Social Laws for Multi-Agent Systems: Logic and Games

Lecture 2: Social Laws for Coordination

Thomas Ågotnes[1]

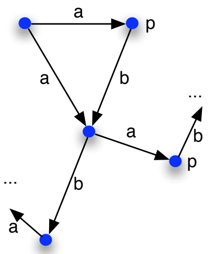
[1]Department of Information Science and Media Studies
University of Bergen, Norway

NII Tokyo 20 December 2011

Contents

Formal Models of Multi-Agent Systems

- The states are *global states*
- We label the transitions with the *name of the agent* that causes the transition by executing some *action*
- This assumes *asynchronous action*



Formally

An agent-labelled Kripke structure (over Φ) is a 6-tuple:

$$K = \langle S, S^0, R, Ag, \alpha, V \rangle, \text{ where}$$

- S is a finite, non-empty set of states,
- $S^0 \subseteq S$ ($S^0 \neq \emptyset$) is the set of initial states;
- $R \subseteq S \times S$ is a total (each state has a successor) binary transition relation on S;
- $Ag = \{1, \ldots, n\}$ is the set of agents;
- $\alpha : R \to Ag$ labels each transition in R with an agent
- $V : S \to 2^{\Phi}$ labels states with a set of propositional atoms

Note: simplifying assumption: single agent execute single action in each state (*interleaved concurrency*)

CTL: language

The language of CTL (CTL formulas) is defined as follows:

- Propositional atoms such as p or *started* are formulas
- Formulas can be combined using propositional connectives such as $\wedge$ (and), $\vee$ (or), $\neg$ (not), $\rightarrow$ (implication), etc.
- We can construct new formulas by putting *temporal connectives* in front of an existing formula. If φ and ψ are formulas, then the following as also formulas:

$\mathsf{E}\bigcirc\varphi$	on some path, φ is true next
$\mathsf{E}(\varphi\,\mathcal{U}\,\psi)$	on some path, φ until ψ
$\mathsf{E}\Diamond\varphi$	on some path, eventually φ
$\mathsf{E}\Box\varphi$	on some path, always φ
$\mathsf{A}\bigcirc\varphi$	on all paths, φ is true next
$\mathsf{A}(\varphi\,\mathcal{U}\,\psi)$	on all paths, φ until ψ
$\mathsf{A}\Diamond\varphi$	on all paths, eventually φ
$\mathsf{A}\Box\varphi$	on all paths, always φ

Contents

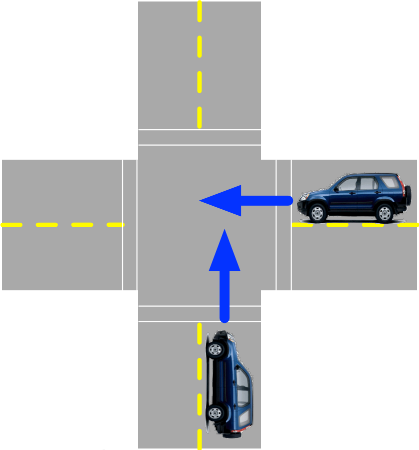
Norms and Social Laws

- A norm, or convention, is a rule for social behaviour, that is generally accepted through some tacit consensus in a multi-agent society, to improve the efficiency of that society.
- Some norms are so important that they become enshrined as social laws.
- Some examples of norms:
  - thou shalt not kill;
  - give your seat to an elderly person;
  - drive on the left/right!

Social Laws for Multi-Agent Systems
(Also known as Normative Systems)

- Mechanism design for *legacy systems*.
- Seminal works by Shoham and Tennenholz (1992, 1996)
- Social laws are *coordination mechanisms* for *pre-existing* systems.
  A set of rules for individual behaviour of the agents in the system with goal of ensuring that some desirable global behaviour, the objective, will result.
- *Prohibit* certain actions in certain states.

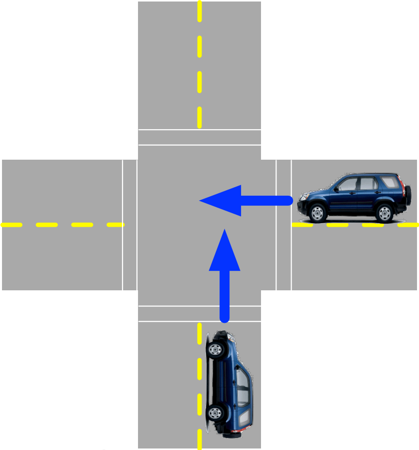
Example



- Objective: no crash
- Social law: right of way if coming from the right

⇒ objective achieved

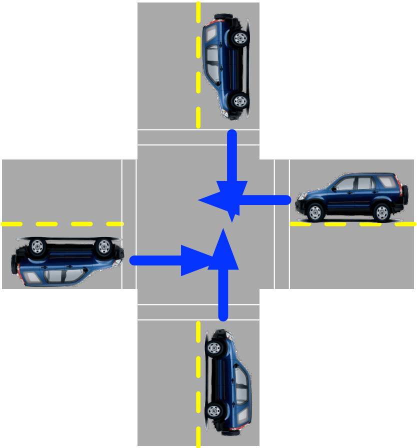
Example



- Objective: no crash
- Social law: right of way if coming from the right

⇒ objective achieved

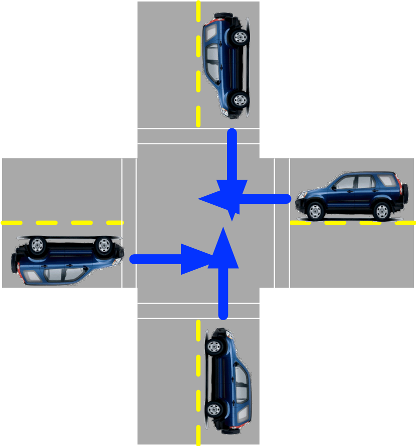
Example



- Objective: no crash
- Social law: right of way if coming from the right

$\Rightarrow$ objective achieved

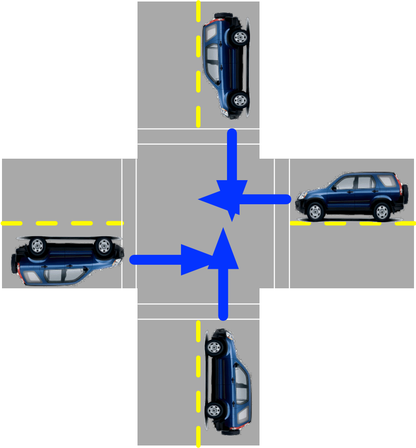
Example



- Objective: traffic flow and no crash
- Social law: right of way if coming from the right

⇒ objective not achieved

Example



- Objective: traffic flow and no crash
- Social law: right of way if coming from the right

⇒ objective not achieved

Offline and Online Design

Two ways social laws can come to exist:

1. *Offline design*
   Mechanisms are engineered at *design time*.
2. *Emergence at run-time*.
   Agents develop the social laws at run-time; typically by co-learning, copying, ...

Offline Design: Advantages & Disadvantages

+ system designer has absolute control;
+ optimality guarantees;
– not flexible $\Rightarrow$ not robust;
– constant reprogramming;
– complexity of design (we will discuss later!).

Emergence at Run-time: Advantages & Disadvantages

+ can adapt to changing/unforeseen circumstances;
– nobody has oversight;
– no optimality guarantees ("local maxima").

