Action Languages for Modelling Norms and Institutions

Marek Sergot

Department of Computing

Imperial College, London

Basic concepts

- action
- obligation, permission (‘deontic logic’)
- conventional generation/‘counts as’
- institutionalised ‘power

Not covered in this talk

- complex normative concepts
  - duty, right, privilege, entitlement, authorisation, responsibility, authority, . . .
- defeasibility (except for some special cases)
- knowledge, belief, intention, . . .

⋮

Structure

Part I: Institutions and Institutionalised Power

Part II: Action languages: $(\mathcal{C}/\mathcal{C}+)^{++}$

Two kinds of applications

- Representation of an existing set of laws/norms.
- Specification of a new system.
  - What are the control/enforcement mechanisms ?
  - What is an ‘implementation’ of a set of norms ?

Institutions and e-Institutions

- Institution
  - regulated interactions
  - convention: ‘institutional facts’ vs. ‘brute facts’ (Searle)
  - procedures and mechanisms for creating and determining institutional facts
- e-Institution
  - artificial agents
  - computer realisation of the institution’s procedures and mechanisms
- Institutionalised Power
  - of an agent x, in institution s, to create/establish institutional fact F.

Ingredients (*Jones & Sergot 1996*)

- instititutional facts

x *owns* y	— an institutional fact
x *has-possession-of* y	— a brute fact

x *kills* y	— a brute fact
x *murders* y	— an institutional fact

- a conditional connective

$A \stackrel{s}{\Longrightarrow} B$ $\qquad$ A counts as B (in institution s)

Example

The United Nations Convention on Contracts
for the International Sale of Goods (1980)

Article 31 If the seller is not bound to deliver the goods at any other particular place, his obligation to deliver consists:

(a) if the contract of sale involves carriage of the goods, in handing the goods over to the first carrier for transmission to the buyer;

Article 15

(1) An offer becomes effective when it reaches the offeree.

(2) . . . .

‘Institutionalised Power’

Power (of an agent x, in institution s) to create/establish institutional fact F

A generalisation of (various names):

- ‘legal power’
- ‘legal capacity’
- ‘norms of competence’
- some uses of the term ‘authority’

Threefold distinction

institutionalised
power/competence
('authority')

permission
to exercise
this power

ability
to exercise
this power

Institutionalised Power

Power (of an agent x, in institution s) to create/establish institutional fact F:

$$\mathrm{Pow}_x^s\, F$$

$$\mathrm{Pow}_x^s\,(a \textit{ owns } z)$$
$$\mathrm{Pow}_x^s\ \mathrm{Pow}_y^s\,(a \textit{ owns } z)$$
$$\mathrm{Pow}_x^s\,(A \overset{s}{\Longrightarrow} B)$$

Agent x could be an 'artificial agent': a company, committee, department, Parliament, 'the state', 'the church' —possibly even 'institution s'.

Institutionalised Power

The means by which an agent exercises power is often specified:

$$\mathrm{Pow}_x^s(F;P)$$

$$\mathrm{Pow}_x^s\,F \;\stackrel{\mathrm{def}}{=}\; \exists P\,\mathrm{Pow}_x^s(F;P)$$

Example

The United Nations Convention on Contracts for the International Sale of Goods (1980)

Article 15

(1) An offer becomes effective when it reaches the offeree.

(2) An offer, even if it is irrevocable, may be withdrawn if the withdrawal reaches the offeree before or at the same time as the offer.

Article 16

(1) Until a contract is concluded an offer may be revoked if the revocation reaches the offeree before he has dispatched an acceptance.

Example

The United Nations Convention on Contracts for the International Sale of Goods (1980)

Article 63

(1) The seller may fix an additional period of time of reasonable length for performance by the buyer of his obligations.

Example

The United Nations Convention on Contracts
for the International Sale of Goods (1980)

Article 64

(1) The seller may declare the contract avoided:

(a) if . . . ; or

(b) if the buyer does not, within the additional period of time fixed by the seller in accordance with paragraph (1) of article 63, perform his obligation to pay the price . . . , or if he declares that he will not do so within the period so fixed.

A formal characterisation (*Jones & Sergot 1996*)

$\mathrm{Pow}^{s}_{x}(F; P)$ is a special case of 'counts as':

$$\mathrm{E}_x\, P \stackrel{s}{\Longrightarrow} \mathrm{E}_x\, F$$

Or possibly:

$$\mathrm{E}_x\, P \stackrel{s}{\Longrightarrow} \mathrm{E}_s\, F$$

$$\mathrm{E}_x\, F \quad \begin{cases} \text{agent } x \text{ brings it about that } F \\ \text{agent } x \text{ sees to it that } F \end{cases}$$

***A formal characterisation** (Jones & Sergot 1996)*

Another common form:

$$\mathrm{E}_x\, P \overset{s}{\Longrightarrow} \mathrm{E}_y\, P$$

- as when e.g. a secretary x signs on behalf of the boss y;
- as when e.g. a real agent x acts on behalf of an artificial entity y;
- as when e.g. a computer agent x acts on behalf of real world entity y.

