What is a Good Domain Description?

Evaluating & Revising Action Theories in Dynamic Logic

Ivan José Varzinczak

IRIT – Université Paul Sabatier

October 27th 2006

Reasoning About Actions

Problem: describing domains by logical formulas

- Actions and their effects
- Executabilities of actions
- Inexecutabilities of actions
- Domain constraints

Example

- A turkey that walks is alive
- Teasing a turkey makes it walk
- It is always possible to tease a turkey
- A dead turkey remains dead after teasing it

Reasoning About Actions

Problem: describing domains by logical formulas

- Actions and their effects
- Executabilities of actions
- Inexecutabilities of actions
- Domain constraints

Example

- A turkey that walks is alive
- Teasing a turkey makes it walk
- It is always possible to tease a turkey
- A dead turkey remains dead after teasing it

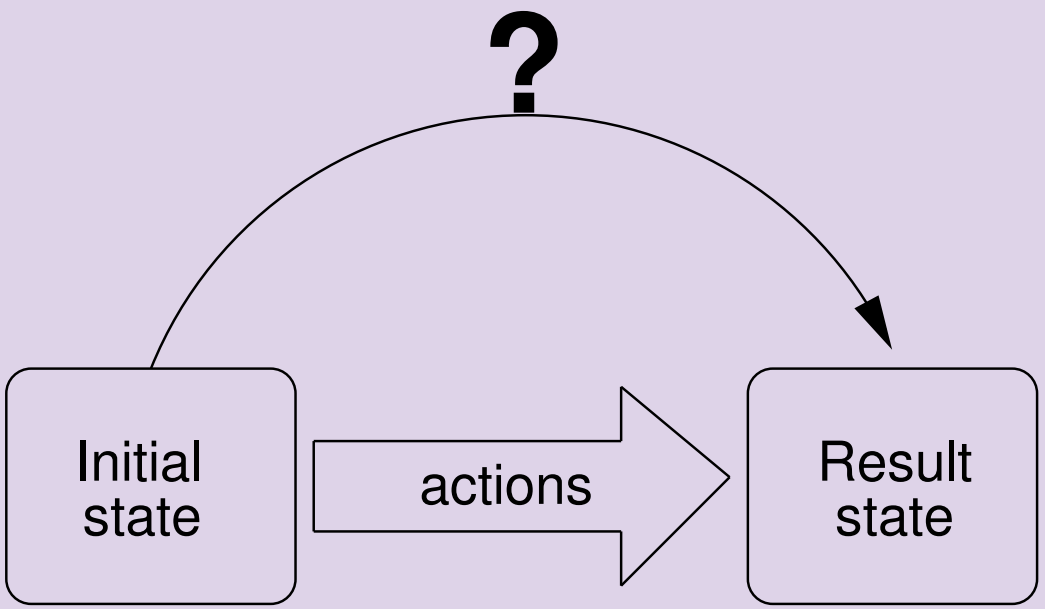
Reasoning About Actions

Goal: inference tasks

- Prediction
- Explanation
- Planning

Reasoning About Actions

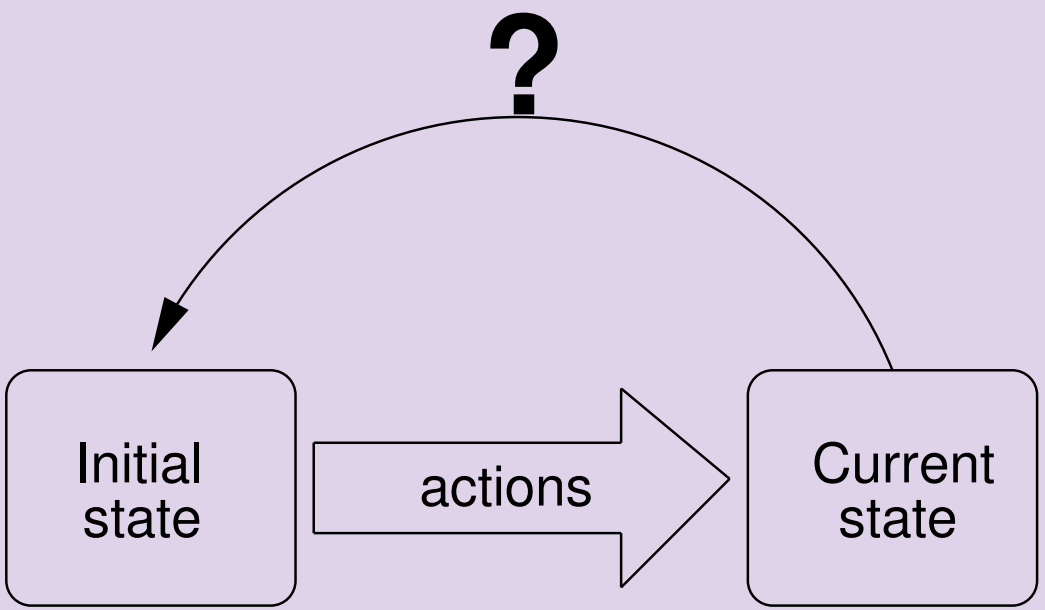
Prediction: reasoning about the future



- After shooting, the turkey stops walking

Reasoning About Actions

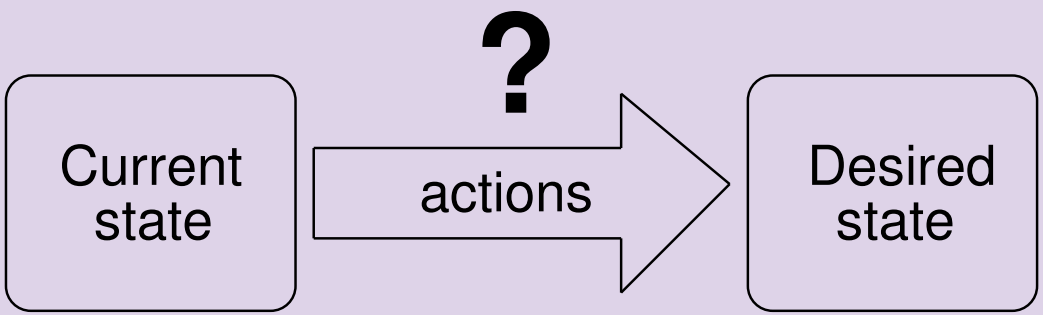
Explanation: reasoning about the past



- After shooting, the turkey is dead: the gun was loaded

Reasoning About Actions

Planning: what to do to achieve a goal



- To have the turkey dead: load the gun, then shoot

Reasoning About Actions

Other important tasks

- Consistency check
- Test of executability/inexecutability
- Theory change
- ...

Outline

Outline

Outline

Outline

Formalizing Domains

Several base formalisms

- Situation calculus [McCarthy & Hayes, 1969]
  - $\forall s.(Holds(loaded, s) \rightarrow \neg Holds(alive, do(shoot, s)))$
- Languages $\mathcal{A}$, $\mathcal{AR}$, etc. [Lifschitz *et al.*, 90's]
  - *shoot* causes $\neg$*alive* if *loaded*
- Fluent calculus [Thielscher, 1995]
  - $Poss(shoot(tk), s) \rightarrow$
    $State(do(shoot(tk), s)) = State(s) \circ dead(tk) - alive(tk)$
- ...

Formalizing Domains

In this work...

- we have chosen Modal Logic
  - Weak version of Propositional Dynamic Logic (PDL)
  - Simple and decidable
  - With a tableaux-based theorem prover: Lotrec

Logical Preliminaries

Ontology

- Actions: $\mathfrak{Act} = \{a_1, a_2, \ldots\}$
- Atomic propositions: $\mathfrak{Prop} = \{p_1, p_2, \ldots\}$
- Literals: $\mathfrak{Lit} = \mathfrak{Prop} \cup \{\neg p : p \in \mathfrak{Prop}\}$
- Classical formulas: $\mathfrak{Fml} = \{\varphi_1, \varphi_2, \ldots\}$

Action operators

For each $a \in \mathfrak{Act}$, a modal operator $[a]$

- $[a]\varphi$: "after execution of a, φ is true"
- $\langle a \rangle \varphi =_{\text{def}} \neg [a] \neg \varphi$
- Complex formulas: $[a_1; a_2]\varphi$, $[a^*]\varphi$, ...

Logical Preliminaries

Ontology

- Actions: $\mathfrak{Act} = \{a_1, a_2, \ldots\}$
- Atomic propositions: $\mathfrak{Prop} = \{p_1, p_2, \ldots\}$
- Literals: $\mathfrak{Lit} = \mathfrak{Prop} \cup \{\neg p : p \in \mathfrak{Prop}\}$
- Classical formulas: $\mathfrak{Fml} = \{\varphi_1, \varphi_2, \ldots\}$

Action operators

For each $a \in \mathfrak{Act}$, a modal operator $[a]$

- $[a]\varphi$: "after execution of a, φ is true"
  - $[a]\bot$: "a is inexecutable"
- $\langle a\rangle\varphi =_{\text{def}} \neg[a]\neg\varphi$
  - $\langle a\rangle\top$: "a is executable"
- Complex formulas: $\Phi_1, \Phi_2, \ldots$

Logical Preliminaries

Ontology

- Actions: $\mathfrak{Act} = \{a_1, a_2, \ldots\}$
- Atomic propositions: $\mathfrak{Prop} = \{p_1, p_2, \ldots\}$
- Literals: $\mathfrak{Lit} = \mathfrak{Prop} \cup \{\neg p : p \in \mathfrak{Prop}\}$
- Classical formulas: $\mathfrak{Fml} = \{\varphi_1, \varphi_2, \ldots\}$

Action operators

For each $a \in \mathfrak{Act}$, a modal operator $[a]$

- $[a]\varphi$: "after execution of a, φ is true"
  - $[a]\bot$: "a is inexecutable"
- $\langle a\rangle\varphi =_{\text{def}} \neg[a]\neg\varphi$
  - $\langle a\rangle\top$: "a is executable"
- Complex formulas: $\Phi_1, \Phi_2, \ldots$

Logical Preliminaries

Ontology

- Actions: $\mathfrak{Act} = \{a_1, a_2, \ldots\}$
- Atomic propositions: $\mathfrak{Prop} = \{p_1, p_2, \ldots\}$
- Literals: $\mathfrak{Lit} = \mathfrak{Prop} \cup \{\neg p : p \in \mathfrak{Prop}\}$
- Classical formulas: $\mathfrak{Fml} = \{\varphi_1, \varphi_2, \ldots\}$

Action operators

For each $a \in \mathfrak{Act}$, a modal operator $[a]$

- $[a]\varphi$: "after execution of a, φ is true"
  - $[a]\bot$: "a is inexecutable"
- $\langle a\rangle\varphi =_{\text{def}} \neg[a]\neg\varphi$
  - $\langle a\rangle\top$: "a is executable"
- Complex formulas: $\Phi_1, \Phi_2, \ldots$

Logical Preliminaries

Example

- Actions: *shoot*, *tease*
- Propositions: *loaded*, *alive*, *walking*
- Formulas: $alive \wedge \neg walking$, $\langle tease \rangle \top$, $loaded \rightarrow [shoot] \neg alive$

Semantics

Multimodal logic K [Popkorn 94,Blackburn *et al.* 2001].

Definition

Models $\mathscr{M} = \langle W, R \rangle$ where

- $W \subseteq 2^{\mathfrak{Prop}}$: set of possible worlds (states)
- R: $\mathfrak{Act} \longrightarrow 2^{W \times W}$

Definition

- $\models^{\mathscr{M}}_{w} p$ (p is true at world w of model $\mathscr{M}$) iff $p \in w$
- $\models^{\mathscr{M}}_{w} [a]\Phi$ iff for every w' such that wR_aw', $\models^{\mathscr{M}}_{w'} \Phi$
- the usual truth conditions for the other connectives

Semantics

Multimodal logic K [Popkorn 94,Blackburn *et al.* 2001].

Definition

Models $\mathscr{M} = \langle W, R \rangle$ where

- $W \subseteq 2^{\mathfrak{Prop}}$: set of possible worlds (states)
- R: $\mathfrak{Act} \longrightarrow 2^{W \times W}$

Definition

- $\models^{\mathscr{M}}_{w} p$ (p is true at world w of model $\mathscr{M}$) iff $p \in w$
- $\models^{\mathscr{M}}_{w} [a]\Phi$ iff for every w' such that wR_aw', $\models^{\mathscr{M}}_{w'} \Phi$
- the usual truth conditions for the other connectives

Semantics

Example

If $\mathfrak{Act} = \{a_1, a_2\}$, and $\mathfrak{Prop} = \{p_1, p_2\}$, $\mathscr{M} = \langle W, R \rangle$, where

$$W = \{\{p_1, p_2\}, \{p_1, \neg p_2\}, \{\neg p_1, p_2\}\},$$

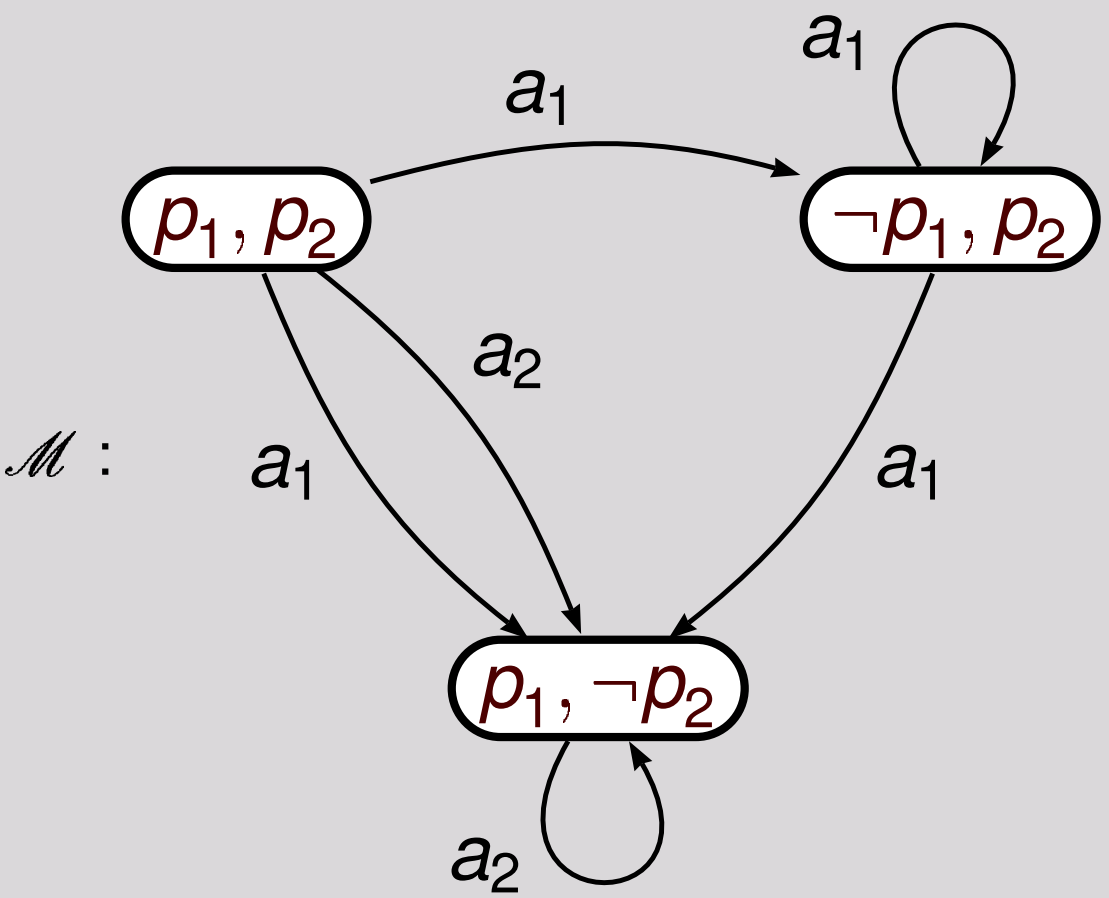
$$R(a_1) = \left\{ \begin{array}{c} (\{p_1, p_2\}, \{p_1, \neg p_2\}), (\{p_1, p_2\}, \{\neg p_1, p_2\}), \\ (\{\neg p_1, p_2\}, \{\neg p_1, p_2\}), (\{\neg p_1, p_2\}, \{p_1, \neg p_2\}) \end{array} \right\}$$

$$R(a_2) = \{(\{p_1, p_2\}, \{p_1, \neg p_2\}), (\{p_1, \neg p_2\}, \{p_1, \neg p_2\})\}$$

is a model

Semantics

Example



$\models^{\mathscr{M}} p_1 \vee p_2$

$\models^{\mathscr{M}} p_1 \rightarrow [a_2]\neg p_2$

$\models^{\mathscr{M}} \neg p_1 \rightarrow \langle a_1 \rangle \top$

$\models^{\mathscr{M}} \neg p_2 \rightarrow [a_1]\bot$

Semantics

Definition

- $\models^{\mathscr{M}} \Phi$ iff for all $w \in W$, $\models^{\mathscr{M}}_{w} \Phi$
- $\models^{\mathscr{M}} \Sigma$ iff $\models^{\mathscr{M}} \Phi$ for every $\Phi \in \Sigma$

Definition

Φ is a *consequence of the set of global axioms* Σ in all PDL-models (noted $\Sigma \models_{\mathrm{PDL}} \Phi$) iff for every $\mathscr{M}$, if $\models^{\mathscr{M}} \Sigma$, then $\models^{\mathscr{M}} \Phi$.

