1^e \wedge p_2^e) \wedge \Box\neg s_2, 3), (Xp_2^e \wedge \Box\neg s_2, 2), (\Box\neg s_2, 1), (\top, 0)] \ \}$$

Verification Problems

Let $\mathcal{M}$ be a transition system, N be a set of norms, $\mathcal{M} \upharpoonright N$ be the transition system $\mathcal{M}$ updated with N.

- ($Nash$, N)-implementation problem ($\mathrm{IP}_{\mathrm{N}}^{\mathrm{Nash}}$)
  Given a norm set N do the outcome paths of $\mathcal{M} \upharpoonright N$ satisfy SCF if agents follow $Nash$-equilibria strategy profiles?
- ($Nash$)-synthesis problem ($\mathrm{SP}^{\mathrm{Nash}}$)
  Is there a norm set M such that . . . ?

Verification Results

The following results are about state based norms and Nash equilibria!

Theorem (($Nash$, N)-implementation problem)

The problem $\mathrm{IP}_{\mathrm{N}}^{\mathrm{Nash}}$ *is* Π_2^P*-complete.*

Verification Results

The following results are about state based norms and Nash equilibria!

Theorem (($Nash$, N)-implementation problem)

The problem $\mathrm{IP}_{\mathrm{N}}^{\mathrm{Nash}}$ is Π_2^P-complete.

Theorem (($Nash$)-Synthesis problem)

The problem $\mathrm{SP}^{\mathrm{Nash}}$ is Σ_3^P-complete.

Conclusions and Future Research

- Development of normative systems.
- Monitors and Norms.
- Game theoretic analysis of normative systems.
- Norms in standard software technology.
- Monitoring, imperfect monitors, and approximated norms.
- Distributed normative systems.