For simplicity suppose . . .

Every state of affairs F of conventional significance in institution s has associated with it a set $\pi^s(F)$ of acts prescribed by institution s for the creation of F.

Agents empowered in institution s to create F do so by performing any of the prescribed acts $\pi^s(F)$.

$$\Pi_x^s\, F \quad \stackrel{\text{def}}{=} \quad \exists P (P \in \pi^s(F) \;\&\; \mathrm{E}_x\, P)$$

$\mathrm{Pow}_x^s\, F$ — agent x is empowered in s to create F

$\mathrm{Pow}_x^s\, F \;\wedge\; \Pi_x^s\, F$ — agent x exercises its power

In practical applications

We can often simplify further.

Often, especially in e-Institutions, there is a fixed and limited repertoire of possible act types.

So then we can devise simpler languages with primitive act expressions such as

$$x \ \texttt{declares} \ F$$

(for example)

Other institutional constraints

$\mathrm{D}^s\, A$ — it is recognised by institution s that A
— it is a constraint of institution s that A

'counts as' are special kinds of institutional constraints:

$$A \overset{s}{\Longrightarrow} B \;\longrightarrow\; \mathrm{D}^s(A \longrightarrow B)$$

In the logic:

$$A \stackrel{s}{\Longrightarrow} B \;\longrightarrow\; \mathrm{D}^s(A \longrightarrow B)$$

$$A \stackrel{s}{\Longrightarrow} B \;\longrightarrow\; (\mathrm{D}^s\, A \longrightarrow \mathrm{D}^s\, B)$$

$$\mathrm{E}x\; A \stackrel{s}{\Longrightarrow} \mathrm{E}x\; B \;\longrightarrow\; (\mathrm{D}^s\, \mathrm{E}x\; A \longrightarrow \mathrm{D}^s\, \mathrm{E}x\; B)$$

In the logic:

$$A \overset{s}{\Longrightarrow} B \;\longrightarrow\; \mathrm{D}^s(A \longrightarrow B)$$

$$A \overset{s}{\Longrightarrow} B \;\longrightarrow\; (\mathrm{D}^s\, A \longrightarrow \mathrm{D}^s\, B)$$

$$\mathrm{E}x\; A \overset{s}{\Longrightarrow} \mathrm{E}x\; B \;\longrightarrow\; (\mathrm{D}^s\, \mathrm{E}x\; A \longrightarrow \mathrm{D}^s\, \mathrm{E}x\; B)$$

$$\mathrm{E}x\; A \overset{s}{\Longrightarrow} \mathrm{E}x\; B \;\longrightarrow\; (\mathrm{D}^s\, \mathrm{E}x\; A \longrightarrow \mathrm{D}^s\, B)$$

In the logic:

$$A \stackrel{s}{\Longrightarrow} B \;\longrightarrow\; \mathrm{D}^{s}(A \longrightarrow B)$$

$$A \stackrel{s}{\Longrightarrow} B \;\longrightarrow\; (\mathrm{D}^{s}\, A \longrightarrow \mathrm{D}^{s}\, B)$$

$$\mathrm{E}x\; A \stackrel{s}{\Longrightarrow} \mathrm{E}x\; B \;\longrightarrow\; (\mathrm{D}^{s}\, \mathrm{E}x\; A \longrightarrow \mathrm{D}^{s}\, \mathrm{E}x\; B)$$

$$\mathrm{E}x\; A \stackrel{s}{\Longrightarrow} \mathrm{E}x\; B \;\longrightarrow\; (\mathrm{D}^{s}\, \mathrm{E}x\; A \longrightarrow \mathrm{D}^{s}\, B)$$

$$\mathrm{Pow}^{s}_{x}\, F \;\longrightarrow\; (\Pi^{s}_{x}\, F \longrightarrow \mathrm{D}^{s}\, F)$$

(exercise of power)

(Other details of the logic omitted)

In combination with deontic modalities

Several possibilities:

$$\mathrm{Pow}^{s}_{x}\,F \;\wedge\; \mathrm{D}^{s}\,\mathrm{P}\;\Pi^{s}_{x}\,F$$

$$\mathrm{Pow}^{s}_{x}\,F \;\wedge\; \mathrm{D}^{s}\,\neg\mathrm{P}\;\Pi^{s}_{x}\,F$$

$$\neg\,\mathrm{Pow}^{s}_{x}\,F \;\wedge\; \mathrm{D}^{s}\,\neg\mathrm{P}\;\Pi^{s}_{x}\,F$$

$$\neg\,\mathrm{Pow}^{s}_{x}\,F \;\wedge\; \mathrm{D}^{s}\,\mathrm{P}\;\Pi^{s}_{x}\,F$$