Semantics

Definition

- $\models^{\mathscr{M}} \Phi$ iff for all $w \in W$, $\models^{\mathscr{M}}_{w} \Phi$
- $\models^{\mathscr{M}} \Sigma$ iff $\models^{\mathscr{M}} \Phi$ for every $\Phi \in \Sigma$

Definition

Φ is a *consequence of the set of global axioms* Σ in all PDL-models (noted $\Sigma \models_{\mathrm{PDL}} \Phi$) iff for every $\mathscr{M}$, if $\models^{\mathscr{M}} \Sigma$, then $\models^{\mathscr{M}} \Phi$.

Outline

The Tale Again

Example

- A turkey that walks is alive: $walking \rightarrow alive$
- Teasing a turkey makes it to walk: $[tease]walking$
- It is always possible to tease a turkey: $\langle tease \rangle \top$
- A dead turkey remains dead after teasing it
  - $\neg alive \rightarrow [tease]\neg alive$
- If the gun is loaded, shooting kills the turkey
  - $loaded \rightarrow [shoot]\neg alive$
- Teasing does not unload the gun
  - $loaded \rightarrow [tease]loaded$

The Tale Again

Example

$$\left\{\begin{array}{c} \langle tease\rangle\top, \\ walking \rightarrow alive, \\ [tease]walking, \\ loaded \rightarrow [shoot]\neg alive, \\ \neg alive \rightarrow [tease]\neg alive, \\ loaded \rightarrow [tease]loaded \end{array}\right\} \models [tease]alive$$

$$\models \neg alive \rightarrow [tease](alive \wedge \neg alive)$$

$$\models \neg alive \rightarrow [tease]\bot$$

$$\models alive$$

N.B.: Such a description is consistent

What is the problem?

The Tale Again

Example

$$\left\{\begin{array}{c} \langle tease\rangle\top, \\ walking \rightarrow alive, \\ [tease]walking, \\ loaded \rightarrow [shoot]\neg alive, \\ \neg alive \rightarrow [tease]\neg alive, \\ loaded \rightarrow [tease]loaded \end{array}\right\} \models [tease]alive$$

$$\models \neg alive \rightarrow [tease](alive \wedge \neg alive)$$

$$\models \neg alive \rightarrow [tease]\bot$$

$$\models alive$$

N.B.: Such a description is consistent

What is the problem?

The Tale Again

Example

$$\left\{\begin{array}{c} \langle tease\rangle\top, \\ walking \rightarrow alive, \\ [tease]walking, \\ loaded \rightarrow [shoot]\neg alive, \\ \neg alive \rightarrow [tease]\neg alive, \\ loaded \rightarrow [tease]loaded \end{array}\right\} \models [tease]alive$$

$$\models \neg alive \rightarrow [tease](alive \wedge \neg alive)$$

$$\models \neg alive \rightarrow [tease]\bot$$

$$\models alive$$

N.B.: Such a description is consistent

What is the problem?

The Tale Again

Example

$$\left\{\begin{array}{c} \langle tease\rangle\top, \\ walking \rightarrow alive, \\ [tease]walking, \\ loaded \rightarrow [shoot]\neg alive, \\ \neg alive \rightarrow [tease]\neg alive, \\ loaded \rightarrow [tease]loaded \end{array}\right\} \begin{array}{l} \models [tease]alive \\ \models \neg alive \rightarrow [tease](alive \wedge \neg alive) \\ \models \neg alive \rightarrow [tease]\bot \\ \not\models alive \end{array}$$

N.B.: Such a description is consistent

What is the problem?

The Tale Again

Example

$$\left\{\begin{array}{c} \langle tease\rangle\top, \\ walking \rightarrow alive, \\ [tease]walking, \\ loaded \rightarrow [shoot]\neg alive, \\ \neg alive \rightarrow [tease]\neg alive, \\ loaded \rightarrow [tease]loaded \end{array}\right\} \begin{array}{l} \models [tease]alive \\ \models \neg alive \rightarrow [tease](alive \wedge \neg alive) \\ \models \neg alive \rightarrow [tease]\bot \\ \models alive \end{array}$$

N.B.: Such a description is consistent

What is the problem?

The Tale Again

Example

$$\left\{\begin{array}{c} \langle tease\rangle\top, \\ walking \rightarrow alive, \\ [tease]walking, \\ loaded \rightarrow [shoot]\neg alive, \\ \neg alive \rightarrow [tease]\neg alive, \\ loaded \rightarrow [tease]loaded \end{array}\right\} \begin{array}{l} \models [tease]alive \\ \models \neg alive \rightarrow [tease](alive \wedge \neg alive) \\ \models \neg alive \rightarrow [tease]\bot \\ \not\models alive \end{array}$$

N.B.: Such a description is consistent

What is the problem?

Outline

Natural Modules in Action Theories

Types of domain laws

Static laws : $walking \rightarrow alive$

Effect laws : $loaded \rightarrow [shoot]\neg alive$

Executability laws : $hasGun \rightarrow \langle shoot \rangle \top$

Inexecutability laws : $\neg hasGun \rightarrow [shoot]\bot$

only formulas of these types

Natural Modules in Action Theories

Types of domain laws

Static laws : $walking \rightarrow alive$

Effect laws : $loaded \rightarrow [shoot]\neg alive$

Executability laws : $hasGun \rightarrow \langle shoot \rangle \top$

Inexecutability laws : $\neg hasGun \rightarrow [shoot]\bot$

only formulas of these types

Natural Modules in Action Theories

Types of domain laws

Static laws : $walking \rightarrow alive$

Effect laws : $loaded \rightarrow [shoot]\neg alive$

Executability laws : $hasGun \rightarrow \langle shoot \rangle \top$

Inexecutability laws : $\neg hasGun \rightarrow [shoot]\bot$

only formulas of these types

Natural Modules in Action Theories

Types of domain laws

Static laws : $walking \rightarrow alive$

Effect laws : $loaded \rightarrow [shoot]\neg alive$

Executability laws : $hasGun \rightarrow \langle shoot \rangle \top$

Inexecutability laws : $\neg hasGun \rightarrow [shoot]\bot$

only formulas of these types

Natural Modules in Action Theories

Types of domain laws

Static laws : $walking \rightarrow alive$

Effect laws : $loaded \rightarrow [shoot]\neg alive$

Executability laws : $hasGun \rightarrow \langle shoot \rangle \top$

Inexecutability laws : $\neg hasGun \rightarrow [shoot]\bot$

only formulas of these types

Natural Modules in Action Theories

Defining modules

- $\mathcal{S}$: set of static laws
- Given $a \in \mathfrak{Act}$
  - $\mathcal{E}^a$: effect laws for a
  - $\mathcal{X}^a$: executability laws for a
  - $\mathcal{I}^a$: inexecutability laws for a
- $\langle \mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \rangle$: domain description for a
- $\mathcal{E} = \bigcup_{a \in \mathfrak{Act}} \mathcal{E}^a$, $\mathcal{X} = \bigcup_{a \in \mathfrak{Act}} \mathcal{X}^a$, and $\mathcal{I} = \bigcup_{a \in \mathfrak{Act}} \mathcal{I}^a$
- $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \rangle$: the action theory of a given domain

Natural Modules in Action Theories

Defining modules

- $\mathcal{S}$: set of static laws
- Given $a \in \mathfrak{Act}$
  - $\mathcal{E}^a$: effect laws for a
  - $\mathcal{X}^a$: executability laws for a
  - $\mathcal{I}^a$: inexecutability laws for a
- $\langle \mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \rangle$: domain description for a
- $\mathcal{E} = \bigcup_{a \in \mathfrak{Act}} \mathcal{E}^a$, $\mathcal{X} = \bigcup_{a \in \mathfrak{Act}} \mathcal{X}^a$, and $\mathcal{I} = \bigcup_{a \in \mathfrak{Act}} \mathcal{I}^a$
- $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \rangle$: the action theory of a given domain

Natural Modules in Action Theories

Defining modules

- $\mathcal{S}$: set of static laws
- Given $a \in \mathfrak{Act}$
  - $\mathcal{E}^a$: effect laws for a
  - $\mathcal{X}^a$: executability laws for a
  - $\mathcal{I}^a$: inexecutability laws for a
- $\langle \mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \rangle$: domain description for a
- $\mathcal{E} = \bigcup_{a \in \mathfrak{Act}} \mathcal{E}^a$, $\mathcal{X} = \bigcup_{a \in \mathfrak{Act}} \mathcal{X}^a$, and $\mathcal{I} = \bigcup_{a \in \mathfrak{Act}} \mathcal{I}^a$
- $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \rangle$: the action theory of a given domain

Natural Modules in Action Theories

Defining modules

- $\mathcal{S}$: set of static laws
- Given $a \in \mathfrak{Act}$
  - $\mathcal{E}^a$: effect laws for a
  - $\mathcal{X}^a$: executability laws for a
  - $\mathcal{I}^a$: inexecutability laws for a
- $\langle \mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \rangle$: domain description for a
- $\mathcal{E} = \bigcup_{a \in \mathfrak{Act}} \mathcal{E}^a$, $\mathcal{X} = \bigcup_{a \in \mathfrak{Act}} \mathcal{X}^a$, and $\mathcal{I} = \bigcup_{a \in \mathfrak{Act}} \mathcal{I}^a$
- $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \rangle$: the action theory of a given domain

What About the Frame Problem?

In our example

- If we had an action *wait*
  - $\mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \not\models_{\mathsf{PDL}} \mathit{loaded} \rightarrow [\mathit{wait}]\mathit{loaded}$

Definition

Dependence relation [Castilho *et al.* 99]: $\rightsquigarrow \subseteq \mathfrak{Act} \times \mathfrak{Lit}$

Example

- $\mathit{shoot} \rightsquigarrow \neg \mathit{alive}$, $\mathit{tease} \rightsquigarrow \mathit{walking}$, $\mathit{tease} \not\rightsquigarrow \mathit{alive}$
- From $\mathit{wait} \not\rightsquigarrow \neg \mathit{loaded}$ conclude $\mathit{loaded} \rightarrow [\mathit{wait}]\mathit{loaded}$

What About the Frame Problem?

In our example

- If we had an action *wait*
  - $\mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \not\models_{\mathsf{PDL}} \mathit{loaded} \rightarrow [\mathit{wait}]\mathit{loaded}$

Definition

Dependence relation [Castilho *et al.* 99]: $\rightsquigarrow \subseteq \mathfrak{Act} \times \mathfrak{Lit}$

Example

- $\mathit{shoot} \rightsquigarrow \neg \mathit{alive}$, $\mathit{tease} \rightsquigarrow \mathit{walking}$, $\mathit{tease} \not\rightsquigarrow \mathit{alive}$
- From $\mathit{wait} \not\rightsquigarrow \neg \mathit{loaded}$ conclude $\mathit{loaded} \rightarrow [\mathit{wait}]\mathit{loaded}$

What About the Frame Problem?

In our example

- If we had an action *wait*
  - $\mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \not\models_{\mathsf{PDL}} \mathit{loaded} \rightarrow [\mathit{wait}]\mathit{loaded}$

Definition

Dependence relation [Castilho *et al.* 99]: $\rightsquigarrow \subseteq \mathfrak{Act} \times \mathfrak{Lit}$

Example

- $\mathit{shoot} \rightsquigarrow \neg \mathit{alive}$, $\mathit{tease} \rightsquigarrow \mathit{walking}$, $\mathit{tease} \not\rightsquigarrow \mathit{alive}$
- From $\mathit{wait} \not\rightsquigarrow \neg \mathit{loaded}$ conclude $\mathit{loaded} \rightarrow [\mathit{wait}]\mathit{loaded}$

What About the Frame Problem?

In our example

- If we had an action *wait*
  - $\mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \not\models_{\mathsf{PDL}} \mathit{loaded} \rightarrow [\mathit{wait}]\mathit{loaded}$

Definition

Dependence relation [Castilho *et al.* 99]: $\rightsquigarrow \subseteq \mathfrak{Act} \times \mathfrak{Lit}$

Example

- $\mathit{shoot} \rightsquigarrow \neg \mathit{alive}$, $\mathit{tease} \rightsquigarrow \mathit{walking}$, $\mathit{tease} \not\rightsquigarrow \mathit{alive}$
- From $\mathit{wait} \not\rightsquigarrow \neg \mathit{loaded}$ conclude $\mathit{loaded} \rightarrow [\mathit{wait}]\mathit{loaded}$

What About the Frame Problem?

Restriction on models

For all wR_aw':

- $\not\models^{\mathcal{M}}_{w} p$ implies $\not\models^{\mathcal{M}}_{w'} p$, if $a \not\rightsquigarrow p$
- $\models^{\mathcal{M}}_{w} p$ implies $\models^{\mathcal{M}}_{w'} p$, if $a \not\rightsquigarrow \neg p$.

New logical consequence

- $\models_{\rightsquigarrow}$ instead of $\models_{\mathrm{PDL}}$

Example

$\mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \models_{\rightsquigarrow} \mathit{loaded} \rightarrow [\mathit{wait}]\mathit{loaded}$

What About the Frame Problem?

Restriction on models

For all wR_aw':

- $\not\models^{\mathcal{M}}_{w} p$ implies $\not\models^{\mathcal{M}}_{w'} p$, if $a \not\rightsquigarrow p$
- $\models^{\mathcal{M}}_{w} p$ implies $\models^{\mathcal{M}}_{w'} p$, if $a \not\rightsquigarrow \neg p$.

New logical consequence

- $\models_{\rightsquigarrow}$ instead of $\models_{\mathrm{PDL}}$

Example

$\mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \models_{\rightsquigarrow} \mathit{loaded} \rightarrow [\mathit{wait}]\mathit{loaded}$

What About the Frame Problem?

Restriction on models

For all wR_aw':

- $\not\models^{\mathcal{M}}_{w} p$ implies $\not\models^{\mathcal{M}}_{w'} p$, if $a \not\rightsquigarrow p$
- $\models^{\mathcal{M}}_{w} p$ implies $\models^{\mathcal{M}}_{w'} p$, if $a \not\rightsquigarrow \neg p$.

New logical consequence

- $\models_{\rightsquigarrow}$ instead of $\models_{\mathrm{PDL}}$

Example

$\mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \models_{\rightsquigarrow} \mathit{loaded} \rightarrow [\mathit{wait}]\mathit{loaded}$

What About the Frame Problem?

The dependence-based approach...

- solves the frame problem
- subsumes Reiter's regression [Demolombe *et al.* 2003]
- does not entirely solve the ramification problem
  - e.g. $shoot \rightsquigarrow \neg walking$
- But is the only approach that works for domains with actions with both indeterminate and indirect effects [Castilho *et al.* 2002], [Herzig & Varzinczak 2004]

What About the Frame Problem?

The dependence-based approach...

- solves the frame problem
- subsumes Reiter's regression [Demolombe *et al.* 2003]
- does not entirely solve the ramification problem
  - e.g. $\mathit{shoot} \rightsquigarrow \neg \mathit{walking}$
- But is the only approach that works for domains with actions with both indeterminate and indirect effects [Castilho *et al.* 2002], [Herzig & Varzinczak 2004]

What About the Frame Problem?

The dependence-based approach...

- solves the frame problem
- subsumes Reiter's regression [Demolombe *et al.* 2003]
- does not entirely solve the ramification problem
  - e.g. $shoot \rightsquigarrow \neg walking$
- But is the only approach that works for domains with actions with both indeterminate and indirect effects [Castilho *et al.* 2002], [Herzig & Varzinczak 2004]

Outline

Consistency and More

Postulates

PC (*Consistency*): $\mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \not\models_{\rightsquigarrow} \bot$

PS (*No implicit static laws*): if $\mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \models_{\rightsquigarrow} \varphi$, then $\mathcal{S} \models \varphi$

PI (*No implicit inexecutability laws*):

if $\mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \models_{\rightsquigarrow} \varphi \rightarrow [a]\bot$,
then $\mathcal{S}, \mathcal{I}^a \models_{\mathrm{PDL}} \varphi \rightarrow [a]\bot$

PX (*No implicit executability laws*):

if $\mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \models_{\rightsquigarrow} \varphi \rightarrow \langle a\rangle\top$,
then $\mathcal{S}, \mathcal{X}^a \models_{\mathrm{PDL}} \varphi \rightarrow \langle a\rangle\top$

Motivation

- Better control what is going on

Consistency and More

Postulates

PC (*Consistency*): $\mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \not\models_{\rightsquigarrow} \bot$

PS (*No implicit static laws*): if $\mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \models_{\rightsquigarrow} \varphi$, then $\mathcal{S} \models \varphi$

PI (*No implicit inexecutability laws*):
if $\mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \models_{\rightsquigarrow} \varphi \rightarrow [a]\bot$,
then $\mathcal{S}, \mathcal{I}^a \models_{\mathrm{PDL}} \varphi \rightarrow [a]\bot$

PX (*No implicit executability laws*):
if $\mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \models_{\rightsquigarrow} \varphi \rightarrow \langle a \rangle \top$,
then $\mathcal{S}, \mathcal{X}^a \models_{\mathrm{PDL}} \varphi \rightarrow \langle a \rangle \top$

Motivation

- Better control what is going on

Consistency and More

Postulates

PC (*Consistency*): $\mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \not\models_{\rightsquigarrow} \bot$

PS (*No implicit static laws*): if $\mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \models_{\rightsquigarrow} \varphi$, then $\mathcal{S} \models \varphi$

PI (*No implicit inexecutability laws*):

if $\mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \models_{\rightsquigarrow} \varphi \rightarrow [a]\bot$,
then $\mathcal{S}, \mathcal{I}^a \models_{\mathrm{PDL}} \varphi \rightarrow [a]\bot$

PX (*No implicit executability laws*):

if $\mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \models_{\rightsquigarrow} \varphi \rightarrow \langle a\rangle\top$,
then $\mathcal{S}, \mathcal{X}^a \models_{\mathrm{PDL}} \varphi \rightarrow \langle a\rangle\top$