Offline design

- In the remainder of this course I will focus on offline design
- Offline design of social laws first investigated by Moses, Shoham and Tennenholtz (1991–97)
- In the remainder of this tutorial, we focus on the use of *logic* (in particular CTL in the *specification* and *synthesis* of social laws.
- Logic gives us a *transparent*, *precise*, and *unambiguous* language with which to express the properties of social laws.

Setting

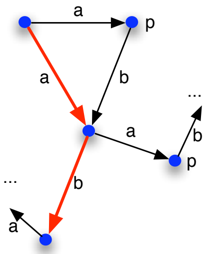
- Model the system as a Kripke model K
- Model the objective as a CTL (or ATL) formula φ
- It is typically the case that

$$K \not\models \varphi$$

Social Laws

A social law is simply a labelling of some of the transitions as undesirable or illegal

- It is typically the case that if none of the illegal transitions are used, the system will behave in a desirable way
- Fundamental assumption: agents choose whether or not to comply



Social Laws: formally

- Formally, a social law

$$\eta \subseteq R$$

is defined in the context of a Kripke structure, and is simply a subset of the transition relation R, such that $R \setminus \eta$ is a total relation.

- Intended interpretation: $(s, s') \in \eta$ means transition (s, s') is forbidden in η.
- Let $\mathcal{C}_\eta(s)$ be the set of η-conformant s-paths (w.r.t. some R).
- We can take union, intersection, etc, of normative systems: a *calculus* of normative systems.

Social Laws: formally

- Formally, a social law

$$\eta \subseteq R$$

  is defined in the context of a Kripke structure, and is simply a subset of the transition relation R, such that $R \setminus \eta$ is a total relation.
- Intended interpretation: $(s, s') \in \eta$ means transition (s, s') is forbidden in η.
- Let $\mathcal{C}_\eta(s)$ be the set of η-conformant s-paths (w.r.t. some R).
- We can take union, intersection, etc, of normative systems: a *calculus* of normative systems.

Social Laws: formally

- Formally, a social law
$$\eta \subseteq R$$
is defined in the context of a Kripke structure, and is simply a subset of the transition relation R, such that $R \setminus \eta$ is a total relation.
- Intended interpretation: $(s, s') \in \eta$ means transition (s, s') is forbidden in η.
- Let $\mathcal{C}_\eta(s)$ be the set of η-conformant s-paths (w.r.t. some R).
- We can take union, intersection, etc, of normative systems: a *calculus* of normative systems.

Implementing social laws

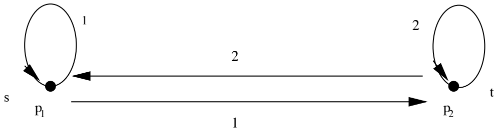
- Implementing a social law on a Kripke structure means eliminating from it all transitions that are forbidden
- If K is a Kripke structure, and η is a social law over K, then

$$K \dagger \eta$$

denotes the Kripke structure obtained from K by deleting transitions in η.

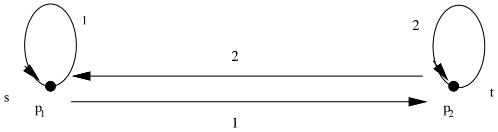
Example

K:

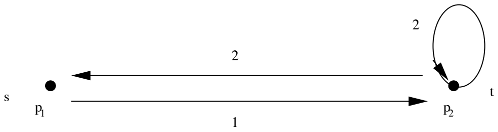


Example

K:

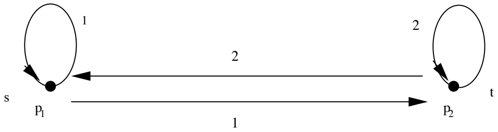


Let $\eta_1 = \{(s, s)\}$. $K \dagger \eta_1$:

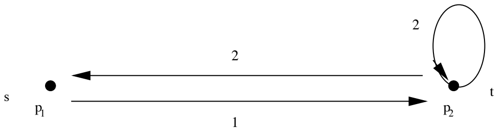


Example

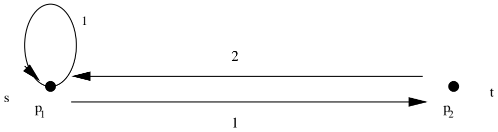
K:



Let $\eta_1 = \{(s, s)\}$. $K \dagger \eta_1$:



Let $\eta_2 = \{(t, t)\}$. $K \dagger \eta_2$:



Effective social laws

- A social law *eta* is *effective* if

$$K \dagger \eta \models \varphi$$

- In this case, implementing the norm will ensure the objective φ holds *under the assumption that everybody comply*.

Effective Social Laws

A social law η is *effective* in K wrt. objective φ if after implementing it, the objective φ is guaranteed to hold:

$$K \dagger \eta \models \varphi$$

EFFECTIVENESS:
Given: K, φ, η.
Question: is η effective?

Effective Social Laws

A social law η is *effective* in K wrt. objective φ if after implementing it, the objective φ is guaranteed to hold:

$$K \dagger \eta \models \varphi$$

EFFECTIVENESS:
Given: K, φ, η.
Question: is η effective?

Checking Effectiveness

Theorem

Effectiveness can be checked in polynomial time (if the model is explicitly represented).

The Feasibility Problem

FEASIBILITY:
Given: K, φ.
Question: *does there exist a social law such that is η effective wrt. objective φ?*

Theorem

The feasibility problem is NP-complete (for explicit representations).

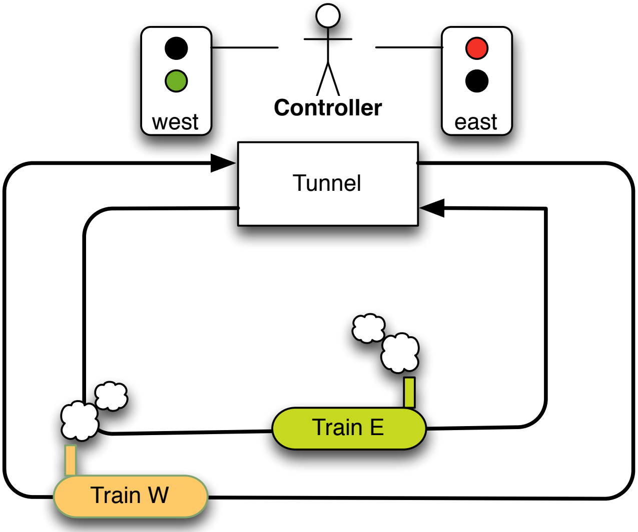
The Feasibility Problem

FEASIBILITY:
Given: K, φ.
Question: *does there exist a social law such that is η effective wrt. objective φ?*

Theorem

The feasibility problem is NP-complete (for explicit representations).