No temporal dimension:

Example: A conditional power (ignoring defeasibility)

$$\mathrm{D}^{S}(\, x \text{ owns } y \;\longrightarrow\; \mathrm{Pow}^{S}_{x}(z \text{ owns } y)\,)$$

But this formalism doesn't deal with change.

$$\mathrm{D}^{S}(\, \mathit{Jim} \text{ owns } \mathit{car1}\,)$$
$$\mathrm{D}^{S}\,\neg(\, x \text{ owns } \mathit{car1} \wedge y \text{ owns } \mathit{car1} \wedge x \neq y\,)$$
$$\mathrm{Pow}^{S}_{\mathit{Jim}}(\, \mathit{Frank} \text{ owns } \mathit{car1}\,)$$

But Jim cannot exercise his power !!

Jim cannot exercise his power . . .

$$\mathrm{D}^{s}(\mathit{Jim}\ \textsf{owns}\ \mathit{car1})$$
$$\mathrm{D}^{s}\,\neg(\,x\ \textsf{owns}\ \mathit{car1} \wedge y\ \textsf{owns}\ \mathit{car1} \wedge x \neq y\,)$$
$$\mathrm{Pow}^{s}_{\mathit{Jim}}(\,\mathit{Frank}\ \textsf{owns}\ \mathit{car1}\,)$$

$$\Pi^{s}_{\mathit{Jim}}(\mathit{Frank}\ \textsf{owns}\ \mathit{car1})$$

implies

$$\mathrm{D}^{s}(\mathit{Frank}\ \textsf{owns}\ \mathit{car1})$$

which contradicts

$$\mathrm{D}^{s}(\mathit{Jim}\ \textsf{owns}\ \mathit{car1})$$

For this — and other reasons — need a temporal component.

e-Institutions: Implementation aspects

Computer realisation of the institution's procedures and mechanisms

One kind of 'implementation':

$$(x, y) \textit{ in file F } \stackrel{s}{\Longrightarrow} x \textit{ owns } y$$

e-Institutions: Implementation aspects

Another kind of 'implementation':

- 'Inter-agents' (e.g., Carles Sierra et al, 'e-institutions')
- can be regarded as a special case of 'regimentation' (Jones & Sergot 1993)

$$\mathrm{P}\ \Pi^{s}_{x}\ F \ \longleftrightarrow\ \mathrm{Pow}^{s}_{x}\ F$$

$$\mathrm{P}\ \mathrm{E}_{x}\ F \ \longleftrightarrow\ \mathit{Can}\ \mathrm{E}_{x}\ F$$

- but there is a much wider range of possible relationships between 'policies' and their implementation

Some applications

- organisational modelling
  roles, responsibilities, powers, delegation
- computer security: delegation and 'authority certificates'
  (in collaboration with Babak Sadighi, SICS)
- protocols
  auctions, negotiation protocols, rules of procedure, contract formation, dispute resolution, . . .
- semantics of ACLs in multi-agent systems
- 'open' multi-agent systems
  (with Alex Artikis and Jeremy Pitt)
  'virtual enterprises' for GRID computing

Part II: The Action Language $(\mathcal{C}/\mathcal{C}+)^{++}$

The Action Language $(\mathcal{C}/\mathcal{C}+)^{++}$

The action language $\mathcal{C}/\mathcal{C}+$
(Giunchiglia, Lee, Lifschitz, McCain, Turner)

Two extensions to the language $\mathcal{C}/\mathcal{C}+$:

- $(\mathcal{C}/\mathcal{C}+)^{+}$ — act generation ('counts as')
- $(\mathcal{C}/\mathcal{C}+)^{++}$ — permitted transitions and states

Aims

Particular interests:

- the representation of norms, in particular
- protocols (auction, contract formation, negotiation, rules of procedure, bureaucratic machinery, communication, . . .)
- animated specification of (computational) societies
- ‘run time’ implementation mechanisms (eventually)