Motivation

- Better control what is going on

Consistency and More

Postulates

PC (*Consistency*): $\mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \not\models_{\rightsquigarrow} \bot$

PS (*No implicit static laws*): if $\mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \models_{\rightsquigarrow} \varphi$, then $\mathcal{S} \models \varphi$

PI (*No implicit inexecutability laws*):

if $\mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \models_{\rightsquigarrow} \varphi \rightarrow [a]\bot$,
then $\mathcal{S}, \mathcal{I}^a \models_{\mathrm{PDL}} \varphi \rightarrow [a]\bot$

PX (*No implicit executability laws*):

if $\mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \models_{\rightsquigarrow} \varphi \rightarrow \langle a \rangle \top$,
then $\mathcal{S}, \mathcal{X}^a \models_{\mathrm{PDL}} \varphi \rightarrow \langle a \rangle \top$

Motivation

- Better control what is going on

Consistency and More

Postulates

PC (*Consistency*): $\mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \not\models_{\rightsquigarrow} \bot$

PS (*No implicit static laws*): if $\mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \models_{\rightsquigarrow} \varphi$, then $\mathcal{S} \models \varphi$

PI (*No implicit inexecutability laws*):

if $\mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \models_{\rightsquigarrow} \varphi \rightarrow [a]\bot$,
then $\mathcal{S}, \mathcal{I}^a \models_{\mathrm{PDL}} \varphi \rightarrow [a]\bot$

PX (*No implicit executability laws*):

if $\mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \models_{\rightsquigarrow} \varphi \rightarrow \langle a \rangle \top$,
then $\mathcal{S}, \mathcal{X}^a \models_{\mathrm{PDL}} \varphi \rightarrow \langle a \rangle \top$

Motivation

- Better control what is going on

No Implicit Static Laws

Example

$$\mathcal{S} = \{walking \rightarrow alive\},\ \mathcal{E} = \left\{ \begin{array}{c} [tease]walking, \\ loaded \rightarrow [shoot]\neg alive \end{array} \right\},$$

$$\mathcal{X} = \{\langle tease\rangle \top\},\ \mathcal{I} = \{\neg alive \rightarrow [tease]\bot\}$$

$$tease \rightsquigarrow walking, shoot \rightsquigarrow \neg alive$$

- $\mathcal{S}, \mathcal{E}^{tease}, \mathcal{X}^{tease}, \mathcal{I}^{tease} \models_{\rightsquigarrow} alive$
- But $\mathcal{S} \not\models alive$

Postulate **PS** violated

No Implicit Static Laws

Example

$$\mathcal{S} = \{walking \rightarrow alive\},\ \mathcal{E} = \left\{ \begin{array}{c} [tease]walking, \\ loaded \rightarrow [shoot]\neg alive \end{array} \right\},$$

$$\mathcal{X} = \{\langle tease\rangle \top\},\ \mathcal{I} = \{\neg alive \rightarrow [tease]\bot\}$$

$$tease \rightsquigarrow walking, shoot \rightsquigarrow \neg alive$$

- $\mathcal{S}, \mathcal{E}^{tease}, \mathcal{X}^{tease}, \mathcal{I}^{tease} \models_{\rightsquigarrow} alive$
- But $\mathcal{S} \not\models alive$

Postulate **PS** violated

No Implicit Static Laws

Example

$$\mathcal{S} = \{walking \rightarrow alive\},\ \mathcal{E} = \left\{ \begin{array}{c} [tease]walking, \\ loaded \rightarrow [shoot]\neg alive \end{array} \right\},$$

$$\mathcal{X} = \{\langle tease\rangle\top\},\ \mathcal{I} = \{\neg alive \rightarrow [tease]\bot\}$$

$$tease \rightsquigarrow walking, shoot \rightsquigarrow \neg alive$$

- $\mathcal{S}, \mathcal{E}^{tease}, \mathcal{X}^{tease}, \mathcal{I}^{tease} \models_{\rightsquigarrow} alive$
- But $\mathcal{S} \not\models alive$

Postulate **PS** violated

No Implicit Static Laws

Example

$$\mathcal{S} = \{walking \rightarrow alive\},\ \mathcal{E} = \left\{ \begin{array}{c} [tease]walking, \\ loaded \rightarrow [shoot]\neg alive \end{array} \right\},$$

$$\mathcal{X} = \{\langle tease\rangle\top\},\ \mathcal{I} = \{\neg alive \rightarrow [tease]\bot\}$$

$$tease \rightsquigarrow walking, shoot \rightsquigarrow \neg alive$$

- $\mathcal{S}, \mathcal{E}^{tease}, \mathcal{X}^{tease}, \mathcal{I}^{tease} \models_{\rightsquigarrow} alive$
- But $\mathcal{S} \not\models alive$

Postulate **PS** violated

No Implicit Static Laws

Idea of algorithm

For each $\varphi \rightarrow \langle a \rangle \top$

1. find $\varphi' \rightarrow [a]\bot$ entailed by the theory
2. if $\varphi \wedge \varphi'$ is consistent with $\mathcal{S}$
   - $\neg(\varphi \wedge \varphi')$ is possibly an implicit law

Result: the set of all implicit static laws

No Implicit Static Laws

Idea of algorithm

For each $\varphi \rightarrow \langle a \rangle \top$

1. find $\varphi' \rightarrow [a]\bot$ entailed by the theory
2. if $\varphi \wedge \varphi'$ is consistent with $\mathcal{S}$
   - $\neg(\varphi \wedge \varphi')$ is possibly an implicit law

Result: the set of all implicit static laws

No Implicit Static Laws

Algorithm 1: Finding all implicit static laws induced by a

input: $\langle \mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \rangle$ and $\rightsquigarrow$
output: $\mathcal{S}_{imp^*}$, the set of all implicit static laws of $\langle \mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \rangle$
calls: $NewCons_\varphi(\psi) = Pl(\varphi \wedge \psi) \setminus Pl(\varphi)$

$\mathcal{S}_{imp^*} := \emptyset$
repeat
  $\mathcal{S}_{imp} := \emptyset$
  for all $\varphi \rightarrow \langle a \rangle \top \in \mathcal{X}^a$ **do**
    for all $\mathcal{C}^{a\prime} \subseteq \mathcal{E}^a \cup \mathcal{I}^a$ **do**
      $\varphi_{\mathcal{C}^{a\prime}} := \bigwedge \{\varphi_i : \varphi_i \rightarrow [a]\psi_i \in \mathcal{C}^{a\prime}\}$
      $\psi_{\mathcal{C}^{a\prime}} := \bigwedge \{\psi_i : \varphi_i \rightarrow [a]\psi_i \in \mathcal{C}^{a\prime}\}$
      for all $\chi \in NewCons_{\mathcal{S}}(\psi_{\mathcal{C}^{a\prime}})$ **do**
        if $\mathcal{S} \cup \mathcal{S}_{imp^*} \cup \{\varphi, \varphi_{\mathcal{C}^{a\prime}}, \neg\chi\} \nvdash \bot$ **and** $\forall l_i \in \chi,\ a \not\rightsquigarrow l_i$ **then**
          $\mathcal{S}_{imp} := \mathcal{S}_{imp} \cup \{\neg(\varphi \wedge \varphi_{\mathcal{C}^{a\prime}} \wedge \neg\chi)\}$
  $\mathcal{S}_{imp^*} := \mathcal{S}_{imp^*} \cup \mathcal{S}_{imp}$
until $\mathcal{S}_{imp} = \emptyset$

No Implicit Static Laws

Algorithm 1: Finding all implicit static laws induced by a

input: $\langle \mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \rangle$ and $\rightsquigarrow$
output: $\mathcal{S}_{imp^*}$, the set of all implicit static laws of $\langle \mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \rangle$
calls: $NewCons_\varphi(\psi) = Pl(\varphi \wedge \psi) \setminus Pl(\varphi)$

$\mathcal{S}_{imp^*} := \emptyset$
repeat
  $\mathcal{S}_{imp} := \emptyset$
  for all $\varphi \rightarrow \langle a \rangle \top \in \mathcal{X}^a$ **do**
    for all $\mathcal{C}^{a\prime} \subseteq \mathcal{E}^a \cup \mathcal{I}^a$ **do**
      $\varphi_{\mathcal{C}^{a\prime}} := \bigwedge \{\varphi_i : \varphi_i \rightarrow [a]\psi_i \in \mathcal{C}^{a\prime}\}$
      $\psi_{\mathcal{C}^{a\prime}} := \bigwedge \{\psi_i : \varphi_i \rightarrow [a]\psi_i \in \mathcal{C}^{a\prime}\}$
      for all $\chi \in NewCons_{\mathcal{S}}(\psi_{\mathcal{C}^{a\prime}})$ **do**
        if $\mathcal{S} \cup \mathcal{S}_{imp^*} \cup \{\varphi, \varphi_{\mathcal{C}^{a\prime}}, \neg\chi\} \nvdash \bot$ **and** $\forall l_i \in \chi,\ a \not\rightsquigarrow l_i$ **then**
          $\mathcal{S}_{imp} := \mathcal{S}_{imp} \cup \{\neg(\varphi \wedge \varphi_{\mathcal{C}^{a\prime}} \wedge \neg\chi)\}$
  $\mathcal{S}_{imp^*} := \mathcal{S}_{imp^*} \cup \mathcal{S}_{imp}$
until $\mathcal{S}_{imp} = \emptyset$

No Implicit Static Laws

Algorithm 1: Finding all implicit static laws induced by a

input: $\langle \mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \rangle$ and $\rightsquigarrow$
output: $\mathcal{S}_{imp^*}$, the set of all implicit static laws of $\langle \mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \rangle$
calls: $NewCons_\varphi(\psi) = Pl(\varphi \wedge \psi) \setminus Pl(\varphi)$

$\mathcal{S}_{imp^*} := \emptyset$
repeat
  $\mathcal{S}_{imp} := \emptyset$
  for all $\varphi \rightarrow \langle a \rangle \top \in \mathcal{X}^a$ **do**
    for all $\mathcal{C}^{a\prime} \subseteq \mathcal{E}^a \cup \mathcal{I}^a$ **do**
      $\varphi_{\mathcal{C}^{a\prime}} := \bigwedge \{\varphi_i : \varphi_i \rightarrow [a]\psi_i \in \mathcal{C}^{a\prime}\}$
      $\psi_{\mathcal{C}^{a\prime}} := \bigwedge \{\psi_i : \varphi_i \rightarrow [a]\psi_i \in \mathcal{C}^{a\prime}\}$
      for all $\chi \in NewCons_{\mathcal{S}}(\psi_{\mathcal{C}^{a\prime}})$ **do**
        if $\mathcal{S} \cup \mathcal{S}_{imp^*} \cup \{\varphi, \varphi_{\mathcal{C}^{a\prime}}, \neg\chi\} \nvdash \bot$ **and** $\forall l_i \in \chi,\ a \not\rightsquigarrow l_i$ **then**
          $\mathcal{S}_{imp} := \mathcal{S}_{imp} \cup \{\neg(\varphi \wedge \varphi_{\mathcal{C}^{a\prime}} \wedge \neg\chi)\}$
  $\mathcal{S}_{imp^*} := \mathcal{S}_{imp^*} \cup \mathcal{S}_{imp}$
until $\mathcal{S}_{imp} = \emptyset$

No Implicit Static Laws

Algorithm 1: Finding all implicit static laws induced by a

input: $\langle \mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \rangle$ and $\rightsquigarrow$
output: $\mathcal{S}_{imp^*}$, the set of all implicit static laws of $\langle \mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \rangle$
calls: $NewCons_\varphi(\psi) = Pl(\varphi \wedge \psi) \setminus Pl(\varphi)$

$\mathcal{S}_{imp^*} := \emptyset$
repeat
  $\mathcal{S}_{imp} := \emptyset$
  for all $\varphi \rightarrow \langle a \rangle \top \in \mathcal{X}^a$ **do**
    for all $\mathcal{C}^{a\prime} \subseteq \mathcal{E}^a \cup \mathcal{I}^a$ **do**
      $\varphi_{\mathcal{C}^{a\prime}} := \bigwedge \{\varphi_i : \varphi_i \rightarrow [a]\psi_i \in \mathcal{C}^{a\prime}\}$
      $\psi_{\mathcal{C}^{a\prime}} := \bigwedge \{\psi_i : \varphi_i \rightarrow [a]\psi_i \in \mathcal{C}^{a\prime}\}$
      for all $\chi \in NewCons_{\mathcal{S}}(\psi_{\mathcal{C}^{a\prime}})$ **do**
        if $\mathcal{S} \cup \mathcal{S}_{imp^*} \cup \{\varphi, \varphi_{\mathcal{C}^{a\prime}}, \neg\chi\} \nvdash \bot$ **and** $\forall l_i \in \chi,\ a \not\rightsquigarrow l_i$ **then**
          $\mathcal{S}_{imp} := \mathcal{S}_{imp} \cup \{\neg(\varphi \wedge \varphi_{\mathcal{C}^{a\prime}} \wedge \neg\chi)\}$
  $\mathcal{S}_{imp^*} := \mathcal{S}_{imp^*} \cup \mathcal{S}_{imp}$
until $\mathcal{S}_{imp} = \emptyset$

No Implicit Static Laws

Algorithm 1: Finding all implicit static laws induced by a

input: $\langle \mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \rangle$ and $\rightsquigarrow$
output: $\mathcal{S}_{imp^*}$, the set of all implicit static laws of $\langle \mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \rangle$
calls: $NewCons_{\varphi}(\psi) = Pl(\varphi \wedge \psi) \setminus Pl(\varphi)$

$\mathcal{S}_{imp^*} := \emptyset$
repeat
  $\mathcal{S}_{imp} := \emptyset$
  for all $\varphi \rightarrow \langle a \rangle \top \in \mathcal{X}^a$ **do**
    for all $\mathcal{C}^{a\prime} \subseteq \mathcal{E}^a \cup \mathcal{I}^a$ **do**
      $\varphi_{\mathcal{C}^{a\prime}} := \bigwedge \{\varphi_i : \varphi_i \rightarrow [a]\psi_i \in \mathcal{C}^{a\prime}\}$
      $\psi_{\mathcal{C}^{a\prime}} := \bigwedge \{\psi_i : \varphi_i \rightarrow [a]\psi_i \in \mathcal{C}^{a\prime}\}$
      for all $\chi \in NewCons_{\mathcal{S}}(\psi_{\mathcal{C}^{a\prime}})$ **do**
        if $\mathcal{S} \cup \mathcal{S}_{imp^*} \cup \{\varphi, \varphi_{\mathcal{C}^{a\prime}}, \neg\chi\} \nvdash \bot$ **and** $\forall l_i \in \chi,\ a \not\rightsquigarrow l_i$ **then**
          $\mathcal{S}_{imp} := \mathcal{S}_{imp} \cup \{\neg(\varphi \wedge \varphi_{\mathcal{C}^{a\prime}} \wedge \neg\chi)\}$
  $\mathcal{S}_{imp^*} := \mathcal{S}_{imp^*} \cup \mathcal{S}_{imp}$
until $\mathcal{S}_{imp} = \emptyset$

No Implicit Static Laws

Algorithm 1: Finding all implicit static laws induced by a

input: $\langle \mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \rangle$ and $\rightsquigarrow$
output: $\mathcal{S}_{imp^*}$, the set of all implicit static laws of $\langle \mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \rangle$
calls: $NewCons_\varphi(\psi) = Pl(\varphi \wedge \psi) \setminus Pl(\varphi)$

$\mathcal{S}_{imp^*} := \emptyset$
repeat
  $\mathcal{S}_{imp} := \emptyset$
  for all $\varphi \rightarrow \langle a \rangle \top \in \mathcal{X}^a$ **do**
    for all $\mathcal{C}^{a\prime} \subseteq \mathcal{E}^a \cup \mathcal{I}^a$ **do**
      $\varphi_{\mathcal{C}^{a\prime}} := \bigwedge \{\varphi_i : \varphi_i \rightarrow [a]\psi_i \in \mathcal{C}^{a\prime}\}$
      $\psi_{\mathcal{C}^{a\prime}} := \bigwedge \{\psi_i : \varphi_i \rightarrow [a]\psi_i \in \mathcal{C}^{a\prime}\}$
      for all $\chi \in NewCons_{\mathcal{S}}(\psi_{\mathcal{C}^{a\prime}})$ **do**
        if $\mathcal{S} \cup \mathcal{S}_{imp^*} \cup \{\varphi, \varphi_{\mathcal{C}^{a\prime}}, \neg\chi\} \nvdash \bot$ **and** $\forall l_i \in \chi,\ a \not\rightsquigarrow l_i$ **then**
          $\mathcal{S}_{imp} := \mathcal{S}_{imp} \cup \{\neg(\varphi \wedge \varphi_{\mathcal{C}^{a\prime}} \wedge \neg\chi)\}$
  $\mathcal{S}_{imp^*} := \mathcal{S}_{imp^*} \cup \mathcal{S}_{imp}$
until $\mathcal{S}_{imp} = \emptyset$

No Implicit Static Laws

Algorithm 1: Finding all implicit static laws induced by a

input: $\langle \mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \rangle$ and $\rightsquigarrow$
output: $\mathcal{S}_{imp^*}$, the set of all implicit static laws of $\langle \mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \rangle$
calls: $NewCons_\varphi(\psi) = Pl(\varphi \wedge \psi) \setminus Pl(\varphi)$