Example

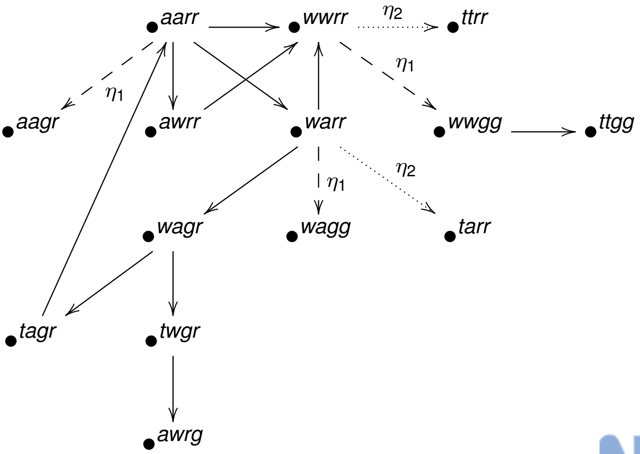


Norms:

n1: controller only sets east to green if E is waiting and W is not waiting, similar for west

n2: only enter on green light

Example: model and social laws



Example

$$crash = (EStatus = tunnel) \wedge (WStatus = tunnel)$$

$$K \dagger \eta_1 \not\models A \square crash$$

$$K \dagger \eta_2 \not\models A \square crash$$

$$K \dagger (\eta_1 \cup \eta_2) \models A \square crash$$

Example

$$crash = (EStatus = tunnel) \wedge (WStatus = tunnel)$$

$$K \dagger \eta_1 \not\models \mathsf{A}\Box crash$$

$$K \dagger \eta_2 \not\models \mathsf{A}\Box crash$$

$$K \dagger (\eta_1 \cup \eta_2) \models \mathsf{A}\Box crash$$

Example

$$crash = (EStatus = tunnel) \wedge (WStatus = tunnel)$$

$$K \dagger \eta_1 \not\models \mathsf{A}\Box crash$$

$$K \dagger \eta_2 \not\models \mathsf{A}\Box crash$$

$$K \dagger (\eta_1 \cup \eta_2) \models \mathsf{A}\Box crash$$

Example

$$crash = (EStatus = tunnel) \wedge (WStatus = tunnel)$$

$$K \dagger \eta_1 \not\models \mathsf{A}\Box\, crash$$

$$K \dagger \eta_2 \not\models \mathsf{A}\Box\, crash$$

$$K \dagger (\eta_1 \cup \eta_2) \models \mathsf{A}\Box\, crash$$

Contents

Symbolic Model Representation: SRML

- In practice: cannot represent state models explicitly
- Instead: need a succinct symbolic representation language
- SIMPLE REACTIVE MODULES LANGUAGE (SRML) [Hoek et al., 2006]: a rule-based language for MAS specifications

module *toggle* controls x
init
$\ell_1 : \top \rightsquigarrow x' := \top$
$\ell_2 : \top \rightsquigarrow x' := \bot$
update
$\ell_3 : x \rightsquigarrow x' := \bot$
$\ell_4 : (\neg x) \rightsquigarrow x' := \top$

Here ℓ are *labels* (think line numbers in BASIC!)

- Each agent is represented as a module, and a set of modules represent a Kripke structure

Symbolic Model Representation: SRML

- In practice: cannot represent state models explicitly
- Instead: need a succinct symbolic representation language
- SIMPLE REACTIVE MODULES LANGUAGE (SRML) [Hoek et al., 2006]: a rule-based language for MAS specifications

module *toggle* controls x

init

$\ell_1 : \top \rightsquigarrow x' := \top$

$\ell_2 : \top \rightsquigarrow x' := \bot$

update

$\ell_3 : x \rightsquigarrow x' := \bot$

$\ell_4 : (\neg x) \rightsquigarrow x' := \top$

Here ℓ are *labels* (think line numbers in BASIC!)

- Each agent is represented as a module, and a set of modules represent a Kripke structure

Symbolic Model Representation: SRML

- In practice: cannot represent state models explicitly
- Instead: need a succinct symbolic representation language
- SIMPLE REACTIVE MODULES LANGUAGE (SRML) [Hoek et al., 2006]: a rule-based language for MAS specifications

$$\begin{array}{l} \texttt{module}\ \mathit{toggle}\ \texttt{controls}\ x \\ \quad \texttt{init} \\ \quad \ell_1 : \top \rightsquigarrow x' := \top \\ \quad \ell_2 : \top \rightsquigarrow x' := \bot \\ \quad \texttt{update} \\ \quad \ell_3 : x \rightsquigarrow x' := \bot \\ \quad \ell_4 : (\neg x) \rightsquigarrow x' := \top \end{array}$$

Here ℓ are *labels* (think line numbers in BASIC!)

- Each agent is represented as a module, and a set of modules represent a Kripke structure

Symbolic Representation for Social Laws

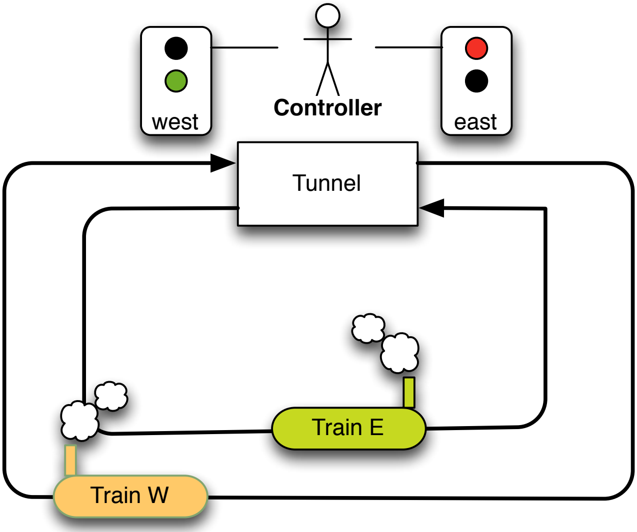
- (S)RML is a standard, general language for model representation
- What about social laws, i.e., model restrictions on such representations?
- We introduce Symbolic Normative Systems Language (SNL), which extends (S)RML with such restrictions
- A big advantage: allows us to write down a description of the model and of one or several social laws *separately* and in a *modular* way
  - can *modify* the social law without modifying the model
  - compare different social laws in the context of the same model

Symbolic Normative Systems Language: SNL

$$
\begin{array}{l}
\texttt{normative-system}\ \textit{id} \\
\quad \chi_1\ \texttt{disables}\ \ell_{1_1}, \ldots, \ell_{1_k} \\
\quad\quad \ldots \\
\quad \chi_m\ \texttt{disables}\ \ell_{m_1}, \ldots, \ell_{m_k}
\end{array}
$$

- An SNL interpretation is a collection of SNL normative systems

Example



Norms:

n1: controller only sets east to green if E is waiting and W is not waiting, similar for west

n2: only enter on green light

Example: SNL representation (1/2)

$$
\begin{array}{l}
\texttt{module}\ \mathit{controller}\ \texttt{controls}\ \mathit{EGreen}, \mathit{WGreen}\\
\quad \texttt{init}\\
\quad \ell_1 : \top \leadsto \mathit{EGreen}' = \top\\
\quad \ell_2 : \top \leadsto \mathit{WGreen}' = \bot\\
\quad \texttt{update}\\
\quad \mathit{SwitchE} : \top \leadsto \mathit{EGreen}' := \neg \mathit{EGreen}\\
\quad \mathit{SwitchW} : \top \leadsto \mathit{WGreen}' := \neg \mathit{WGreen}
\end{array}
$$

$$
\begin{array}{l}
\texttt{normative-system}\ \eta_c\\
\quad \neg \mathit{EGreen} \wedge \neg(\mathit{EStatus} = \mathit{waiting} \wedge \neg \mathit{WStatus} = \mathit{waiting})\\
\qquad \texttt{disables}\ \mathit{SwitchE}\\
\quad \neg \mathit{WGreen} \wedge \neg(\mathit{WStatus} = \mathit{waiting} \wedge \neg \mathit{EStatus} = \mathit{waiting})\\
\qquad \texttt{disables}\ \mathit{SwitchW}
\end{array}
$$

Example: SNL representation (2/2)

```
module TrainE controls EStatus
  init
  ℓ3 : ⊤ ⇝ EStatus' := waiting
  update
  ℓ4 : EStatus = away ⇝ EStatus' := away
  ℓ5 : EStatus = away ⇝ EStatus' := waiting
  ℓ6 : EStatus = waiting ⇝ EStatus' := waiting
  Eenter : EStatus = waiting ⇝ EStatus' := tunnel
  ℓ7 : EStatus = tunnel ⇝ EStatus' := away
```

```
normative-system η2
  ¬EGreen disables Eenter
  ¬WGreen disables Wenter
```

(For brevity we take a little liberty with the notation; variables should really be Booleans)

Relations Between Symbolic Normative Systems

- Suppose we ask whether η is a *subset* of η', i.e., $\eta \sqsubseteq \eta'$.
- For *explicit* representations, checking this is easy (set containment).