$\mathcal{S}_{imp^*} := \emptyset$
repeat
  $\mathcal{S}_{imp} := \emptyset$
  for all $\varphi \rightarrow \langle a \rangle \top \in \mathcal{X}^a$ **do**
    for all $\mathcal{C}^{a\prime} \subseteq \mathcal{E}^a \cup \mathcal{I}^a$ **do**
      $\varphi_{\mathcal{C}^{a\prime}} := \bigwedge \{\varphi_i : \varphi_i \rightarrow [a]\psi_i \in \mathcal{C}^{a\prime}\}$
      $\psi_{\mathcal{C}^{a\prime}} := \bigwedge \{\psi_i : \varphi_i \rightarrow [a]\psi_i \in \mathcal{C}^{a\prime}\}$
      for all $\chi \in NewCons_{\mathcal{S}}(\psi_{\mathcal{C}^{a\prime}})$ **do**
        if $\mathcal{S} \cup \mathcal{S}_{imp^*} \cup \{\varphi, \varphi_{\mathcal{C}^{a\prime}}, \neg\chi\} \nvdash \bot$ **and** $\forall l_i \in \chi,\ a \not\rightsquigarrow l_i$ **then**
          $\mathcal{S}_{imp} := \mathcal{S}_{imp} \cup \{\neg(\varphi \wedge \varphi_{\mathcal{C}^{a\prime}} \wedge \neg\chi)\}$
  $\mathcal{S}_{imp^*} := \mathcal{S}_{imp^*} \cup \mathcal{S}_{imp}$
until $\mathcal{S}_{imp} = \emptyset$

No Implicit Static Laws

Algorithm 1: Finding all implicit static laws induced by a

input: $\langle \mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \rangle$ and $\rightsquigarrow$
output: $\mathcal{S}_{imp^*}$, the set of all implicit static laws of $\langle \mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \rangle$
calls: $NewCons_\varphi(\psi) = Pl(\varphi \wedge \psi) \setminus Pl(\varphi)$

$\mathcal{S}_{imp^*} := \emptyset$
repeat
  $\mathcal{S}_{imp} := \emptyset$
  for all $\varphi \rightarrow \langle a \rangle \top \in \mathcal{X}^a$ **do**
    for all $\mathcal{C}^{a\prime} \subseteq \mathcal{E}^a \cup \mathcal{I}^a$ **do**
      $\varphi_{\mathcal{C}^{a\prime}} := \bigwedge \{\varphi_i : \varphi_i \rightarrow [a]\psi_i \in \mathcal{C}^{a\prime}\}$
      $\psi_{\mathcal{C}^{a\prime}} := \bigwedge \{\psi_i : \varphi_i \rightarrow [a]\psi_i \in \mathcal{C}^{a\prime}\}$
      for all $\chi \in NewCons_{\mathcal{S}}(\psi_{\mathcal{C}^{a\prime}})$ **do**
        if $\mathcal{S} \cup \mathcal{S}_{imp^*} \cup \{\varphi, \varphi_{\mathcal{C}^{a\prime}}, \neg\chi\} \nvdash \bot$ **and** $\forall l_i \in \chi,\ a \not\rightsquigarrow l_i$ **then**
          $\mathcal{S}_{imp} := \mathcal{S}_{imp} \cup \{\neg(\varphi \wedge \varphi_{\mathcal{C}^{a\prime}} \wedge \neg\chi)\}$
  $\mathcal{S}_{imp^*} := \mathcal{S}_{imp^*} \cup \mathcal{S}_{imp}$
until $\mathcal{S}_{imp} = \emptyset$

No Implicit Static Laws

Example

$$\mathcal{S} = \{walking \rightarrow alive\},\ \mathcal{E} = \left\{ \begin{array}{c} [tease]walking, \\ loaded \rightarrow [shoot]\neg alive \end{array} \right\},$$

$$\mathcal{X} = \{\langle tease\rangle\top\},\ \mathcal{I} = \{\neg alive \rightarrow [tease]\bot\}$$

$$tease \rightsquigarrow walking, shoot \rightsquigarrow \neg alive$$

For $\langle tease\rangle\top$ and $[tease]walking$:

- $NewCons_{\mathcal{S}}(walking) = alive$: $[tease]alive$
- $tease \not\rightsquigarrow alive$: $\neg alive \rightarrow [tease]\neg alive$
- Hence $\neg alive \rightarrow [tease]\bot$
- $\mathcal{S} \cup \{\top \wedge \neg alive\} \not\models \bot$: $\mathcal{S} \not\models alive$
- $\mathcal{S}_{imp} = \{alive\}$

No Implicit Static Laws

Example

$$\mathcal{S} = \{walking \rightarrow alive\},\ \mathcal{E} = \left\{ \begin{array}{c} [tease]walking, \\ loaded \rightarrow [shoot]\neg alive \end{array} \right\},$$

$$\mathcal{X} = \{\langle tease \rangle \top\},\ \mathcal{I} = \{\neg alive \rightarrow [tease]\bot\}$$

$$tease \rightsquigarrow walking, shoot \rightsquigarrow \neg alive$$

For $\langle tease \rangle \top$ and $[tease]walking$:

- $NewCons_{\mathcal{S}}(walking) = alive$: $[tease]alive$
- $tease \not\rightsquigarrow alive$: $\neg alive \rightarrow [tease]\neg alive$
- Hence $\neg alive \rightarrow [tease]\bot$
- $\mathcal{S} \cup \{\top \wedge \neg alive\} \not\models \bot$: $\mathcal{S} \not\models alive$
- $\mathcal{S}_{imp} = \{alive\}$

No Implicit Static Laws

Example

$$\mathcal{S} = \{walking \rightarrow alive\},\ \mathcal{E} = \left\{ \begin{array}{c} [tease]walking, \\ loaded \rightarrow [shoot]\neg alive \end{array} \right\},$$

$$\mathcal{X} = \{\langle tease \rangle \top\},\ \mathcal{I} = \{\neg alive \rightarrow [tease]\bot\}$$

$$tease \rightsquigarrow walking, shoot \rightsquigarrow \neg alive$$

For $\langle tease \rangle \top$ and $[tease]walking$:

- $NewCons_{\mathcal{S}}(walking) = alive$: $[tease]alive$
- $tease \not\rightsquigarrow alive$: $\neg alive \rightarrow [tease]\neg alive$
- Hence $\neg alive \rightarrow [tease]\bot$
- $\mathcal{S} \cup \{\top \wedge \neg alive\} \not\models \bot$: $\mathcal{S} \not\models alive$
- $\mathcal{S}_{imp} = \{alive\}$

No Implicit Static Laws

Example

$$\mathcal{S} = \{walking \rightarrow alive\},\ \mathcal{E} = \left\{ \begin{array}{c} [tease]walking, \\ loaded \rightarrow [shoot]\neg alive \end{array} \right\},$$

$$\mathcal{X} = \{\langle tease \rangle \top\},\ \mathcal{I} = \{\neg alive \rightarrow [tease]\bot\}$$

$$tease \rightsquigarrow walking, shoot \rightsquigarrow \neg alive$$

For $\langle tease \rangle \top$ and $[tease]walking$:

- $NewCons_{\mathcal{S}}(walking) = alive$: $[tease]alive$
- $tease \not\rightsquigarrow alive$: $\neg alive \rightarrow [tease]\neg alive$
- Hence $\neg alive \rightarrow [tease]\bot$
- $\mathcal{S} \cup \{\top \wedge \neg alive\} \not\models \bot$: $\mathcal{S} \not\models alive$
- $\mathcal{S}_{imp} = \{alive\}$

No Implicit Static Laws

Example

$$\mathcal{S} = \{walking \rightarrow alive\},\ \mathcal{E} = \left\{ \begin{array}{c} [tease]walking, \\ loaded \rightarrow [shoot]\neg alive \end{array} \right\},$$

$$\mathcal{X} = \{\langle tease\rangle\top\},\ \mathcal{I} = \{\neg alive \rightarrow [tease]\bot\}$$

$$tease \rightsquigarrow walking, shoot \rightsquigarrow \neg alive$$

For $\langle tease\rangle\top$ and $[tease]walking$:

- $NewCons_{\mathcal{S}}(walking) = alive$: $[tease]alive$
- $tease \not\rightsquigarrow alive$: $\neg alive \rightarrow [tease]\neg alive$
- Hence $\neg alive \rightarrow [tease]\bot$
- $\mathcal{S} \cup \{\top \wedge \neg alive\} \not\models \bot$: $\mathcal{S} \not\models alive$
- $\mathcal{S}_{imp} = \{alive\}$

No Implicit Static Laws

Example

$$\mathcal{S} = \{walking \rightarrow alive\},\ \mathcal{E} = \left\{ \begin{array}{c} [tease]walking, \\ loaded \rightarrow [shoot]\neg alive \end{array} \right\},$$

$$\mathcal{X} = \{\langle tease \rangle \top\},\ \mathcal{I} = \{\neg alive \rightarrow [tease]\bot\}$$

$$tease \rightsquigarrow walking, shoot \rightsquigarrow \neg alive$$

For $\langle tease \rangle \top$ and $[tease]walking$:

- $NewCons_{\mathcal{S}}(walking) = alive$: $[tease]alive$
- $tease \not\rightsquigarrow alive$: $\neg alive \rightarrow [tease]\neg alive$
- Hence $\neg alive \rightarrow [tease]\bot$
- $\mathcal{S} \cup \{\top \wedge \neg alive\} \not\models \bot$: $\mathcal{S} \not\models alive$
- $\mathcal{S}_{imp} = \{alive\}$

No Implicit Static Laws

Example

$$\mathcal{S} = \{walking \rightarrow alive\},\ \mathcal{E} = \left\{ \begin{array}{c} [tease]walking, \\ loaded \rightarrow [shoot]\neg alive \end{array} \right\},$$

$$\mathcal{X} = \{\langle tease \rangle \top\},\ \mathcal{I} = \{\neg alive \rightarrow [tease]\bot\}$$

$$tease \rightsquigarrow walking, shoot \rightsquigarrow \neg alive$$

For $\langle tease \rangle \top$ and $[tease]walking$:

- $NewCons_{\mathcal{S}}(walking) = alive$: $[tease]alive$
- $tease \not\rightsquigarrow alive$: $\neg alive \rightarrow [tease]\neg alive$
- Hence $\neg alive \rightarrow [tease]\bot$
- $\mathcal{S} \cup \{\top \wedge \neg alive\} \not\models \bot$: $\mathcal{S} \not\models alive$
- $\mathcal{S}_{imp} = \{alive\}$

No Implicit Static Laws

Example (cont.)

Alternatives for repairing:

- $\mathcal{S} := \mathcal{S} \cup \{\mathit{alive}\}$
- add $\mathit{tease} \rightsquigarrow \mathit{alive}$
- weaken $[\mathit{tease}]\mathit{walking}$: $\mathit{alive} \rightarrow [\mathit{tease}]\mathit{walking}$
- weaken $\langle \mathit{tease} \rangle \top$: $\mathit{alive} \rightarrow \langle \mathit{tease} \rangle \top$
  - contraction of action theories (addressed later)

No Implicit Static Laws

Example (cont.)

Alternatives for repairing:

- $\mathcal{S} := \mathcal{S} \cup \{alive\}$
- add $tease \rightsquigarrow alive$
- weaken $[tease]walking$: $alive \rightarrow [tease]walking$
- weaken $\langle tease \rangle \top$: $alive \rightarrow \langle tease \rangle \top$
  - contraction of action theories (addressed later)

No Implicit Static Laws

Example (cont.)

Alternatives for repairing:

- $\mathcal{S} := \mathcal{S} \cup \{alive\}$
- add $tease \rightsquigarrow alive$
- weaken $[tease]walking$: $alive \rightarrow [tease]walking$
- weaken $\langle tease \rangle \top$: $alive \rightarrow \langle tease \rangle \top$
  - contraction of action theories (addressed later)

No Implicit Static Laws

Theorem

$\langle \mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \rangle$ and $\rightsquigarrow$ satisfy Postulate **PS** *iff $\mathcal{S}_{imp^*} = \emptyset$.*

Theorem

Let $\mathcal{S}_{imp^*}$ be the output of Algorithm 1 on input $\langle \mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \rangle$ and $\rightsquigarrow$. Then

- *$\langle \mathcal{S} \cup \mathcal{S}_{imp^*}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \rangle$ has no implicit static law.*
- *$\mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \models_{\rightsquigarrow} \bigwedge \mathcal{S}_{imp^*}$.*

Corollary

For all $\varphi \in \mathfrak{Fml}$, $\mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \models_{\rightsquigarrow} \varphi$ iff $\mathcal{S} \cup \mathcal{S}_{imp^*} \models \varphi$.

No Implicit Static Laws

Theorem

$\langle \mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \rangle$ *and* $\rightsquigarrow$ *satisfy Postulate* **PS** *iff* $\mathcal{S}_{imp^*} = \emptyset$.

Theorem

Let $\mathcal{S}_{imp^*}$ *be the output of Algorithm 1 on input* $\langle \mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \rangle$ *and* $\rightsquigarrow$. *Then*

- $\langle \mathcal{S} \cup \mathcal{S}_{imp^*}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \rangle$ *has no implicit static law.*
- $\mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \models_{\rightsquigarrow} \bigwedge \mathcal{S}_{imp^*}$.

Corollary

For all $\varphi \in \mathfrak{Fml}$, $\mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \models_{\rightsquigarrow} \varphi$ *iff* $\mathcal{S} \cup \mathcal{S}_{imp^*} \models \varphi$.

No Implicit Static Laws

Theorem

$\langle \mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \rangle$ and $\rightsquigarrow$ satisfy Postulate **PS** *iff $\mathcal{S}_{imp^*} = \emptyset$.*

Theorem

Let $\mathcal{S}_{imp^*}$ be the output of Algorithm 1 on input $\langle \mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \rangle$ and $\rightsquigarrow$. Then

- *$\langle \mathcal{S} \cup \mathcal{S}_{imp^*}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \rangle$ has no implicit static law.*
- *$\mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \models_{\rightsquigarrow} \bigwedge \mathcal{S}_{imp^*}$.*

Corollary

For all $\varphi \in \mathfrak{Fml}$, $\mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \models_{\rightsquigarrow} \varphi$ iff $\mathcal{S} \cup \mathcal{S}_{imp^*} \models \varphi$.

No Implicit Inexecutability Laws

Example

$$\mathcal{S} = \{walking \rightarrow alive\},\ \mathcal{E} = \left\{ \begin{array}{c} [tease]walking, \\ loaded \rightarrow [shoot]\neg alive \end{array} \right\},$$

$$\mathcal{X} = \mathcal{I} = \emptyset,\ tease \rightsquigarrow walking, shoot \rightsquigarrow \neg alive$$

- $\mathcal{S}, \mathcal{E}^{tease} \models_{\rightsquigarrow} [tease]alive$
- $\mathcal{S}, \mathcal{E}^{tease} \models_{\rightsquigarrow} \neg alive \rightarrow [tease]\neg alive$ (from $tease \not\rightsquigarrow alive$)
- Thus $\mathcal{S}, \mathcal{E}^{tease}, \mathcal{X}^{tease}, \mathcal{I}^{tease} \models_{\rightsquigarrow} \neg alive \rightarrow [tease]\bot$
- But $\mathcal{S}, \mathcal{I}^{tease} \not\models_{PDL} \neg alive \rightarrow [tease]\bot$

Postulate **PI** violated

No Implicit Inexecutability Laws

Example

$$\mathcal{S} = \{walking \rightarrow alive\},\ \mathcal{E} = \left\{ \begin{array}{c} [tease]walking, \\ loaded \rightarrow [shoot]\neg alive \end{array} \right\},$$

$$\mathcal{X} = \mathcal{I} = \emptyset,\ tease \rightsquigarrow walking, shoot \rightsquigarrow \neg alive$$

- $\mathcal{S}, \mathcal{E}^{tease} \models_{\rightsquigarrow} [tease]alive$
- $\mathcal{S}, \mathcal{E}^{tease} \models_{\rightsquigarrow} \neg alive \rightarrow [tease]\neg alive$ (from $tease \not\rightsquigarrow alive$)
- Thus $\mathcal{S}, \mathcal{E}^{tease}, \mathcal{X}^{tease}, \mathcal{I}^{tease} \models_{\rightsquigarrow} \neg alive \rightarrow [tease]\bot$
- But $\mathcal{S}, \mathcal{I}^{tease} \not\models_{PDL} \neg alive \rightarrow [tease]\bot$

Postulate **PI** violated

No Implicit Inexecutability Laws

Example

$$\mathcal{S} = \{walking \rightarrow alive\},\ \mathcal{E} = \left\{ \begin{array}{c} [tease]walking, \\ loaded \rightarrow [shoot]\neg alive \end{array} \right\},$$

$$\mathcal{X} = \mathcal{I} = \emptyset,\ tease \rightsquigarrow walking, shoot \rightsquigarrow \neg alive$$

- $\mathcal{S}, \mathcal{E}^{tease} \models_{\rightsquigarrow} [tease]alive$
- $\mathcal{S}, \mathcal{E}^{tease} \models_{\rightsquigarrow} \neg alive \rightarrow [tease]\neg alive$ (from $tease \not\rightsquigarrow alive$)
- Thus $\mathcal{S}, \mathcal{E}^{tease}, \mathcal{X}^{tease}, \mathcal{I}^{tease} \models_{\rightsquigarrow} \neg alive \rightarrow [tease]\bot$
- But $\mathcal{S}, \mathcal{I}^{tease} \not\models_{PDL} \neg alive \rightarrow [tease]\bot$

Postulate **PI** violated

No Implicit Inexecutability Laws

Example

$$\mathcal{S} = \{walking \rightarrow alive\},\ \mathcal{E} = \left\{ \begin{array}{c} [tease]walking, \\ loaded \rightarrow [shoot]\neg alive \end{array} \right\},$$

$$\mathcal{X} = \mathcal{I} = \emptyset,\ tease \rightsquigarrow walking, shoot \rightsquigarrow \neg alive$$