Theorem

This problem is PSPACE-complete for SNL representations

Theorem

Checking equivalence of SNL normative systems is also PSPACE-complete

Effectiveness under Symbolic Representations

<u>EFFECTIVENESS</u>:
Given: K, φ, η.
Question: is η effective?

Theorem

Effectiveness can be checked in polynomial time (if the model is explicitly represented). If the model is represented using the reactive modules language, checking effectiveness is PSPACE-complete.

Feasibility under Symbolic Representations

FEASIBILITY:

Given: K, φ.

Question: *does there exist a social law such that is* η *effective wrt. objective* φ*?*

Theorem

The feasibility problem is NP-complete for (explicit representations), and PSPACE-complete for reactive modules.

Feasibility under Symbolic Representations

FEASIBILITY:
Given: K, φ.
Question: *does there exist a social law such that is η effective wrt. objective φ?*

Theorem

The feasibility problem is NP-complete for (explicit representations), and PSPACE-complete for reactive modules.

Contents

Representing Social Laws

- We have used logic as a specification for the desirable properties of social laws.
- But we haven't (yet) seen a logic *about* social laws, i.e., where we can talk about social laws *in the object language*.
- Since we are in the realm of talking about what is *prohibited* and *permissible*, we are here close to the real of *deontic logic*: the logic of permissions and obligations.

Normative Temporal Logic (NTL)

- Based on CTL, with quantifiers *for each normative system*.
- Deontic modalities are *contextualised* to normative systems
  Can only talk about whether something is permissible/obligated *in the context of a specific normative systems*
  Makes it possible to talk about inconsistencies *between* normative systems in the object language
- Have deontic modalities that are *bound* to temporal operators.

NTL: syntax

- Basic operators, where η is a name for a normative system:

$\mathsf{P}_\eta\varphi$	φ is permissible in η
$\mathsf{O}_\eta\varphi$	φ is obligatory in η

- Combined with CTL tense operators:

$\bigcirc$	next
$\Diamond$	eventually
$\Box$	always
$\mathcal{U}$	until

- and the usual propositional connectives
- Example: $\mathsf{P}_{Tokyo} \Box \, eatnoodles$
- Example: $\mathsf{O}_{Tokyo} \Diamond paynoodles$

NTL: syntax

- Basic operators, where η is a name for a normative system:

$\mathsf{P}_\eta \varphi$	φ is permissible in η
$\mathsf{O}_\eta \varphi$	φ is obligatory in η

- Combined with CTL tense operators:

$\bigcirc$	next
$\Diamond$	eventually
$\Box$	always
$\mathcal{U}$	until

- and the usual propositional connectives
- Example: $\mathsf{P}_{Tokyo} \Box \, eatnoodles$
- Example: $\mathsf{O}_{Tokyo} \Diamond paynoodles$

NTL: syntax

- Basic operators, where η is a name for a normative system:

$\mathsf{P}_\eta\varphi$	φ is permissible in η
$\mathsf{O}_\eta\varphi$	φ is obligatory in η

- Combined with CTL tense operators:

$\bigcirc$	next
$\diamondsuit$	eventually
$\square$	always
$\mathcal{U}$	until

- and the usual propositional connectives
- Example: $\mathsf{P}_{\mathit{Tokyo}} \square \mathit{eatnoodles}$
- Example: $\mathsf{O}_{\mathit{Tokyo}} \diamondsuit \mathit{paynoodles}$

NTL: syntax

- Basic operators, where η is a name for a normative system:

$\mathsf{P}_\eta\varphi$	φ is permissible in η
$\mathsf{O}_\eta\varphi$	φ is obligatory in η

- Combined with CTL tense operators:

$\bigcirc$	next
$\Diamond$	eventually
$\Box$	always
$\mathcal{U}$	until

- and the usual propositional connectives
- Example: $\mathsf{P}_{\mathit{Tokyo}} \Box \mathit{eatnoodles}$
- Example: $\mathsf{O}_{\mathit{Tokyo}} \Diamond \mathit{paynoodles}$

NTL: syntax

- Basic operators, where η is a name for a normative system:

$\mathsf{P}_\eta\varphi$	φ is permissible in η
$\mathsf{O}_\eta\varphi$	φ is obligatory in η

- Combined with CTL tense operators:

$\bigcirc$	next
$\Diamond$	eventually
$\Box$	always
$\mathcal{U}$	until

- and the usual propositional connectives
- Example: $\mathsf{P}_{\mathit{Tokyo}}\,\Box\,\mathit{eatnoodles}$
- Example: $\mathsf{O}_{\mathit{Tokyo}}\Diamond\mathit{paynoodles}$

NTL: Semantics

- For semantics, we need an *interpretation* I for normative systems named in formulae.
- Require: $I(\eta_\emptyset) = \emptyset$
- Interpreting obligations...
  φ is *obligatory* in η if φ is true on *all* η-conformant computations
  $K, s \models_I O_\eta \bigcirc \varphi$ iff $\forall \pi \in C_{I(\eta)}(s) : K, \pi[1] \models_I \varphi$;
- Interpreting permissions...
  φ is *permissible* in η if φ is true on *some* η-conformant computation
  $K, s \models_I P_\eta \bigcirc \varphi$ iff $\exists \pi \in C_{I(\eta)}(s) : K, \pi[1] \models_I \varphi$;

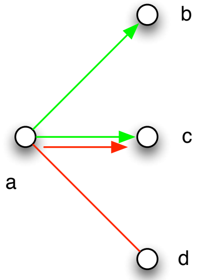
NTL: Semantics

- For semantics, we need an *interpretation* I for normative systems named in formulae.
- Require: $I(\eta_\emptyset) = \emptyset$
- Interpreting obligations...
  φ is *obligatory* in η if φ is true on *all* η-conformant computations
  $K, s \models_I \mathsf{O}_\eta \bigcirc \varphi$ iff $\forall \pi \in C_{I(\eta)}(s) : K, \pi[1] \models_I \varphi$;
- Interpreting permissions...
  φ is *permissible* in η if φ is true on *some* η-conformant computation
  $K, s \models_I \mathsf{P}_\eta \bigcirc \varphi$ iff $\exists \pi \in C_{I(\eta)}(s) : K, \pi[1] \models_I \varphi$;

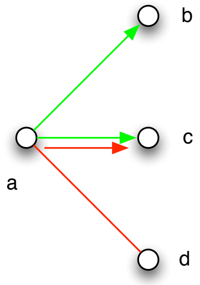
NTL: Semantics

- For semantics, we need an *interpretation* I for normative systems named in formulae.
- Require: $I(\eta_\emptyset) = \emptyset$
- Interpreting obligations...
  φ is *obligatory* in η if φ is true on *all* η-conformant computations
  $K, s \models_I O_\eta \bigcirc \varphi \quad \text{iff} \quad \forall \pi \in C_{I(\eta)}(s) : K, \pi[1] \models_I \varphi;$
- Interpreting permissions...
  φ is *permissible* in η if φ is true on *some* η-conformant computation
  $K, s \models_I P_\eta \bigcirc \varphi \quad \text{iff} \quad \exists \pi \in C_{I(\eta)}(s) : K, \pi[1] \models_I \varphi;$

Example

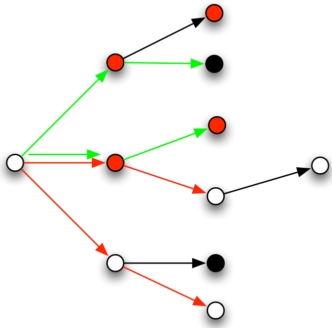


Example

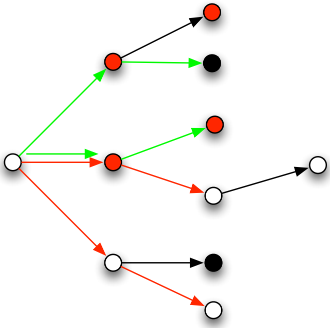


- $I(\eta) = \{(a, b), (a, c)\}$
- $I(\eta) = \{(a, c), (a, d)\}$

Example

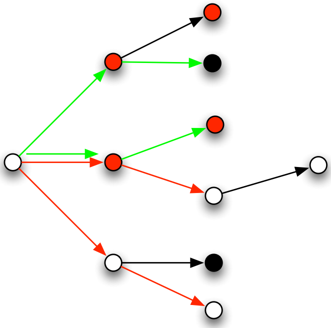


Example



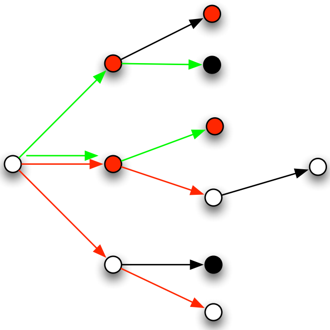
- $O_{\color{red}\eta} \bigcirc$ *red*

Example



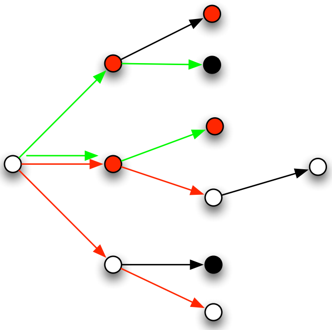
- $O_{\eta} \bigcirc \mathit{red} \wedge \neg O_{\eta} \bigcirc \mathit{red}$

Example



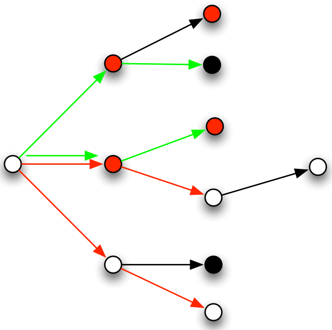
- $O_{\eta} \bigcirc \mathit{red} \wedge \neg O_{\eta} \bigcirc \mathit{red} \wedge P_{\eta} \bigcirc \mathit{white}$

Example



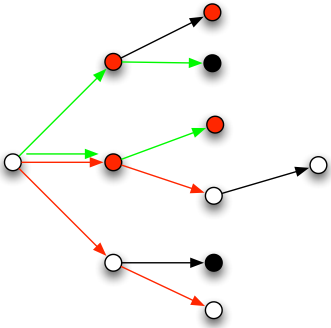
- $O_{\eta} \bigcirc \mathit{red} \wedge \neg O_{\eta} \bigcirc \mathit{red} \wedge P_{\eta} \bigcirc \mathit{white} \wedge \neg P_{\eta} \bigcirc \mathit{white}$

Example



- $O_{\eta} \Box (red \vee black)$

Example



- $P_{\eta} \bigcirc O_{\eta} \bigcirc$ *red*

Some properties

We write:

$$A\varphi \equiv \mathsf{O}_{\eta_\emptyset}\varphi \quad E\varphi \equiv \mathsf{P}_{\eta_\emptyset}\varphi$$

For any normative system η:

$$\models (A\varphi \to \mathsf{O}_\eta\varphi) \quad \models (\mathsf{O}_\eta\varphi \to \mathsf{P}_\eta\varphi) \quad \models (\mathsf{P}_\eta\varphi \to E\varphi)$$

Some properties

We write:

$$A\varphi \equiv \mathsf{O}_{\eta_\emptyset}\varphi \quad E\varphi \equiv \mathsf{P}_{\eta_\emptyset}\varphi$$

For any normative system η:

$$\models (A\varphi \rightarrow \mathsf{O}_\eta\varphi) \quad \models (\mathsf{O}_\eta\varphi \rightarrow \mathsf{P}_\eta\varphi) \quad \models (\mathsf{P}_\eta\varphi \rightarrow E\varphi)$$

NTL: Axioms

(Ax1) All validities of propositional logic

(Ax2) $P_\eta \Diamond \varphi \leftrightarrow P_\eta(\top \mathcal{U} \varphi)$

(Ax2b) $O_\eta \Box \varphi \leftrightarrow \neg P_\eta \Diamond \neg \varphi$

(Ax3) $O_\eta \Diamond \varphi \leftrightarrow O_\eta(\top \mathcal{U} \varphi)$

(Ax3b) $P_\eta \Box \varphi \leftrightarrow \neg O_\eta \Diamond \neg \varphi$

(Ax4) $P_\eta \bigcirc (\varphi \vee \psi) \leftrightarrow (P_\eta \bigcirc \varphi \vee P_\eta \bigcirc \psi)$

(Ax5) $O_\eta \bigcirc \varphi \leftrightarrow \neg P_\eta \bigcirc \neg \varphi$

(Ax6) $P_\eta(\varphi \mathcal{U} \psi) \leftrightarrow (\psi \vee (\varphi \wedge P_\eta \bigcirc P_\eta(\varphi \mathcal{U} \psi)))$

(Ax7) $O_\eta(\varphi \mathcal{U} \psi) \leftrightarrow (\psi \vee (\varphi \wedge O_\eta \bigcirc O_\eta(\varphi \mathcal{U} \psi)))$

(Ax8) $P_\eta \bigcirc \top \wedge O_\eta \bigcirc \top$

NTL: Axioms

(Ax9) $O_\eta(\varphi \to (\neg\psi \wedge P_\eta \bigcirc \varphi)) \to (\varphi \to \neg O_\eta(\gamma \mathcal{U} \psi))$

(Ax9b) $O_\eta \Box(\varphi \to (\neg\psi \wedge P_\eta \bigcirc \varphi)) \to (\varphi \to \neg O_\eta \Diamond \psi)$

(Ax10) $O_\eta \Box(\varphi \to (\neg\psi \wedge (\gamma \to O_\eta \bigcirc \varphi))) \to (\varphi \to \neg P_\eta(\gamma \mathcal{U} \psi))$

(Ax10b) $O_\eta \Box(\varphi \to (\neg\psi \wedge O_\eta \bigcirc \varphi)) \to (\varphi \to \neg P_\eta \Diamond \psi)$