- $\mathcal{S}, \mathcal{E}^{tease} \models_{\rightsquigarrow} [tease]alive$
- $\mathcal{S}, \mathcal{E}^{tease} \models_{\rightsquigarrow} \neg alive \rightarrow [tease]\neg alive$ (from $tease \not\rightsquigarrow alive$)
- Thus $\mathcal{S}, \mathcal{E}^{tease}, \mathcal{X}^{tease}, \mathcal{I}^{tease} \models_{\rightsquigarrow} \neg alive \rightarrow [tease]\bot$
- But $\mathcal{S}, \mathcal{I}^{tease} \not\models_{PDL} \neg alive \rightarrow [tease]\bot$

Postulate **PI** violated

No Implicit Inexecutability Laws

Example

$$\mathcal{S} = \{walking \rightarrow alive\},\ \mathcal{E} = \left\{ \begin{array}{c} [tease]walking, \\ loaded \rightarrow [shoot]\neg alive \end{array} \right\},$$

$$\mathcal{X} = \mathcal{I} = \emptyset,\ tease \rightsquigarrow walking, shoot \rightsquigarrow \neg alive$$

- $\mathcal{S}, \mathcal{E}^{tease} \models_{\rightsquigarrow} [tease]alive$
- $\mathcal{S}, \mathcal{E}^{tease} \models_{\rightsquigarrow} \neg alive \rightarrow [tease]\neg alive$ (from $tease \not\rightsquigarrow alive$)
- Thus $\mathcal{S}, \mathcal{E}^{tease}, \mathcal{X}^{tease}, \mathcal{I}^{tease} \models_{\rightsquigarrow} \neg alive \rightarrow [tease]\bot$
- But $\mathcal{S}, \mathcal{I}^{tease} \not\models_{PDL} \neg alive \rightarrow [tease]\bot$

Postulate **PI** violated

No Implicit Inexecutability Laws

Example

$$\mathcal{S} = \{walking \rightarrow alive\},\ \mathcal{E} = \left\{ \begin{array}{c} [tease]walking, \\ loaded \rightarrow [shoot]\neg alive \end{array} \right\},$$

$$\mathcal{X} = \mathcal{I} = \emptyset,\ tease \rightsquigarrow walking, shoot \rightsquigarrow \neg alive$$

- $\mathcal{S}, \mathcal{E}^{tease} \models_{\rightsquigarrow} [tease]alive$
- $\mathcal{S}, \mathcal{E}^{tease} \models_{\rightsquigarrow} \neg alive \rightarrow [tease]\neg alive$ (from $tease \not\rightsquigarrow alive$)
- Thus $\mathcal{S}, \mathcal{E}^{tease}, \mathcal{X}^{tease}, \mathcal{I}^{tease} \models_{\rightsquigarrow} \neg alive \rightarrow [tease]\bot$
- But $\mathcal{S}, \mathcal{I}^{tease} \not\models_{PDL} \neg alive \rightarrow [tease]\bot$

Postulate **PI** violated

No Implicit Inexecutability Laws

Idea of algorithm

For each combination of effect laws

1. find inconsistent consequents
2. mark it as an implicit inexecutability

Result: the set of all implicit inexecutabilities

No Implicit Inexecutability Laws

Idea of algorithm

For each combination of effect laws

1. find inconsistent consequents
2. mark it as an implicit inexecutability

Result: the set of all implicit inexecutabilities

No Implicit Inexecutability Laws

Algorithm 2: Finding implicit inexecutability laws for a

input: $\langle \mathcal{S}, \mathcal{E}^a, \mathcal{I}^a \rangle$ and $\rightsquigarrow$

output: $\mathcal{I}^a_{imp}$, the set of implicit inexecutability laws for a

calls: $NewCons_\varphi(\psi) = Pl(\varphi \wedge \psi) \setminus Pl(\varphi)$

$\mathcal{I}^a_{imp} := \emptyset$

for all $\mathcal{E}^{a\prime} \subseteq \mathcal{E}^a$ **do**

$\varphi_{\mathcal{E}^{a\prime}} := \bigwedge\{\varphi_i : \varphi_i \rightarrow [a]\psi_i \in \mathcal{E}^{a\prime}\}$

$\psi_{\mathcal{E}^{a\prime}} := \bigwedge\{\psi_i : \varphi_i \rightarrow [a]\psi_i \in \mathcal{E}^{a\prime}\}$

for all $\chi \in NewCons_{\mathcal{S}}(\psi_{\mathcal{E}^{a\prime}})$ **do**

if $\forall l_i \in \chi,\ a \not\rightsquigarrow l_i$ **and** $\mathcal{S}, \mathcal{I}^a \nvdash (\varphi_{\mathcal{E}^{a\prime}} \wedge \neg\chi) \rightarrow [a]\bot$ **then**

$\mathcal{I}^a_{imp} := \mathcal{I}^a_{imp} \cup \{(\varphi_{\mathcal{E}^{a\prime}} \wedge \neg\chi) \rightarrow [a]\bot\}$

No Implicit Inexecutability Laws

Algorithm 2: Finding implicit inexecutability laws for a

input: $\langle \mathcal{S}, \mathcal{E}^a, \mathcal{I}^a \rangle$ and $\rightsquigarrow$

output: $\mathcal{I}^a_{imp}$, the set of implicit inexecutability laws for a

calls: $NewCons_\varphi(\psi) = Pl(\varphi \wedge \psi) \setminus Pl(\varphi)$

$\mathcal{I}^a_{imp} := \emptyset$

for all $\mathcal{E}^{a\prime} \subseteq \mathcal{E}^a$ **do**

$\varphi_{\mathcal{E}^{a\prime}} := \bigwedge\{\varphi_i : \varphi_i \rightarrow [a]\psi_i \in \mathcal{E}^{a\prime}\}$

$\psi_{\mathcal{E}^{a\prime}} := \bigwedge\{\psi_i : \varphi_i \rightarrow [a]\psi_i \in \mathcal{E}^{a\prime}\}$

for all $\chi \in NewCons_{\mathcal{S}}(\psi_{\mathcal{E}^{a\prime}})$ **do**

if $\forall l_i \in \chi,\ a \not\rightsquigarrow l_i$ **and** $\mathcal{S}, \mathcal{I}^a \nvdash (\varphi_{\mathcal{E}^{a\prime}} \wedge \neg\chi) \rightarrow [a]\bot$ **then**

$\mathcal{I}^a_{imp} := \mathcal{I}^a_{imp} \cup \{(\varphi_{\mathcal{E}^{a\prime}} \wedge \neg\chi) \rightarrow [a]\bot\}$

No Implicit Inexecutability Laws

Algorithm 2: Finding implicit inexecutability laws for a

input: $\langle \mathcal{S}, \mathcal{E}^a, \mathcal{I}^a \rangle$ and $\rightsquigarrow$

output: $\mathcal{I}^a_{imp}$, the set of implicit inexecutability laws for a

calls: $NewCons_\varphi(\psi) = Pl(\varphi \wedge \psi) \setminus Pl(\varphi)$

$\mathcal{I}^a_{imp} := \emptyset$

for all $\mathcal{E}^{a\prime} \subseteq \mathcal{E}^a$ **do**

$\varphi_{\mathcal{E}^{a\prime}} := \bigwedge\{\varphi_i : \varphi_i \rightarrow [a]\psi_i \in \mathcal{E}^{a\prime}\}$

$\psi_{\mathcal{E}^{a\prime}} := \bigwedge\{\psi_i : \varphi_i \rightarrow [a]\psi_i \in \mathcal{E}^{a\prime}\}$

for all $\chi \in NewCons_{\mathcal{S}}(\psi_{\mathcal{E}^{a\prime}})$ **do**

if $\forall l_i \in \chi,\ a \not\rightsquigarrow l_i$ **and** $\mathcal{S}, \mathcal{I}^a \nvdash (\varphi_{\mathcal{E}^{a\prime}} \wedge \neg\chi) \rightarrow [a]\bot$ **then**

$\mathcal{I}^a_{imp} := \mathcal{I}^a_{imp} \cup \{(\varphi_{\mathcal{E}^{a\prime}} \wedge \neg\chi) \rightarrow [a]\bot\}$

No Implicit Inexecutability Laws

Algorithm 2: Finding implicit inexecutability laws for a

input: $\langle \mathcal{S}, \mathcal{E}^a, \mathcal{I}^a \rangle$ and $\rightsquigarrow$

output: $\mathcal{I}^a_{imp}$, the set of implicit inexecutability laws for a

calls: $NewCons_\varphi(\psi) = Pl(\varphi \wedge \psi) \setminus Pl(\varphi)$

$\mathcal{I}^a_{imp} := \emptyset$

for all $\mathcal{E}^{a\prime} \subseteq \mathcal{E}^a$ **do**

$\varphi_{\mathcal{E}^{a\prime}} := \bigwedge\{\varphi_i : \varphi_i \rightarrow [a]\psi_i \in \mathcal{E}^{a\prime}\}$

$\psi_{\mathcal{E}^{a\prime}} := \bigwedge\{\psi_i : \varphi_i \rightarrow [a]\psi_i \in \mathcal{E}^{a\prime}\}$

for all $\chi \in NewCons_{\mathcal{S}}(\psi_{\mathcal{E}^{a\prime}})$ **do**

if $\forall l_i \in \chi,\ a \not\rightsquigarrow l_i$ **and** $\mathcal{S}, \mathcal{I}^a \nvdash (\varphi_{\mathcal{E}^{a\prime}} \wedge \neg\chi) \rightarrow [a]\bot$ **then**

$\mathcal{I}^a_{imp} := \mathcal{I}^a_{imp} \cup \{(\varphi_{\mathcal{E}^{a\prime}} \wedge \neg\chi) \rightarrow [a]\bot\}$

No Implicit Inexecutability Laws

Algorithm 2: Finding implicit inexecutability laws for a

input: $\langle \mathcal{S}, \mathcal{E}^a, \mathcal{I}^a \rangle$ and $\rightsquigarrow$

output: $\mathcal{I}^a_{imp}$, the set of implicit inexecutability laws for a

calls: $NewCons_\varphi(\psi) = Pl(\varphi \wedge \psi) \setminus Pl(\varphi)$

$\mathcal{I}^a_{imp} := \emptyset$

for all $\mathcal{E}^{a\prime} \subseteq \mathcal{E}^a$ **do**

$\varphi_{\mathcal{E}^{a\prime}} := \bigwedge\{\varphi_i : \varphi_i \rightarrow [a]\psi_i \in \mathcal{E}^{a\prime}\}$

$\psi_{\mathcal{E}^{a\prime}} := \bigwedge\{\psi_i : \varphi_i \rightarrow [a]\psi_i \in \mathcal{E}^{a\prime}\}$

for all $\chi \in NewCons_{\mathcal{S}}(\psi_{\mathcal{E}^{a\prime}})$ **do**

if $\forall l_i \in \chi,\ a \not\rightsquigarrow l_i$ **and** $\mathcal{S}, \mathcal{I}^a \nvdash (\varphi_{\mathcal{E}^{a\prime}} \wedge \neg\chi) \rightarrow [a]\bot$ **then**

$\mathcal{I}^a_{imp} := \mathcal{I}^a_{imp} \cup \{(\varphi_{\mathcal{E}^{a\prime}} \wedge \neg\chi) \rightarrow [a]\bot\}$

No Implicit Inexecutability Laws

Algorithm 2: Finding implicit inexecutability laws for a

input: $\langle \mathcal{S}, \mathcal{E}^a, \mathcal{I}^a \rangle$ and $\rightsquigarrow$

output: $\mathcal{I}^a_{imp}$, the set of implicit inexecutability laws for a

calls: $NewCons_\varphi(\psi) = Pl(\varphi \wedge \psi) \setminus Pl(\varphi)$

$\mathcal{I}^a_{imp} := \emptyset$

for all $\mathcal{E}^{a\prime} \subseteq \mathcal{E}^a$ **do**

$\varphi_{\mathcal{E}^{a\prime}} := \bigwedge\{\varphi_i : \varphi_i \rightarrow [a]\psi_i \in \mathcal{E}^{a\prime}\}$

$\psi_{\mathcal{E}^{a\prime}} := \bigwedge\{\psi_i : \varphi_i \rightarrow [a]\psi_i \in \mathcal{E}^{a\prime}\}$

for all $\chi \in NewCons_{\mathcal{S}}(\psi_{\mathcal{E}^{a\prime}})$ **do**

if $\forall l_i \in \chi,\ a \not\rightsquigarrow l_i$ **and** $\mathcal{S}, \mathcal{I}^a \not\vdash (\varphi_{\mathcal{E}^{a\prime}} \wedge \neg\chi) \rightarrow [a]\bot$ **then**

$\mathcal{I}^a_{imp} := \mathcal{I}^a_{imp} \cup \{(\varphi_{\mathcal{E}^{a\prime}} \wedge \neg\chi) \rightarrow [a]\bot\}$

No Implicit Inexecutability Laws

Algorithm 2: Finding implicit inexecutability laws for a

input: $\langle \mathcal{S}, \mathcal{E}^a, \mathcal{I}^a \rangle$ and $\rightsquigarrow$

output: $\mathcal{I}^a_{imp}$, the set of implicit inexecutability laws for a

calls: $NewCons_\varphi(\psi) = Pl(\varphi \wedge \psi) \setminus Pl(\varphi)$

$\mathcal{I}^a_{imp} := \emptyset$

for all $\mathcal{E}^{a\prime} \subseteq \mathcal{E}^a$ **do**

$\varphi_{\mathcal{E}^{a\prime}} := \bigwedge\{\varphi_i : \varphi_i \rightarrow [a]\psi_i \in \mathcal{E}^{a\prime}\}$

$\psi_{\mathcal{E}^{a\prime}} := \bigwedge\{\psi_i : \varphi_i \rightarrow [a]\psi_i \in \mathcal{E}^{a\prime}\}$

for all $\chi \in NewCons_{\mathcal{S}}(\psi_{\mathcal{E}^{a\prime}})$ **do**

if $\forall l_i \in \chi,\ a \not\rightsquigarrow l_i$ **and** $\mathcal{S}, \mathcal{I}^a \nvdash (\varphi_{\mathcal{E}^{a\prime}} \wedge \neg\chi) \rightarrow [a]\bot$ **then**

$\mathcal{I}^a_{imp} := \mathcal{I}^a_{imp} \cup \{(\varphi_{\mathcal{E}^{a\prime}} \wedge \neg\chi) \rightarrow [a]\bot\}$

No Implicit Inexecutability Laws

Example

$$\mathcal{S} = \{walking \rightarrow alive\},\ \mathcal{E} = \left\{ \begin{array}{c} [tease]walking, \\ loaded \rightarrow [shoot]\neg alive \end{array} \right\},$$

$$\mathcal{X} = \mathcal{I} = \emptyset, tease \rightsquigarrow walking, shoot \rightsquigarrow \neg alive$$

For action *tease*:

- $NewCons_{\mathcal{S}}(walking) = alive$: $[tease]alive$
- $tease \not\rightsquigarrow alive$: $\neg alive \rightarrow [tease]\neg alive$
- Then $\neg alive \rightarrow [tease]\bot$
- $\mathcal{S}, \mathcal{I}^{tease} \not\vdash_{PDL} \neg alive \rightarrow [tease]\bot$
- $\mathcal{I}^{tease}_{imp} = \{\neg alive \rightarrow [tease]\bot\}$

No Implicit Inexecutability Laws

Example

$$\mathcal{S} = \{walking \rightarrow alive\},\ \mathcal{E} = \left\{ \begin{array}{c} [tease]walking, \\ loaded \rightarrow [shoot]\neg alive \end{array} \right\},$$

$$\mathcal{X} = \mathcal{I} = \emptyset, tease \rightsquigarrow walking, shoot \rightsquigarrow \neg alive$$

For action *tease*:

- $NewCons_{\mathcal{S}}(walking) = alive$: $[tease]alive$
- $tease \not\rightsquigarrow alive$: $\neg alive \rightarrow [tease]\neg alive$
- Then $\neg alive \rightarrow [tease]\bot$
- $\mathcal{S}, \mathcal{I}^{tease} \not\vdash_{PDL} \neg alive \rightarrow [tease]\bot$
- $\mathcal{I}^{tease}_{imp} = \{\neg alive \rightarrow [tease]\bot\}$

No Implicit Inexecutability Laws

Example

$$\mathcal{S} = \{walking \rightarrow alive\},\ \mathcal{E} = \left\{ \begin{array}{c} [tease]walking, \\ loaded \rightarrow [shoot]\neg alive \end{array} \right\},$$

$$\mathcal{X} = \mathcal{I} = \emptyset, tease \rightsquigarrow walking, shoot \rightsquigarrow \neg alive$$

For action *tease*:

- $NewCons_{\mathcal{S}}(walking) = alive$: $[tease]alive$
- $tease \not\rightsquigarrow alive$: $\neg alive \rightarrow [tease]\neg alive$
- Then $\neg alive \rightarrow [tease]\bot$
- $\mathcal{S}, \mathcal{I}^{tease} \not\vdash_{PDL} \neg alive \rightarrow [tease]\bot$
- $\mathcal{I}^{tease}_{imp} = \{\neg alive \rightarrow [tease]\bot\}$

No Implicit Inexecutability Laws

Example

$$\mathcal{S} = \{walking \rightarrow alive\},\ \mathcal{E} = \left\{ \begin{array}{c} [tease]walking, \\ loaded \rightarrow [shoot]\neg alive \end{array} \right\},$$

$$\mathcal{X} = \mathcal{I} = \emptyset, tease \rightsquigarrow walking, shoot \rightsquigarrow \neg alive$$

For action *tease*:

- $NewCons_{\mathcal{S}}(walking) = alive$: $[tease]alive$
- $tease \not\rightsquigarrow alive$: $\neg alive \rightarrow [tease]\neg alive$
- Then $\neg alive \rightarrow [tease]\bot$
- $\mathcal{S}, \mathcal{I}^{tease} \not\models_{PDL} \neg alive \rightarrow [tease]\bot$
- $\mathcal{I}^{tease}_{imp} = \{\neg alive \rightarrow [tease]\bot\}$

No Implicit Inexecutability Laws

Example

$$\mathcal{S} = \{walking \rightarrow alive\},\ \mathcal{E} = \left\{ \begin{array}{c} [tease]walking, \\ loaded \rightarrow [shoot]\neg alive \end{array} \right\},$$

$$\mathcal{X} = \mathcal{I} = \emptyset, tease \rightsquigarrow walking, shoot \rightsquigarrow \neg alive$$

For action *tease*:

- $NewCons_{\mathcal{S}}(walking) = alive$: $[tease]alive$
- $tease \not\rightsquigarrow alive$: $\neg alive \rightarrow [tease]\neg alive$
- Then $\neg alive \rightarrow [tease]\bot$
- $\mathcal{S}, \mathcal{I}^{tease} \not\models_{PDL} \neg alive \rightarrow [tease]\bot$
- $\mathcal{I}^{tease}_{imp} = \{\neg alive \rightarrow [tease]\bot\}$

No Implicit Inexecutability Laws

Example

$$\mathcal{S} = \{walking \rightarrow alive\},\ \mathcal{E} = \left\{ \begin{array}{c} [tease]walking, \\ loaded \rightarrow [shoot]\neg alive \end{array} \right\},$$

$$\mathcal{X} = \mathcal{I} = \emptyset, tease \rightsquigarrow walking, shoot \rightsquigarrow \neg alive$$

For action *tease*:

- $NewCons_{\mathcal{S}}(walking) = alive$: $[tease]alive$
- $tease \not\rightsquigarrow alive$: $\neg alive \rightarrow [tease]\neg alive$
- Then $\neg alive \rightarrow [tease]\bot$
- $\mathcal{S}, \mathcal{I}^{tease} \not\models_{\mathsf{PDL}} \neg alive \rightarrow [tease]\bot$
- $\mathcal{I}^{tease}_{imp} = \{\neg alive \rightarrow [tease]\bot\}$

No Implicit Inexecutability Laws

Example

$$\mathcal{S} = \{walking \rightarrow alive\},\ \mathcal{E} = \left\{ \begin{array}{c} [tease]walking, \\ loaded \rightarrow [shoot]\neg alive \end{array} \right\},$$

$$\mathcal{X} = \mathcal{I} = \emptyset, tease \rightsquigarrow walking, shoot \rightsquigarrow \neg alive$$

For action *tease*:

- $NewCons_{\mathcal{S}}(walking) = alive$: $[tease]alive$
- $tease \not\rightsquigarrow alive$: $\neg alive \rightarrow [tease]\neg alive$
- Then $\neg alive \rightarrow [tease]\bot$
- $\mathcal{S}, \mathcal{I}^{tease} \not\models_{\mathrm{PDL}} \neg alive \rightarrow [tease]\bot$
- $\mathcal{I}^{tease}_{imp} = \{\neg alive \rightarrow [tease]\bot\}$

No Implicit Inexecutability Laws

Theorem

If $\langle \mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \rangle$ *and* $\rightsquigarrow$ *satisfy Postulate* **PS**, *then* $\langle \mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \rangle$ *and* $\rightsquigarrow$ *satisfy Postulate* **PI** *iff* $\mathcal{I}_{imp} = \emptyset$.

Generalizing the Postulates

Postulate

- **PS*** (**No implicit static laws**):
  if $\mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \models_{\rightsquigarrow} \varphi$, then $\mathcal{S} \models_{\text{PDL}} \varphi$

Theorem

$\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \rangle$ and $\rightsquigarrow$ satisfy **PS*** *iff $\langle \mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \rangle$ and $\rightsquigarrow$ satisfy* **PS** *for all $a \in \mathfrak{Act}$.*

Generalizing the Postulates

Postulate

- **PS* (No implicit static laws)**:
  if $\mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \models_{\rightsquigarrow} \varphi$, then $\mathcal{S} \models_{\text{PDL}} \varphi$

Theorem

$\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \rangle$ and $\rightsquigarrow$ satisfy **PS*** *iff $\langle \mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \rangle$ and $\rightsquigarrow$ satisfy* **PS** *for all $a \in \mathfrak{Act}$.*

Generalizing the Postulates

Postulate

- **PC*** (**Logical consistency**): $\mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \not\models_{\rightsquigarrow} \bot$

Theorem

If $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \rangle$ *and* $\rightsquigarrow$ *satisfy* **PS***, *then* $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \rangle$ *and* $\rightsquigarrow$ *satisfy* **PC*** *iff* $\langle \mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \rangle$ *and* $\rightsquigarrow$ *satisfies* **PC** *for all* $a \in \mathfrak{Act}$.

Generalizing the Postulates

Postulate

- **PC*** (**Logical consistency**): $\mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \not\models_{\rightsquigarrow} \bot$

Theorem

If $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \rangle$ *and* $\rightsquigarrow$ *satisfy* **PS***, *then* $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \rangle$ *and* $\rightsquigarrow$ *satisfy* **PC*** *iff* $\langle \mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \rangle$ *and* $\rightsquigarrow$ *satisfies* **PC** *for all* $a \in \mathfrak{Act}$.

Generalizing the Postulates

Postulate

- **PI*** (**No implicit inexecutability laws**):
  if $\mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \models_{\rightsquigarrow} \varphi \rightarrow [a]\bot$, then $\mathcal{S}, \mathcal{I} \models_{\mathrm{PDL}} \varphi \rightarrow [a]\bot$

Theorem

Let $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \rangle$ and $\rightsquigarrow$ satisfy **PS***. $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \rangle$ and $\rightsquigarrow$ satisfy* **PI*** *iff $\langle \mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \rangle$ and $\rightsquigarrow$ satisfy* **PI** *for all $a \in \mathfrak{Act}$.*

Generalizing the Postulates

Postulate

- **PI*** (**No implicit inexecutability laws**):
  if $\mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \models_{\rightsquigarrow} \varphi \rightarrow [a]\bot$, then $\mathcal{S}, \mathcal{I} \models_{\mathsf{PDL}} \varphi \rightarrow [a]\bot$

Theorem

Let $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \rangle$ *and* $\rightsquigarrow$ *satisfy* **PS***. $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \rangle$ *and* $\rightsquigarrow$ *satisfy* **PI*** *iff* $\langle \mathcal{S}, \mathcal{E}^a, \mathcal{X}^a, \mathcal{I}^a \rangle$ *and* $\rightsquigarrow$ *satisfy* **PI** *for all* $a \in \mathfrak{Act}$.

Outline

Reasoning Modularly

If $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \rangle$ and $\sim\!\!>$ satisfy Postulate **PS***, then

Theorem

$\mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \mid\!\!\sim_{\sim\!\!>} \bot$ *iff* $\mathcal{S} \models \bot$.

Theorem

$\mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \mid\!\!\sim_{\sim\!\!>} \varphi \rightarrow [a]\psi$ *iff* $\mathcal{S}, \mathcal{E}^a, \mathcal{I}^a \mid\!\!\sim_{\sim\!\!>} \varphi \rightarrow [a]\psi$.

Reasoning Modularly

If $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \rangle$ and $\sim\!\!>$ satisfy Postulate **PS***, then

Theorem

$\mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \mid\!\!\approx_{\sim\!\!>} \bot$ *iff* $\mathcal{S} \models \bot$.

Theorem

$\mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \mid\!\!\approx_{\sim\!\!>} \varphi \rightarrow [a]\psi$ *iff* $\mathcal{S}, \mathcal{E}^a, \mathcal{I}^a \mid\!\!\approx_{\sim\!\!>} \varphi \rightarrow [a]\psi$.

Reasoning Modularly

If $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \rangle$ and $\sim\!\!>$ satisfy Postulate **PS***, then

Theorem

$\mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \mid\!\!\approx_{\sim\!\!>} \varphi \rightarrow \langle a \rangle \top$ *iff* $\mathcal{S}, \mathcal{X}^a \mid\!\!\approx_{\sim\!\!>} \varphi \rightarrow \langle a \rangle \top$.

Corollary

PX *is a consequence of* **PS**.

Theorem

If $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \rangle$ *and* $\sim\!\!>$ *satisfy Postulates* **PS*** *and* **PI***, *then* $\mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \mid\!\!\approx_{\sim\!\!>} \varphi \rightarrow [a] \bot$ *iff* $\mathcal{S}, \mathcal{I}^a \mid\!\!\approx_{\sim\!\!>} \varphi \rightarrow [a] \bot$.

Reasoning Modularly

If $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \rangle$ and $\sim\!\!>$ satisfy Postulate **PS***, then

Theorem

$\mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \mid\!\!\sim_{\sim\!\!>} \varphi \rightarrow \langle a \rangle \top$ *iff* $\mathcal{S}, \mathcal{X}^a \mid\!\!\sim_{\sim\!\!>} \varphi \rightarrow \langle a \rangle \top$.

Corollary

PX *is a consequence of* **PS**.

Theorem

If $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \rangle$ *and* $\sim\!\!>$ *satisfy Postulates* **PS*** *and* **PI***, *then* $\mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \mid\!\!\sim_{\sim\!\!>} \varphi \rightarrow [a] \bot$ *iff* $\mathcal{S}, \mathcal{I}^a \mid\!\!\sim_{\sim\!\!>} \varphi \rightarrow [a] \bot$.

Reasoning Modularly

If $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \rangle$ and $\sim\!\!>$ satisfy Postulate **PS***, then

Theorem

$\mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \mid\!\!\sim\!\!> \varphi \rightarrow \langle a \rangle \top$ *iff* $\mathcal{S}, \mathcal{X}^a \mid\!\!\sim\!\!> \varphi \rightarrow \langle a \rangle \top$.

Corollary

PX *is a consequence of* **PS**.

Theorem

If $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \rangle$ *and* $\sim\!\!>$ *satisfy Postulates* **PS*** *and* **PI***, *then* $\mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \mid\!\!\sim\!\!> \varphi \rightarrow [a] \bot$ *iff* $\mathcal{S}, \mathcal{I}^a \mid\!\!\sim\!\!> \varphi \rightarrow [a] \bot$.

Reasoning Modularly

If $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \rangle$ and $\rightsquigarrow$ satisfy Postulate **PS***, then

Theorem

$\mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \models_{\overrightarrow{\sim}} \varphi \rightarrow [a_1, \ldots, a_n]\psi$ *iff*
$\mathcal{S}, \mathcal{E}^{a_1,\ldots,a_n}, \overrightarrow{\mathcal{I}}^{a_1,\ldots,a_n} \models_{\rightsquigarrow} \varphi \rightarrow [a_1, \ldots, a_n]\psi.$

Theorem

$\mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \models_{\overrightarrow{\sim}} \varphi \rightarrow \langle a_1, \ldots, a_n \rangle \psi$ *iff*
$\mathcal{S}, \mathcal{E}^{a_1,\ldots,a_n}, \overrightarrow{\mathcal{X}}^{a_1,\ldots,a_n}, \mathcal{I}^{a_1,\ldots,a_n} \models_{\rightsquigarrow} \varphi \rightarrow \langle a_1, \ldots, a_n \rangle \psi.$

Reasoning Modularly

If $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \rangle$ and $\rightsquigarrow$ satisfy Postulate **PS***, then

Theorem

$\mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \mid\!\!\approx_{\rightsquigarrow} \varphi \rightarrow [a_1, \ldots, a_n]\psi$ *iff*
$\mathcal{S}, \mathcal{E}^{a_1,\ldots,a_n}, \vec{\mathcal{I}}^{a_1,\ldots,a_n} \mid\!\!\approx_{\rightsquigarrow} \varphi \rightarrow [a_1, \ldots, a_n]\psi.$

Theorem

$\mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \mid\!\!\approx_{\rightsquigarrow} \varphi \rightarrow \langle a_1, \ldots, a_n\rangle\psi$ *iff*
$\mathcal{S}, \mathcal{E}^{a_1,\ldots,a_n}, \vec{\mathcal{X}}^{a_1,\ldots,a_n}, \mathcal{I}^{a_1,\ldots,a_n} \mid\!\!\approx_{\rightsquigarrow} \varphi \rightarrow \langle a_1, \ldots, a_n\rangle\psi.$

Outline

Another Tale

Example

- If the switch is up, the room is lit up
  - $up \rightarrow light$
- Toggling the switch changes its position
  - $\neg up \rightarrow [toggle]up$
  - $up \rightarrow [toggle]\neg up$
- It is always possible to toggle the switch
  - $\langle toggle \rangle \top$

The Need for Theory Change

You observe that...

- even if the switch is up the light is off.
- in a blackout, you do not succeed to switch the light on.
- despite your efforts you do not manage to toggle the switch.

Contraction: Motivation

Contracting by a static law

- You observe that even if the switch is up the light is off
- Static law $up \rightarrow light$ must be given up
- Can we just contract the static laws of $\mathcal{S}$?
  - May not be enough: side effects!
    - Conflict with $\mathcal{X}$
    - The contracted law may be an implicit one

Contraction: Motivation

Contracting by a static law

- You observe that even if the switch is up the light is off
- Static law $up \rightarrow light$ must be given up
- Can we just contract the static laws of $\mathcal{S}$?
  - May not be enough: side effects!
    - Conflict with $\mathcal{X}$
    - The contracted law may be an implicit one

Contraction: Motivation

Contracting by an effect law

- During a blackout you do not succeed to switch the light on
- Effect law $\neg up \rightarrow [toggle]light$ must be given up
- Important issue: give up as few as possible

Contraction: Motivation

Contracting by an effect law

- During a blackout you do not succeed to switch the light on
- Effect law $\neg up \rightarrow [toggle]light$ must be given up
- Important issue: give up as few as possible

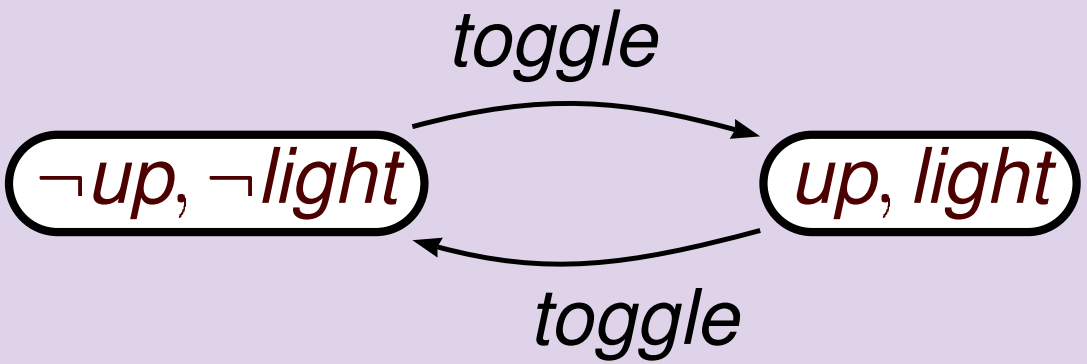
Contraction: Motivation

Contracting by an executability law

- Despite your efforts you do not manage to toggle the switch
- Executability law $up \rightarrow \langle toggle \rangle \top$ must be given up
- Side effects?