(Ax11) $O_\eta \Box(\varphi \to \psi) \to (P_\eta \bigcirc \varphi \to P_\eta \bigcirc \psi)$

(R1) If $\vdash \varphi$ then $\vdash O_\eta \Box \varphi$ (generalization)

(R2) If $\vdash \varphi$ and $\vdash \varphi \to \psi$ then $\vdash \psi$ (modus ponens)

NTL: Axioms

(Obl) If something is obligatory in "nature", then it must be obligatory in any normative system you invent.

$$O_{\eta_\emptyset}\alpha \rightarrow O_\eta \alpha$$

(Perm) You cannot make things possible in a normative system that were not possible in nature.

$$P_\eta \alpha \rightarrow P_{\eta_\emptyset}\alpha$$

NTL: Axioms

(Obl) If something is obligatory in "nature", then it must be obligatory in any normative system you invent.

$$O_{\eta_\emptyset} \alpha \to O_\eta \alpha$$

(Perm) You cannot make things possible in a normative system that were not possible in nature.

$$P_\eta \alpha \to P_{\eta_\emptyset} \alpha$$

NTL: Axioms: adding normative systems dependencies

- Let us write $\eta \sqsubseteq \eta'$ if $I(\eta) \subseteq I(\eta')$
- Then $\eta \sqsubseteq \eta'$ means η is *less restrictive* than η'.
- This gives following two axioms...
- If η is *less restrictive* than η' then anything obligatory in η is obligatory in η'

$$\eta \sqsubseteq \eta' \rightarrow (\mathsf{O}_{\eta}\alpha \rightarrow \mathsf{O}_{\eta'}\alpha)$$

- If η is *less restrictive* than η' then anything permissible in η' is permissible in η

$$\eta \sqsubseteq \eta' \rightarrow (\mathsf{P}_{\eta'}\alpha \rightarrow \mathsf{P}_{\eta}\alpha)$$

NTL: Axioms: adding normative systems dependencies

- Let us write $\eta \sqsubseteq \eta'$ if $I(\eta) \subseteq I(\eta')$
- Then $\eta \sqsubseteq \eta'$ means η is *less restrictive* than η'.
- This gives following two axioms...
- If η is *less restrictive* than η' then anything obligatory in η is obligatory in η'

$$\eta \sqsubseteq \eta' \rightarrow (\mathsf{O}_\eta \alpha \rightarrow \mathsf{O}_{\eta'} \alpha)$$

- If η is *less restrictive* than η' then anything permissible in η' is permissible in η

$$\eta \sqsubseteq \eta' \rightarrow (\mathsf{P}_{\eta'} \alpha \rightarrow \mathsf{P}_\eta \alpha)$$

NTL: Axioms: adding normative systems dependencies

- Let us write $\eta \sqsubseteq \eta'$ if $I(\eta) \subseteq I(\eta')$
- Then $\eta \sqsubseteq \eta'$ means η is *less restrictive* than η'.
- This gives following two axioms...
- If η is *less restrictive* than η' then anything obligatory in η is obligatory in η'

$$\eta \sqsubseteq \eta' \to (\mathsf{O}_\eta \alpha \to \mathsf{O}_{\eta'} \alpha)$$

- If η is *less restrictive* than η' then anything permissible in η' is permissible in η

$$\eta \sqsubseteq \eta' \to (\mathsf{P}_{\eta'} \alpha \to \mathsf{P}_\eta \alpha)$$

NTL: Axioms: adding normative systems dependencies

- Let us write $\eta \sqsubseteq \eta'$ if $I(\eta) \subseteq I(\eta')$
- Then $\eta \sqsubseteq \eta'$ means η is *less restrictive* than η'.
- This gives following two axioms...
- If η is *less restrictive* than η' then anything obligatory in η is obligatory in η'

$$\eta \sqsubseteq \eta' \rightarrow (\mathsf{O}_{\eta}\alpha \rightarrow \mathsf{O}_{\eta'}\alpha)$$

- If η is *less restrictive* than η' then anything permissible in η' is permissible in η

$$\eta \sqsubseteq \eta' \rightarrow (\mathsf{P}_{\eta'}\alpha \rightarrow \mathsf{P}_{\eta}\alpha)$$

NTL: Axioms

Theorem

The axiomatic system is a sound and complete axiomatisation of NTL.

Relationship to Deontic Logic

Two main differences to the language of (conventional) deontic logic:

- The NTL operators are contextual; they refer to specific normative system. One formula can refer to several different normative systems.
- All deontic operators in NTL are bound to temporal operators – all deontic epressions refer to time. Conventional deontic logic contains no notion of time

Relationship to Deontic Logic

Two main differences to the language of (conventional) deontic logic:

- The NTL operators are contextual; they refer to specific normative system. One formula can refer to several different normative systems.
- All deontic operators in NTL are bound to temporal operators – all deontic epressions refer to time. Conventional deontic logic contains no notion of time

Relationship to Deontic Logic

Two main differences to the language of (conventional) deontic logic:

- The NTL operators are contextual; they refer to specific normative system. One formula can refer to several different normative systems.
- All deontic operators in NTL are bound to temporal operators – all deontic epressions refer to time. Conventional deontic logic contains no notion of time

Relationship to Deontic Logic

- How to compare?
  - Possibility 1: interpret "obligatory" ($O\varphi$) in conventional deontic logic to mean "always obligatory" ($O_\eta \Box \varphi$)
  - Possibility 2: interpret "obligatory" ($O\varphi$) in conventional deontic logic to mean "obligatory at the next point in time" ($O_\eta \bigcirc \varphi$)

In either case: all the principles of Standard Deontic Logic (STD) hold in NTL

- $O(\varphi \to \psi) \to (O\varphi \to O\psi)$ (K)
- $\neg O\bot$ (D)
- from φ infer $O\varphi$ (N)

Relationship to Deontic Logic

- How to compare?
  - Possibility 1: interpret "obligatory" ($O\varphi$) in conventional deontic logic to mean "always obligatory" ($O_\eta \square \varphi$)
  - Possibility 2: interpret "obligatory" ($O\varphi$) in conventional deontic logic to mean "obligatory at the next point in time" ($O_\eta \bigcirc \varphi$)

In either case: all the principles of Standard Deontic Logic (STD) hold in NTL

- $O(\varphi \rightarrow \psi) \rightarrow (O\varphi \rightarrow O\psi)$ (K)
- $\neg O\bot$ (D)
- from φ infer $O\varphi$ (N)

Relationship to Deontic Logic

- How to compare?
  - Possibility 1: interpret "obligatory" ($O\varphi$) in conventional deontic logic to mean "always obligatory" ($O_\eta \Box \varphi$)
  - Possibility 2: interpret "obligatory" ($O\varphi$) in conventional deontic logic to mean "obligatory at the next point in time" ($O_\eta \bigcirc \varphi$)

In either case: all the principles of Standard Deontic Logic (STD) hold in NTL

- $O(\varphi \rightarrow \psi) \rightarrow (O\varphi \rightarrow O\psi)$ (K)
- $\neg O\bot$ (D)
- from φ infer $O\varphi$ (N)