Contraction: Semantics

Playing with models

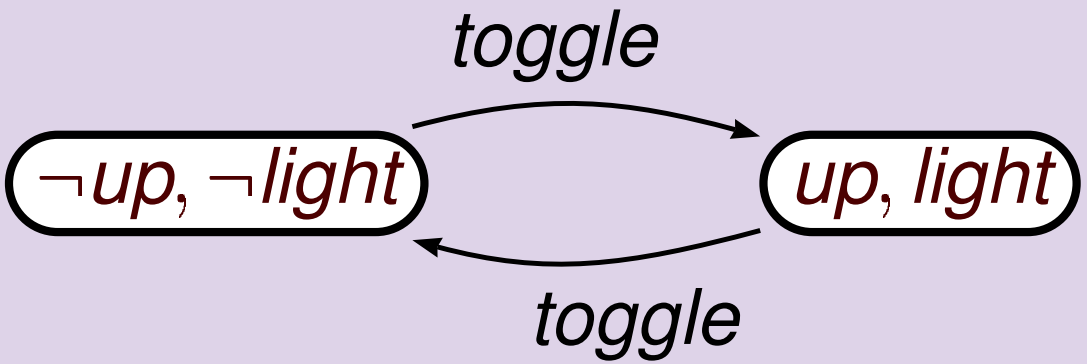


- Semantical contraction produces a set of models:

$$\langle W, R\rangle^{-}_{\Phi} = \{\langle W', R'\rangle \ : \ \ldots\}$$

Contraction: Semantics

Playing with models



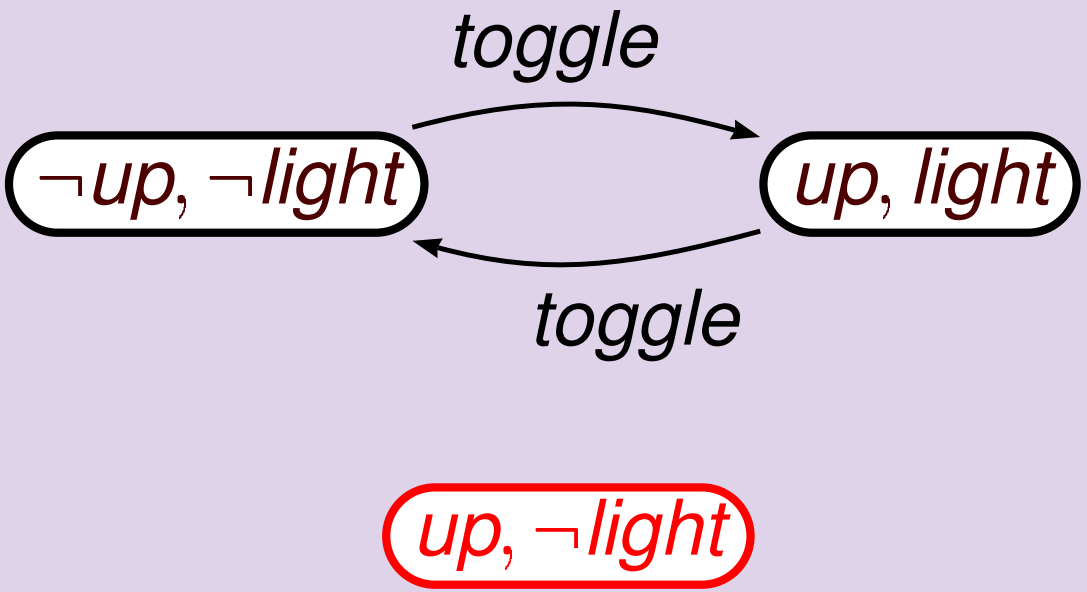
- Semantical contraction produces a set of models:

$$\langle W, R\rangle^{-}_{\Phi} = \{\langle W', R'\rangle \ : \ \ldots\}$$

Contraction: Semantics

Contracting static laws

- $\langle W, R \rangle^{-}_{up \to light} = ?$
- Intuition: add some $up \wedge \neg light$-worlds to W

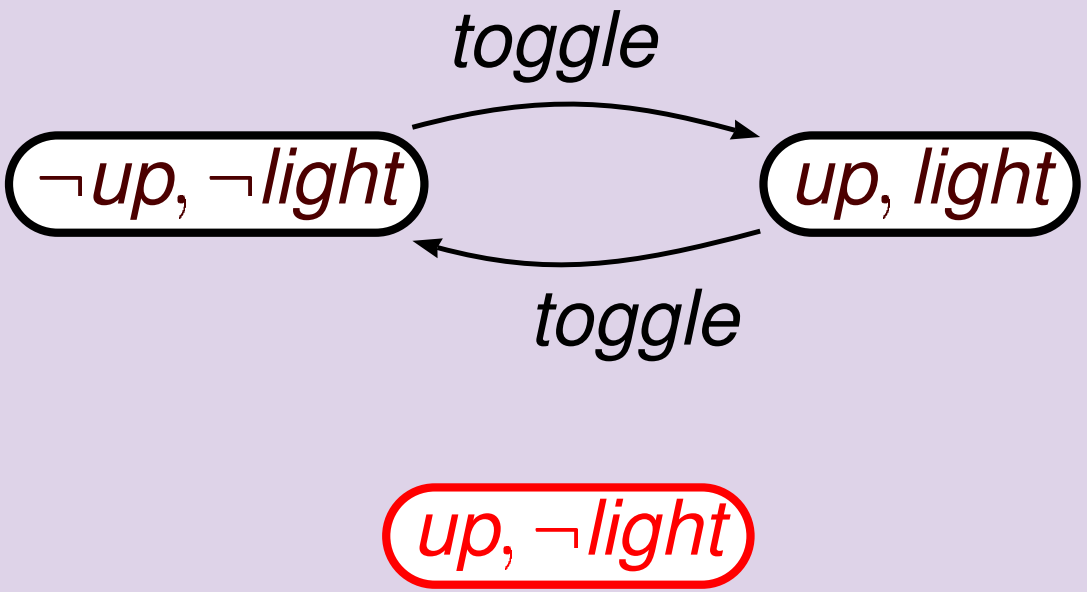


- Don't add new arrows to R!

Contraction: Semantics

Contracting static laws

- $\langle W, R \rangle^{-}_{up \to light} = ?$
- Intuition: add some $up \wedge \neg light$-worlds to W



- Don't add new arrows to R!

Contraction: Semantics

Contracting static laws

- Rely on any belief change operator for classical logic
  - Say PMA, ...
- $\langle W, R\rangle^{-}_{up \to light} = \{\langle W', R'\rangle\}$, where
  - $W' = W -_{PMA} up \to light$
  - $R' = R$
- N.B.: $\langle W', R'\rangle \not\models \langle toggle\rangle \top$
  - Executability laws to be weakened!

Contraction: Semantics

Contracting static laws

- Rely on any belief change operator for classical logic
  - Say PMA, ...
- $\langle W, R\rangle^{-}_{up \to light} = \{\langle W', R'\rangle\}$, where
  - $W' = W -_{PMA} up \to light$
  - $R' = R$
- N.B.: $\langle W', R'\rangle \not\models \langle toggle\rangle \top$
  - Executability laws to be weakened!

Contraction: Semantics

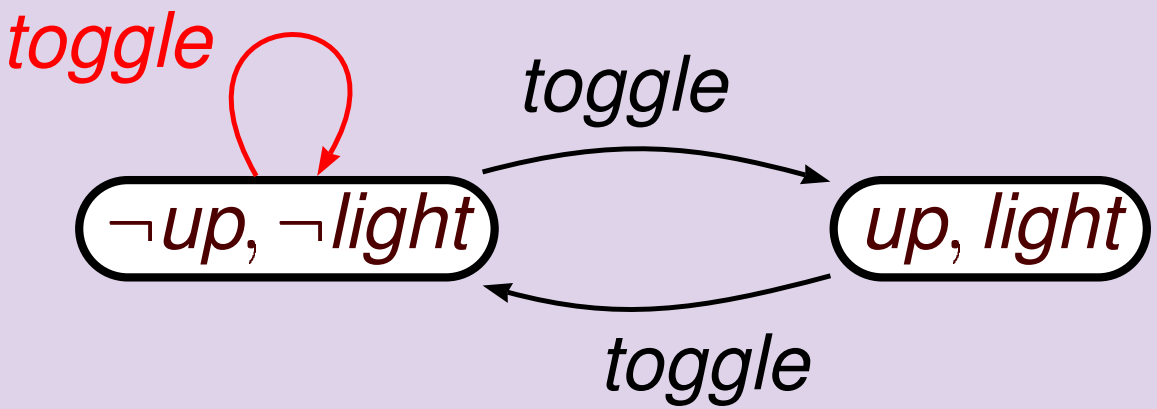
Contracting static laws

- Rely on any belief change operator for classical logic
  - Say PMA, ...
- $\langle W, R\rangle^{-}_{up \rightarrow light} = \{\langle W', R'\rangle\}$, where
  - $W' = W -_{PMA} up \rightarrow light$
  - $R' = R$
- N.B.: $\langle W', R'\rangle \not\models \langle toggle\rangle \top$
  - Executability laws to be weakened!

Contraction: Semantics

Contracting effect laws

- $\langle W, R \rangle^{-}_{\neg up \rightarrow [toggle] light} = ?$
- Intuition: add some arrows from $\neg up$-worlds to $\neg light$-worlds



Contraction: Semantics

Contracting effect laws

- $\langle W, R \rangle^{-}_{\neg up \rightarrow [toggle] light} =$

$$\{\langle W, R \cup R'_a \rangle \ : \ R'_a \subseteq \{(w, w') \ : \ w \models \neg up\}\}$$

- Problems:
  - Don't link *light*-worlds
  - Don't link all $\neg$*light*-worlds

Contraction: Semantics

Contracting effect laws

- $\langle W, R \rangle^{-}_{\neg up \rightarrow [toggle] light} =$

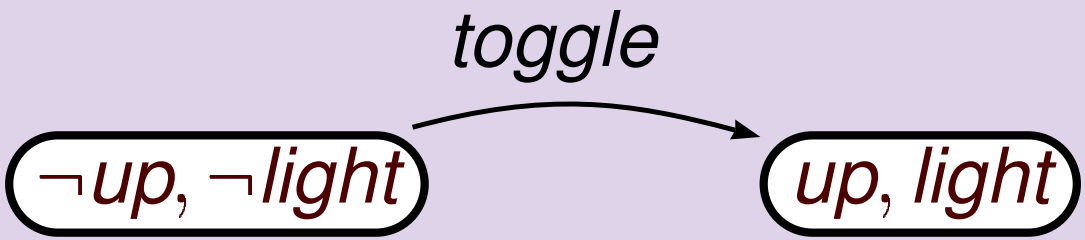
$$\{\langle W, R \cup R'_a \rangle \ : \ R'_a \subseteq \{(w, w') \ : \ w \models \neg up\}\}$$

- Problems:
  - Don't link *light*-worlds
  - Don't link all $\neg$*light*-worlds

Contraction: Semantics

Contracting executability laws

- $\langle W, R \rangle^{-}_{up \rightarrow \langle toggle \rangle \top} = ?$
- Intuition: delete some arrows leaving *up*-worlds



Contraction: Semantics

Contracting executability laws

- $\langle W, R \rangle^{-}_{up \rightarrow \langle toggle \rangle \top} =$

$$\{\langle W, R \setminus R'_a \rangle \ : \ R'_a \subseteq \{(w, w') \ : \ wR_aw' \text{ and } w \models up\}\}$$

- N.B.: if there is no *up*-world, then contraction is not successful!

Contraction: Semantics

Contracting executability laws

- $\langle W, R \rangle^{-}_{up \rightarrow \langle toggle \rangle \top} =$

$$\{\langle W, R \setminus R'_a \rangle \ : \ R'_a \subseteq \{(w, w') \ : \ wR_aw' \text{ and } w \models up\}\}$$

- N.B.: if there is no *up*-world, then contraction is not successful!

Contraction: Syntax

Domain descriptions

- Simplification: $\langle \mathcal{S}, \mathcal{E}, \mathcal{X} \rangle$
- Resulting action theory
  - $\langle \mathcal{S}, \mathcal{E}, \mathcal{X} \rangle^{-}_{\Phi} = \langle \mathcal{S}', \mathcal{E}', \mathcal{X}' \rangle$

Contraction: Syntax

Contracting static laws

- $\langle \mathcal{S}, \mathcal{E}, \mathcal{X} \rangle^{-}_{up \to light} = \langle \mathcal{S}', \mathcal{E}, \mathcal{X}' \rangle$, where
  - $\mathcal{S}' = \mathcal{S} -_{PMA} up \to light = \{light \to up\}$
  - $\mathcal{X}' =$

$$\{((\neg up \vee light) \wedge \varphi) \to \langle toggle \rangle \top \ : \ \varphi \to \langle toggle \rangle \top \in \mathcal{X}\}$$

Contraction: Syntax

Contracting static laws

- $\langle \mathcal{S}, \mathcal{E}, \mathcal{X} \rangle^{-}_{up \to light} = \langle \mathcal{S}', \mathcal{E}, \mathcal{X}' \rangle$, where
  - $\mathcal{S}' = \mathcal{S} -_{PMA} up \to light = \{light \to up\}$
  - $\mathcal{X}' =$

$$\{((\neg up \vee light) \wedge \varphi) \to \langle toggle \rangle \top \ : \ \varphi \to \langle toggle \rangle \top \in \mathcal{X}\}$$

Contraction: Syntax

Contracting effect laws

- $\langle \mathcal{S}, \mathcal{E}, \mathcal{X} \rangle^{-}_{\neg up \rightarrow [toggle] light} = \langle \mathcal{S}, \mathcal{E}', \mathcal{X} \rangle$, where
  - $\mathcal{E}' = \{(up \wedge \varphi) \rightarrow [toggle]\psi \ : \ \varphi \rightarrow [toggle]\psi \in \mathcal{E}\}$

Contraction: Syntax

Contracting executability laws

- $\langle \mathcal{S}, \mathcal{E}, \mathcal{X} \rangle^{-}_{up \to \langle toggle \rangle \top} = \langle \mathcal{S}, \mathcal{E}, \mathcal{X}' \rangle$, where
  - $\mathcal{X}' = \{(\neg up \wedge \varphi) \to \langle toggle \rangle \top \ : \ \varphi \to \langle toggle \rangle \top \in \mathcal{X}\}$

Soundness

Theorem

If $\langle W, R\rangle \models \mathcal{S} \wedge \mathcal{E} \wedge \mathcal{X}$, *then* $\langle W, R\rangle^{-}_{\Phi} \models \langle \mathcal{S}, \mathcal{E}, \mathcal{X}\rangle^{-}_{\Phi}$.

Incompleteness

Example

- $\mathcal{S} = \emptyset, \mathcal{E} = \{p \rightarrow [a]\bot\}, \mathcal{X} = \{\langle a\rangle\top\}$
- Unique model: $\langle W, R\rangle = \langle\{\{\neg p\}\}, \{(\{\neg p\}, \{\neg p\})\}\rangle$
- $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}\rangle^{-}_{p\rightarrow\langle a\rangle\top} = \langle\emptyset, \{p \rightarrow [a]\bot\}, \{\neg p \rightarrow \langle a\rangle\top\}\rangle$.
- Syntactically: successful, as $\mathcal{S}, \mathcal{E}, \mathcal{X}' \nvdash_{\mathrm{PDL}} p \rightarrow \langle a\rangle\top$.
- Semantically: contraction is unsuccessful!

Incompleteness

Example

- $\mathcal{S} = \emptyset, \mathcal{E} = \{p \rightarrow [a]\bot\}, \mathcal{X} = \{\langle a\rangle\top\}$
- Unique model: $\langle W, R\rangle = \langle\{\{\neg p\}\}, \{(\{\neg p\}, \{\neg p\})\}\rangle$
- $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}\rangle^{-}_{p\rightarrow\langle a\rangle\top} = \langle\emptyset, \{p \rightarrow [a]\bot\}, \{\neg p \rightarrow \langle a\rangle\top\}\rangle$.
- Syntactically: successful, as $\mathcal{S}, \mathcal{E}, \mathcal{X}' \not\vdash_{\mathrm{PDL}} p \rightarrow \langle a\rangle\top$.
- Semantically: contraction is unsuccessful!

Incompleteness

Example

- $\mathcal{S} = \emptyset, \mathcal{E} = \{p \rightarrow [a]\bot\}, \mathcal{X} = \{\langle a\rangle\top\}$
- Unique model: $\langle W, R\rangle = \langle\{\{\neg p\}\}, \{(\{\neg p\}, \{\neg p\})\}\rangle$
- $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}\rangle^{-}_{p\rightarrow\langle a\rangle\top} = \langle\emptyset, \{p \rightarrow [a]\bot\}, \{\neg p \rightarrow \langle a\rangle\top\}\rangle$.
- Syntactically: successful, as $\mathcal{S}, \mathcal{E}, \mathcal{X}' \not\vdash_{\mathrm{PDL}} p \rightarrow \langle a\rangle\top$.
- Semantically: contraction is unsuccessful!

Incompleteness

Example

- $\mathcal{S} = \emptyset, \mathcal{E} = \{p \rightarrow [a]\bot\}, \mathcal{X} = \{\langle a\rangle\top\}$
- Unique model: $\langle W, R\rangle = \langle\{\{\neg p\}\}, \{(\{\neg p\}, \{\neg p\})\}\rangle$
- $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}\rangle^{-}_{p\rightarrow\langle a\rangle\top} = \langle\emptyset, \{p \rightarrow [a]\bot\}, \{\neg p \rightarrow \langle a\rangle\top\}\rangle$.
- Syntactically: successful, as $\mathcal{S}, \mathcal{E}, \mathcal{X}' \not\vdash_{\mathsf{PDL}} p \rightarrow \langle a\rangle\top$.
- Semantically: contraction is unsuccessful!

Incompleteness

Example

- $\mathcal{S} = \emptyset, \mathcal{E} = \{p \rightarrow [a]\bot\}, \mathcal{X} = \{\langle a\rangle\top\}$
- Unique model: $\langle W, R\rangle = \langle\{\{\neg p\}\}, \{(\{\neg p\}, \{\neg p\})\}\rangle$
- $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}\rangle^{-}_{p\rightarrow\langle a\rangle\top} = \langle\emptyset, \{p \rightarrow [a]\bot\}, \{\neg p \rightarrow \langle a\rangle\top\}\rangle$.
- Syntactically: successful, as $\mathcal{S}, \mathcal{E}, \mathcal{X}' \not\models_{\mathrm{PDL}} p \rightarrow \langle a\rangle\top$.
- Semantically: contraction is unsuccessful!

Completeness: Modularity

Theorem

If $\langle \mathcal{S}, \mathcal{E}, \mathcal{X} \rangle$ *and* $\rightsquigarrow$ *satisfy Postulate* **PS***, *then* $\langle \mathcal{S}, \mathcal{E}, \mathcal{X} \rangle^{-}_{\Phi} \models \Psi$ *iff* $\langle W, R \rangle^{-}_{\Phi} \models \Psi$, *for every* $\langle W, R \rangle$ *such that* $\langle W, R \rangle \models \mathcal{S} \wedge \mathcal{E} \wedge \mathcal{X}$.

Outlook: Semantics of Revision

Levi identity

- Revise by $glued \rightarrow [toggle]\bot$ amounts to
  1. Contract by $\neg(glued \rightarrow [toggle]\bot)$
  2. Expand by $glued \rightarrow [toggle]\bot$
- Problem: we can contract by domain laws only
  - $\neg(glued \rightarrow [toggle]\bot) \leftrightarrow (glued \wedge \langle toggle\rangle\top)$
  - $\langle \mathcal{S}, \mathcal{E}, \mathcal{X}\rangle^{-}_{glued \wedge \langle toggle\rangle\top}$ not defined
  - What is the negation of
    - an effect law?
    - an executability law?

Outlook: Semantics of Revision

Levi identity

- Revise by $\mathit{glued} \rightarrow [\mathit{toggle}]\bot$ amounts to
  1. Contract by $\neg(\mathit{glued} \rightarrow [\mathit{toggle}]\bot)$
  2. Expand by $\mathit{glued} \rightarrow [\mathit{toggle}]\bot$
- Problem: we can contract by domain laws only
  - $\neg(\mathit{glued} \rightarrow [\mathit{toggle}]\bot) \leftrightarrow (\mathit{glued} \wedge \langle \mathit{toggle} \rangle \top)$
  - $\langle \mathcal{S}, \mathcal{E}, \mathcal{X} \rangle^{-}_{\mathit{glued} \wedge \langle \mathit{toggle} \rangle \top}$ not defined
  - What is the negation of
    - an effect law?
    - an executability law?

Related Work

Modularity

- [Pirri & Reiter 1999]: deterministic actions without ramifications in Situation Calculus
- [Amir 2000]: object-oriented concepts in Situation Calculus
- [Zhang *et al.* 2002]: EPDL approach/normal form
- [Lang *et al.* 2003]: computational complexity
- [Kakas *et al.* 2005]: elaboration tolerance, concurrent actions
- [Ghilardi, Lutz & Wolter, KR'06]: uniform interpolation and conservative extensions in $\mathcal{ALC}$

Related Work

Theory change

- [Li& Pereira 1996]: motivations
- [Liberatore 2000]: meta-results
- [Eiter *et al.* 2005/06]: update in action languages

Summary

Claim

- Consistency is not enough to evaluate a domain description
- The dynamic part of an action theory should not influence the non-dynamic one (otherwise: problems)

Contribution

- Fine-grained postulates of modularity
- Algorithms to check/give hints on modularity
- Satisfaction of modularity
  - More efficient reasoning
  - Important for updating theories [Herzig *et al.* ECAI'06]
- Our results apply to every approach allowing for $\mathcal{S}$, $\mathcal{E}$, $\mathcal{X}$ and $\mathcal{I}$

Summary

Claim

- Consistency is not enough to evaluate a domain description
- The dynamic part of an action theory should not influence the non-dynamic one (otherwise: problems)

Contribution

- Fine-grained postulates of modularity
- Algorithms to check/give hints on modularity
- Satisfaction of modularity
  - More efficient reasoning
  - Important for updating theories [Herzig *et al.* ECAI'06]
- Our results apply to every approach allowing for $\mathcal{S}$, $\mathcal{E}$, $\mathcal{X}$ and $\mathcal{I}$

Summary

Contribution (cont.)

- Semantics of action theory contraction
  - Domain-independent
  - Does not require extra information (preferences/epistemic entrenchment relation/. . .)
  - Fully automatic
- Completeness result: highlights importance of modularity

Summary

Modularity is also fruitful. . .

- for theories in general [Herzig & Varzinczak AiML'04]
- in the Situation Calculus [Herzig & Varzinczak IJCAI'05]
- in Description Logics [Herzig & Varzinczak JELIA'06]
  - (See next slide)

Future work

- Fine tune contraction of effect laws
- Contract by any formulas (not just laws)
- Postulates about effect laws? about causation?

Summary

Modularity is also fruitful. . .

- for theories in general [Herzig & Varzinczak AiML'04]
- in the Situation Calculus [Herzig & Varzinczak IJCAI'05]
- in Description Logics [Herzig & Varzinczak JELIA'06]
  - (See next slide)

Future work

- Fine tune contraction of effect laws
- Contract by any formulas (not just laws)
- Postulates about effect laws? about causation?

Outlook: Modularity in Description Logics

Example

- Suppose a passport control system in an airport
- Such a system is composed of many software components
- One of them an *ontology* (knowledge base) about passengers
- All passengers must be controlled

Outlook: Modularity in Description Logics

Example (Ontology)

- A passenger has a passport
- European citizens have European passports
- Foreigners have non-European passports
- Someone with double citizenship is a foreigner and a European

Outlook: Modularity in Description Logics

Example (The ontology in DL)

- Terminology:
  - $\text{Passenger} \sqsubseteq \exists \text{passport}.\top$
  - $\text{EUcitizen} \sqsubseteq \forall \text{passport}.\text{EU}$
  - $\text{Foreigner} \sqsubseteq \forall \text{passport}.\neg\text{EU}$
  - $\text{2Citizen} \sqsubseteq \text{Foreigner} \sqcap \text{EUcitizen}$
- Assertions:
  - EU(POLAND)
  - EUcitizen(JAN)
  - passport(JAN, POLAND)

Outlook: Modularity in Description Logics

Nevertheless

$$\left\{\begin{array}{c} \mathsf{Passenger} \sqsubseteq \exists \mathsf{passport}.\top, \\ \mathsf{EUcitizen} \sqsubseteq \forall \mathsf{passport}.\mathsf{EU}, \\ \mathsf{Foreigner} \sqsubseteq \forall \mathsf{passport}.\neg\mathsf{EU}, \\ \mathsf{2Citizen} \sqsubseteq \mathsf{Foreigner} \sqcap \mathsf{EUcitizen} \end{array}\right\} \models \mathsf{2Citizen} \sqsubseteq \forall \mathsf{passport}.(\mathsf{EU} \sqcap \neg\mathsf{EU})$$

$$\models \mathsf{2Citizen} \sqsubseteq \forall \mathsf{passport}.\bot$$

$$\models \mathsf{2Citizen} \sqsubseteq \neg\mathsf{Passenger}$$

Someone with double citizenship is not a passenger

Hence...

if we have 2Citizen(BINLADEN),
this individual is not obliged to be controlled!

Outlook: Modularity in Description Logics

Nevertheless

$$\left\{\begin{array}{c} \mathsf{Passenger} \sqsubseteq \exists \mathsf{passport}.\top, \\ \mathsf{EUcitizen} \sqsubseteq \forall \mathsf{passport}.\mathsf{EU}, \\ \mathsf{Foreigner} \sqsubseteq \forall \mathsf{passport}.\neg\mathsf{EU}, \\ \mathsf{2Citizen} \sqsubseteq \mathsf{Foreigner} \sqcap \mathsf{EUcitizen} \end{array}\right\} \models \mathsf{2Citizen} \sqsubseteq \forall \mathsf{passport}.(\mathsf{EU} \sqcap \neg\mathsf{EU})$$

$$\models \mathsf{2Citizen} \sqsubseteq \forall \mathsf{passport}.\bot$$

$$\models \mathsf{2Citizen} \sqsubseteq \neg\mathsf{Passenger}$$

Someone with double citizenship is not a passenger

Hence...

if we have 2Citizen(BINLADEN),
this individual is not obliged to be controlled!

Outlook: Modularity in Description Logics

Nevertheless

$$\left\{\begin{array}{c} \mathsf{Passenger} \sqsubseteq \exists \mathsf{passport}.\top, \\ \mathsf{EUcitizen} \sqsubseteq \forall \mathsf{passport}.\mathsf{EU}, \\ \mathsf{Foreigner} \sqsubseteq \forall \mathsf{passport}.\neg\mathsf{EU}, \\ \mathsf{2Citizen} \sqsubseteq \mathsf{Foreigner} \sqcap \mathsf{EUcitizen} \end{array}\right\} \models \mathsf{2Citizen} \sqsubseteq \forall \mathsf{passport}.(\mathsf{EU} \sqcap \neg\mathsf{EU})$$

$$\models \mathsf{2Citizen} \sqsubseteq \forall \mathsf{passport}.\bot$$

$$\models \mathsf{2Citizen} \sqsubseteq \neg\mathsf{Passenger}$$

Someone with double citizenship is not a passenger

Hence...

if we have 2Citizen(BINLADEN),
this individual is not obliged to be controlled!

Outlook: Modularity in Description Logics

Nevertheless

$$\left\{\begin{array}{c} \text{Passenger} \sqsubseteq \exists \text{passport}.\top, \\ \text{EUcitizen} \sqsubseteq \forall \text{passport}.\text{EU}, \\ \text{Foreigner} \sqsubseteq \forall \text{passport}.\neg\text{EU}, \\ \text{2Citizen} \sqsubseteq \text{Foreigner} \sqcap \text{EUcitizen} \end{array}\right\} \models \text{2Citizen} \sqsubseteq \forall \text{passport}.(\text{EU} \sqcap \neg\text{EU})$$

$$\models \text{2Citizen} \sqsubseteq \forall \text{passport}.\bot$$

$$\models \text{2Citizen} \sqsubseteq \neg\text{Passenger}$$

Someone with double citizenship is not a passenger

Hence...

if we have 2Citizen(BINLADEN),
this individual is not obliged to be controlled!

Outlook: Modularity in Description Logics

Nevertheless

$$\left\{\begin{array}{c} \mathsf{Passenger} \sqsubseteq \exists \mathsf{passport}.\top, \\ \mathsf{EUcitizen} \sqsubseteq \forall \mathsf{passport}.\mathsf{EU}, \\ \mathsf{Foreigner} \sqsubseteq \forall \mathsf{passport}.\neg\mathsf{EU}, \\ \mathsf{2Citizen} \sqsubseteq \mathsf{Foreigner} \sqcap \mathsf{EUcitizen} \end{array}\right\} \begin{array}{l} \models \mathsf{2Citizen} \sqsubseteq \forall \mathsf{passport}.(\mathsf{EU} \sqcap \neg\mathsf{EU}) \\ \models \mathsf{2Citizen} \sqsubseteq \forall \mathsf{passport}.\bot \\ \models \mathsf{2Citizen} \sqsubseteq \neg\mathsf{Passenger} \end{array}$$

Someone with double citizenship is not a passenger

Hence...

if we have 2Citizen(BINLADEN),
this individual is not obliged to be controlled!

Outlook: Modularity in Description Logics

Nevertheless

$$\left\{\begin{array}{c} \mathsf{Passenger} \sqsubseteq \exists \mathsf{passport}.\top, \\ \mathsf{EUcitizen} \sqsubseteq \forall \mathsf{passport}.\mathsf{EU}, \\ \mathsf{Foreigner} \sqsubseteq \forall \mathsf{passport}.\neg\mathsf{EU}, \\ \mathsf{2Citizen} \sqsubseteq \mathsf{Foreigner} \sqcap \mathsf{EUcitizen} \end{array}\right\} \begin{array}{l} \models \mathsf{2Citizen} \sqsubseteq \forall \mathsf{passport}.(\mathsf{EU} \sqcap \neg\mathsf{EU}) \\ \models \mathsf{2Citizen} \sqsubseteq \forall \mathsf{passport}.\bot \\ \models \mathsf{2Citizen} \sqsubseteq \neg\mathsf{Passenger} \end{array}$$

Someone with double citizenship is not a passenger

Hence...

if we have 2Citizen(BINLADEN),
this individual is not obliged to be controlled!

Outlook: Modularity in Description Logics

Our results. . .

can be applied in DL, too

Thank you!

Merci beaucoup !

Danke schön!

Choukran!

¡Muchas gracias!

Muito obrigado!

Can We Ask for More?

Postulate about effects

- **PE** (**No implicit effect laws**):
  if $\mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \models_{\rightsquigarrow} \varphi \rightarrow [a]\psi$ and $\mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \not\models_{\rightsquigarrow} \varphi \rightarrow [a]\bot$,
  then $\mathcal{S}, \mathcal{E} \models_{\rightsquigarrow} \varphi \rightarrow [a]\psi$

Can We Ask for More?

Example

$$\mathcal{S} = \emptyset,\ \mathcal{E} = \left\{ \begin{array}{c} \mathit{loaded} \rightarrow [\mathit{shoot}]\neg \mathit{alive}, \\ (\neg \mathit{loaded} \wedge \mathit{alive}) \rightarrow [\mathit{shoot}]\mathit{alive} \end{array} \right\}$$

$$\mathcal{X} = \{\mathit{hasGun} \rightarrow \langle \mathit{shoot} \rangle \top\},\ \mathcal{I} = \{\neg \mathit{hasGun} \rightarrow [\mathit{shoot}]\bot\}$$

$$\mathit{shoot} \rightsquigarrow \neg \mathit{alive}$$

- $\mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \models_{\rightsquigarrow} \neg \mathit{hasGun} \vee \mathit{loaded} \rightarrow [\mathit{shoot}]\neg \mathit{alive}$
- $\mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \not\models_{\rightsquigarrow} \neg \mathit{hasGun} \vee \mathit{loaded} \rightarrow [\mathit{shoot}]\bot$
- but $\mathcal{S}, \mathcal{E} \not\models_{\rightsquigarrow} \neg \mathit{hasGun} \vee \mathit{loaded} \rightarrow [\mathit{shoot}]\neg \mathit{alive}$

Can We Ask for More?

Example

$$\mathcal{S} = \emptyset,\ \mathcal{E} = \left\{ \begin{array}{c} \mathit{loaded} \rightarrow [\mathit{shoot}]\neg \mathit{alive}, \\ (\neg \mathit{loaded} \wedge \mathit{alive}) \rightarrow [\mathit{shoot}]\mathit{alive} \end{array} \right\}$$

$$\mathcal{X} = \{\mathit{hasGun} \rightarrow \langle \mathit{shoot} \rangle \top\},\ \mathcal{I} = \{\neg \mathit{hasGun} \rightarrow [\mathit{shoot}]\bot\}$$

$$\mathit{shoot} \rightsquigarrow \neg \mathit{alive}$$

- $\mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \models_{\rightsquigarrow} \neg \mathit{hasGun} \vee \mathit{loaded} \rightarrow [\mathit{shoot}]\neg \mathit{alive}$
- $\mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \not\models_{\rightsquigarrow} \neg \mathit{hasGun} \vee \mathit{loaded} \rightarrow [\mathit{shoot}]\bot$
- but $\mathcal{S}, \mathcal{E} \not\models_{\rightsquigarrow} \neg \mathit{hasGun} \vee \mathit{loaded} \rightarrow [\mathit{shoot}]\neg \mathit{alive}$

Can We Ask for More?

Example

$$\mathcal{S} = \emptyset,\ \mathcal{E} = \left\{ \begin{array}{c} \mathit{loaded} \rightarrow [\mathit{shoot}]\neg \mathit{alive}, \\ (\neg \mathit{loaded} \wedge \mathit{alive}) \rightarrow [\mathit{shoot}]\mathit{alive} \end{array} \right\}$$

$$\mathcal{X} = \{\mathit{hasGun} \rightarrow \langle \mathit{shoot} \rangle \top\},\ \mathcal{I} = \{\neg \mathit{hasGun} \rightarrow [\mathit{shoot}]\bot\}$$

$$\mathit{shoot} \rightsquigarrow \neg \mathit{alive}$$

- $\mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \models_{\rightsquigarrow} \neg \mathit{hasGun} \vee \mathit{loaded} \rightarrow [\mathit{shoot}]\neg \mathit{alive}$
- $\mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \not\models_{\rightsquigarrow} \neg \mathit{hasGun} \vee \mathit{loaded} \rightarrow [\mathit{shoot}]\bot$
- but $\mathcal{S}, \mathcal{E} \not\models_{\rightsquigarrow} \neg \mathit{hasGun} \vee \mathit{loaded} \rightarrow [\mathit{shoot}]\neg \mathit{alive}$

Can We Ask for More?

Postulate about effects

- **P**$\bot$ (**No unattainable effects**):
  if $\mathcal{S}, \mathcal{E} \models_{\rightsquigarrow} \varphi \rightarrow [a]\psi$, then $\mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \not\models_{\rightsquigarrow} \varphi \rightarrow [a]\bot$

Can We Ask for More?

Example

$$\mathcal{S} = \emptyset,\ \mathcal{E} = \left\{ \begin{array}{c} \mathit{loaded} \rightarrow [\mathit{shoot}]\neg \mathit{alive}, \\ (\neg \mathit{loaded} \wedge \mathit{alive}) \rightarrow [\mathit{shoot}]\mathit{alive} \end{array} \right\}$$

$$\mathcal{X} = \{\mathit{hasGun} \rightarrow \langle \mathit{shoot} \rangle \top\},\ \mathcal{I} = \{\neg \mathit{hasGun} \rightarrow [\mathit{shoot}]\bot\}$$

$$\mathit{shoot} \rightsquigarrow \neg \mathit{alive}$$

- $\mathcal{E} \models_{\rightsquigarrow} (\neg \mathit{hasGun} \wedge \mathit{loaded}) \rightarrow [\mathit{shoot}]\neg \mathit{alive}$
- but $\mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \models_{\rightsquigarrow} (\neg \mathit{hasGun} \wedge \mathit{loaded}) \rightarrow [\mathit{shoot}]\bot$

Can We Ask for More?

Example

$$\mathcal{S} = \emptyset,\ \mathcal{E} = \left\{ \begin{array}{c} \mathit{loaded} \rightarrow [\mathit{shoot}]\neg \mathit{alive}, \\ (\neg \mathit{loaded} \wedge \mathit{alive}) \rightarrow [\mathit{shoot}]\mathit{alive} \end{array} \right\}$$

$$\mathcal{X} = \{\mathit{hasGun} \rightarrow \langle \mathit{shoot} \rangle \top\},\ \mathcal{I} = \{\neg \mathit{hasGun} \rightarrow [\mathit{shoot}]\bot\}$$

$$\mathit{shoot} \rightsquigarrow \neg \mathit{alive}$$

- $\mathcal{E} \models_{\rightsquigarrow} (\neg \mathit{hasGun} \wedge \mathit{loaded}) \rightarrow [\mathit{shoot}]\neg \mathit{alive}$
- but $\mathcal{S}, \mathcal{E}, \mathcal{X}, \mathcal{I} \models_{\rightsquigarrow} (\neg \mathit{hasGun} \wedge \mathit{loaded}) \rightarrow [\mathit{shoot}]\bot$