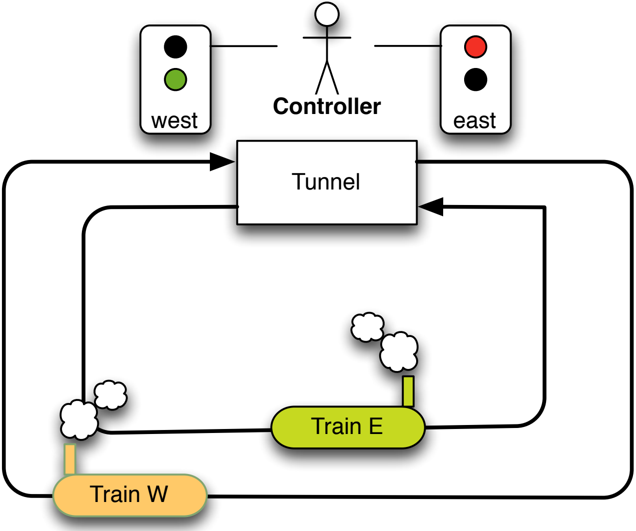
Relationship to Deontic Logic

- How to compare?
  - Possibility 1: interpret "obligatory" ($\mathsf{O}\varphi$) in conventional deontic logic to mean "always obligatory" ($\mathsf{O}_\eta \Box \varphi$)
  - Possibility 2: interpret "obligatory" ($\mathsf{O}\varphi$) in conventional deontic logic to mean "obligatory at the next point in time" ($\mathsf{O}_\eta \bigcirc \varphi$)

In either case: all the principles of Standard Deontic Logic (STD) hold in NTL

- $\mathsf{O}(\varphi \rightarrow \psi) \rightarrow (\mathsf{O}\varphi \rightarrow \mathsf{O}\psi)$ (K)
- $\neg\mathsf{O}\bot$ (D)
- from φ infer $\mathsf{O}\varphi$ (N)

Example



Norms:

n1: controller only sets east to green if E is waiting and W is not waiting, similar for west

n2: only enter on green light

Example: SNL representation (1/2)

```
module controller controls EGreen, WGreen
 init
 ℓ1 : ⊤ ⇝ EGreen' = ⊤
 ℓ2 : ⊤ ⇝ WGreen' = ⊥
 update
 SwitchE : ⊤ ⇝ EGreen' := ¬EGreen
 SwitchW : ⊤ ⇝ WGreen' := ¬WGreen
```

normative-system η_1

$\neg EGreen \wedge \neg(EStatus = waiting \wedge \neg WStatus = waiting)$

disables *SwitchE*

$\neg WGreen \wedge \neg(WStatus = waiting \wedge \neg EStatus = waiting)$

disables *SwitchW*

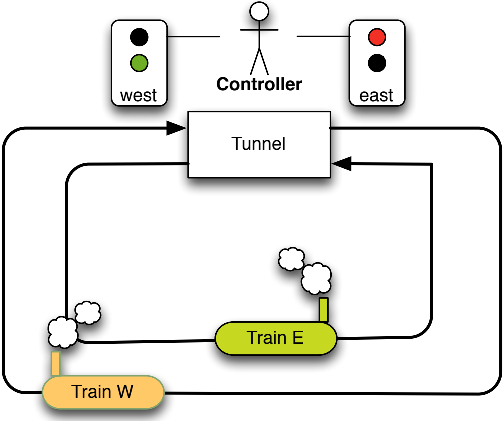
Example: SNL representation (2/2)

$$
\begin{array}{l}
\texttt{module}\ \mathit{TrainE}\ \texttt{controls}\ \mathit{EStatus} \\
\quad \texttt{init} \\
\quad \ell_3 : \top \leadsto \mathit{EStatus}' := \mathit{waiting} \\
\quad \texttt{update} \\
\quad \ell_4 : \mathit{EStatus} = \mathit{away} \leadsto \mathit{EStatus}' := \mathit{away} \\
\quad \ell_5 : \mathit{EStatus} = \mathit{away} \leadsto \mathit{EStatus}' := \mathit{waiting} \\
\quad \ell_6 : \mathit{EStatus} = \mathit{waiting} \leadsto \mathit{EStatus}' := \mathit{waiting} \\
\quad \mathit{Eenter} : \mathit{EStatus} = \mathit{waiting} \leadsto \mathit{EStatus}' := \mathit{tunnel} \\
\quad \ell_7 : \mathit{EStatus} = \mathit{tunnel} \leadsto \mathit{EStatus}' := \mathit{away}
\end{array}
$$

$$
\begin{array}{l}
\texttt{normative-system}\ \eta_2 \\
\quad \neg \mathit{EGreen}\ \texttt{disables}\ \mathit{Eenter} \quad \neg \mathit{WGreen}\ \texttt{disables}\ \mathit{Wenter}
\end{array}
$$

(For brevity we take a little liberty with the notation; variables should really be Booleans)

Example: NTL properties



Norms:

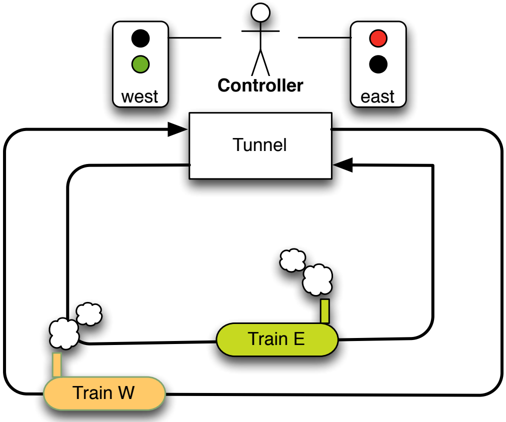
n1: controller only sets east to green if E is waiting and W is not waiting, similar for west

n2: only enter on green light

Let $crash = (EStatus = tunnel) \wedge (WStatus = tunnel)$

- $P_{\eta_\emptyset} \Diamond crash$
- $P_{\eta_2} \Diamond crash$
- $O_{\eta_1 \cup \eta_2} \Box \neg crash$

Example: NTL properties



Norms:

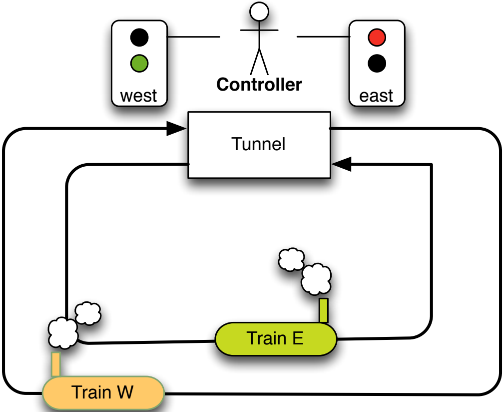
n1: controller only sets east to green if E is waiting and W is not waiting, similar for west

n2: only enter on green light

Let $crash = (EStatus = tunnel) \wedge (WStatus = tunnel)$

- $P_{\eta_\emptyset} \Diamond crash$
- $P_{\eta_2} \Diamond crash$
- $O_{\eta_1 \cup \eta_2} \Box \neg crash$

Example: NTL properties



Norms:

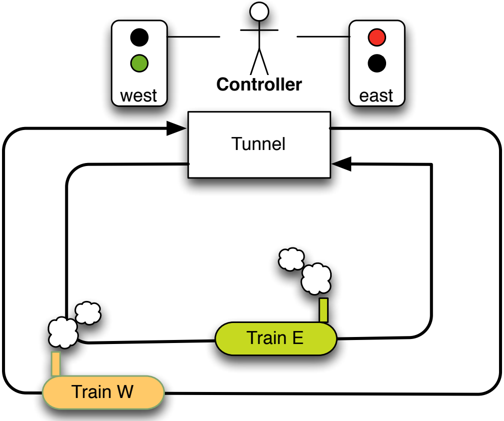
n1: controller only sets east to green if E is waiting and W is not waiting, similar for west

n2: only enter on green light

Let $crash = (EStatus = tunnel) \wedge (WStatus = tunnel)$

- $P_{\eta_\emptyset} \Diamond crash$
- $P_{\eta_2} \Diamond crash$
- $O_{\eta_1 \cup \eta_2} \Box \neg crash$

Example: NTL properties



Norms:

n1: controller only sets east to green if E is waiting and W is not waiting, similar for west

n2: only enter on green light

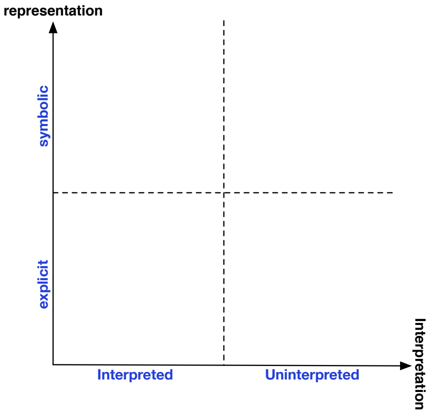
Let $crash = (EStatus = tunnel) \wedge (WStatus = tunnel)$

- $P_{\eta_\emptyset} \Diamond crash$
- $P_{\eta_2} \Diamond crash$
- $O_{\eta_1 \cup \eta_2} \Box \neg crash$

Model Checking

Variants:

- Model representation: explicit or symbolic?
- Interpretation of normative systems named in the formula: given or synthesised?



Interpreted Explicit State Model Checking

Definition

Given a Kripke structure $K = \langle S, S^0, R, V \rangle$, interpretation $I : \Sigma_\eta \rightarrow N(R)$ and formula φ of NTL, is it the case that $K \models_I \varphi$?

Theorem

P-complete

Interpreted Explicit State Model Checking

Definition

Given a Kripke structure $K = \langle S, S^0, R, V \rangle$, interpretation $I : \Sigma_\eta \rightarrow N(R)$ and formula φ of NTL, is it the case that $K \models_I \varphi$?

Theorem

P-complete

Uninterpreted Explicit State Model Checking

Definition

Given a Kripke structure $K = \langle S, S^0, R, V \rangle$ and formula φ of NTL, does there exist an interpretation I such that $K \models_I \varphi$?

Theorem

NP-complete

Uninterpreted Explicit State Model Checking

Definition

Given a Kripke structure $K = \langle S, S^0, R, V \rangle$ and formula φ of NTL, does there exist an interpretation I such that $K \models_I \varphi$?

Theorem

NP-complete

Interpreted SRML Model Checking

Definition

Given SRML system R, SNL normative systems $\eta_1, \ldots, \eta_k$, and NTL formula φ over these, is φ satisfied by these?

Theorem

PSPACE-complete

Interpreted SRML Model Checking

Definition

Given SRML system R, SNL normative systems $\eta_1, \ldots, \eta_k$, and NTL formula φ over these, is φ satisfied by these?

Theorem

PSPACE-complete

Uninterpreted SRML Model Checking

Definition

Given SRML system R and NTL formula φ over R, does there exist an interpretation I such that φ is satisfied under these?

Theorem

EXPTIME-hard

Uninterpreted SRML Model Checking

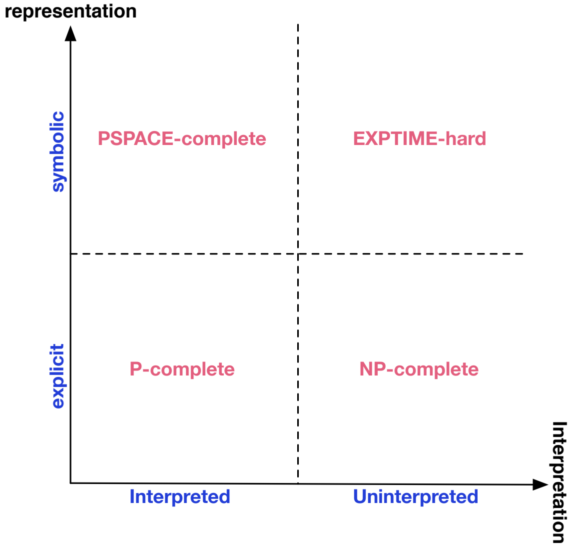
Definition

Given SRML system R and NTL formula φ over R, does there exist an interpretation I such that φ is satisfied under these?

Theorem

EXPTIME-hard

Complexity: summary



Some references I

Thomas Ågotnes, Wiebe van der Hoek, Carles Sierra Juan A. Rodriguez-Aguilar, and Michael Wooldridge.
A temporal logic of normative systems, volume 28 of *Trends in Logic*, pages 69–104.
2009.

Thomas Ågotnes, Wiebe van der Hoek, Carles Sierra Juan A. Rodriguez-Aguilar, and Michael Wooldridge.
On the logic of normative systems.
In M. M. Veloso, editor, *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI 2007)*, pages 1175–1180, California, 2007. AAAI Press.

Thomas Ågotnes, Wiebe van der Hoek, Juan A. Rodriguez-Aguilar, Carles Sierra, and Michael Wooldridge.
Multi-modal CTL: Completeness, complexity and an application.
Studia Logica, 92(1):1–26, 2009.

Thomas Ågotnes, Wiebe van der Hoek, Carles Sierra Juan A. Rodriguez-Aguilar, and Michael Wooldridge.
The simple normative systems language.
In Virginia Dignum, Frank Dignum, Bruce Edmonds, and Eric Matson, editors, *Agent Organizations: Models and Simulations, Proceedings of the IJCAI 07 Workshop (AOMS 2007)*, Hyderabad, India, January 2007.

HOEK, W. VAN DER, A. LOMUSCIO, and M. WOOLDRIDGE, 'On the complexity of practical ATL model checking', in *Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-2006)*, Hakodate, Japan, 2005, pp. 201–208.

SHOHAM, Y., and M. TENNENHOLTZ, 'Emergent conventions in multi-agent systems', in C. Rich, W. Swartout, and B. Nebel, (eds.), *Proceedings of Knowledge Representation and Reasoning (KR&R-92)*, 1992, pp. 225–231.

Some references II

- SHOHAM, Y., and M. TENNENHOLTZ, ‘On the synthesis of useful social laws for artificial agent societies’, in *Proceedings of the Tenth National Conference on Artificial Intelligence (AAAI-92)*, San Diego, CA, 1992, pp. 276–281.
- MOSES, Y., and M. TENNENHOLTZ, ‘Artificial social systems’, *Computers and AI*, 14 (1995), 6, 533–562.
- SHOHAM, Y., and M. TENNENHOLTZ, ‘On social laws for artificial agent societies: Off-line design’, in P. E. Agre, and S. J. Rosenschein, (eds.), *Computational Theories of Interaction and Agency*, The MIT Press: Cambridge, MA, 1996, pp. 597–618.
- SHOHAM, Y., and M. TENNENHOLTZ, ‘On the emergence of social conventions: Modelling, analysis, and simulations’, *Artificial Intelligence*, 94 (1997), 1-2, 139–166.
- HOEK, W. VAN DER, M. ROBERTS, and M. WOOLDRIDGE, ‘Social laws in alternating time: Effectiveness, feasibility, and synthesis’, *Synthese*, 156 (2007), 1, 1–19.