Aims

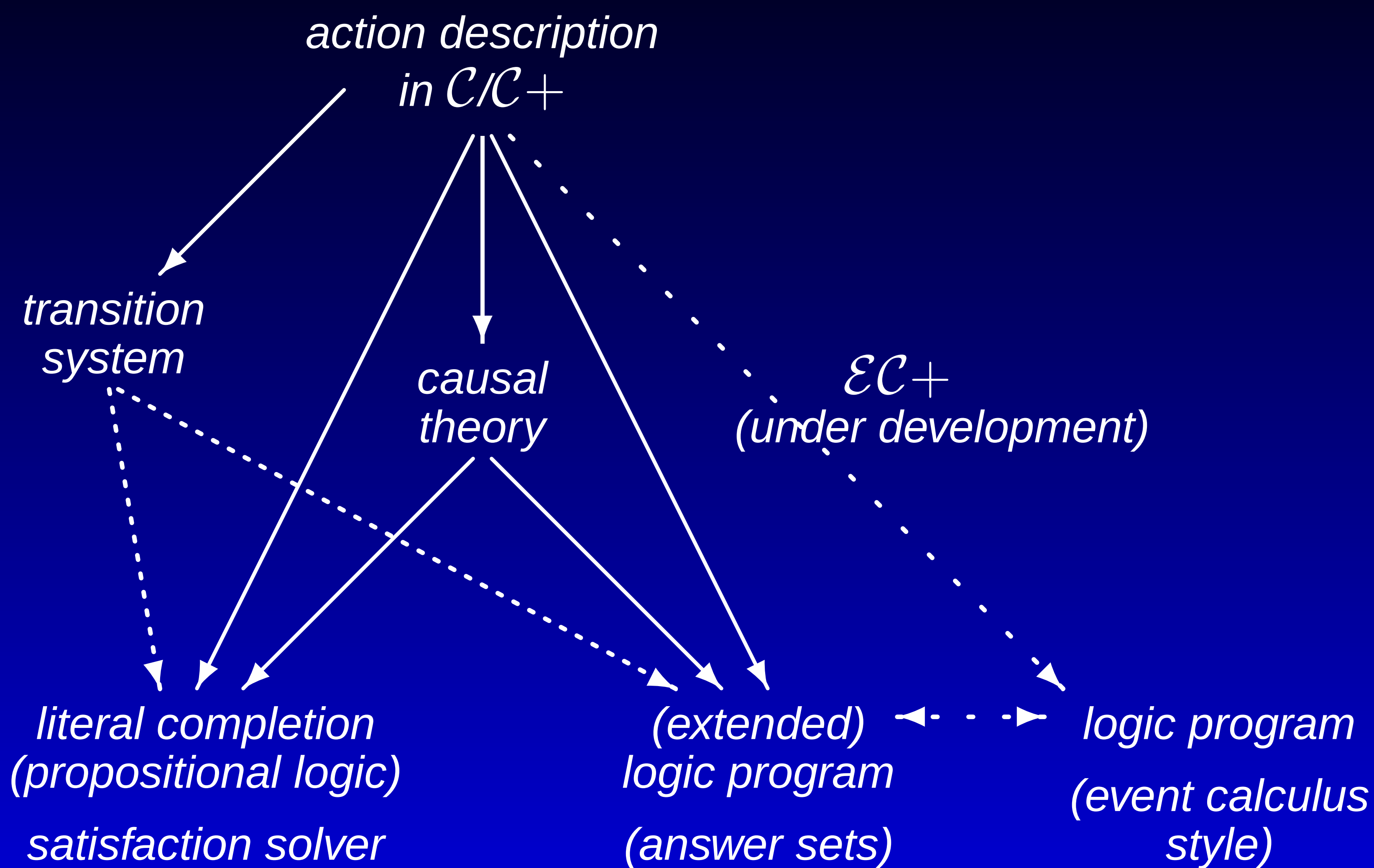
We want a representational formalism that will support:

- computational tasks
- verification (ideally)

How far can we get by building on

- transition systems?
- the language $\mathcal{C}/\mathcal{C}+$ in particular?

The language $\mathcal{C}/\mathcal{C}+$



Implementation: the Causal Calculator (CCALC) —Univ. of Texas.

The language $\mathcal{C}$/$\mathcal{C}+$

An action description in $\mathcal{C}$/$\mathcal{C}+$ is a set of $\mathcal{C}$/$\mathcal{C}+$ laws that define a transition system of a particular kind.

- — fluent constants $f{=}v,\ p,\ \neg p$
  — rigid constants
  — action constants $a{=}v,\ a,\ \neg a$

- Static laws F if G
- Fluent dynamic laws F if G after ψ
- Action dynamic laws A if ψ

- Various abbreviations

The language $\mathcal{C}$/$\mathcal{C}+$

Various abbreviations:

- A causes F if $G \;=\; F$ if $\top$ after $A \wedge G$
- nonexecutable A if $\psi \;=\; \bot$ after $A \wedge \psi$
- inertial $F \;=\; F$ if F after F

- default $F \;=\; F$ if F

Example

inertial $alive, \neg alive$
inertial $rich, \neg rich$
inertial $happy, \neg happy$

$birth$ causes $alive$
nonexecutable $birth$ if $alive$

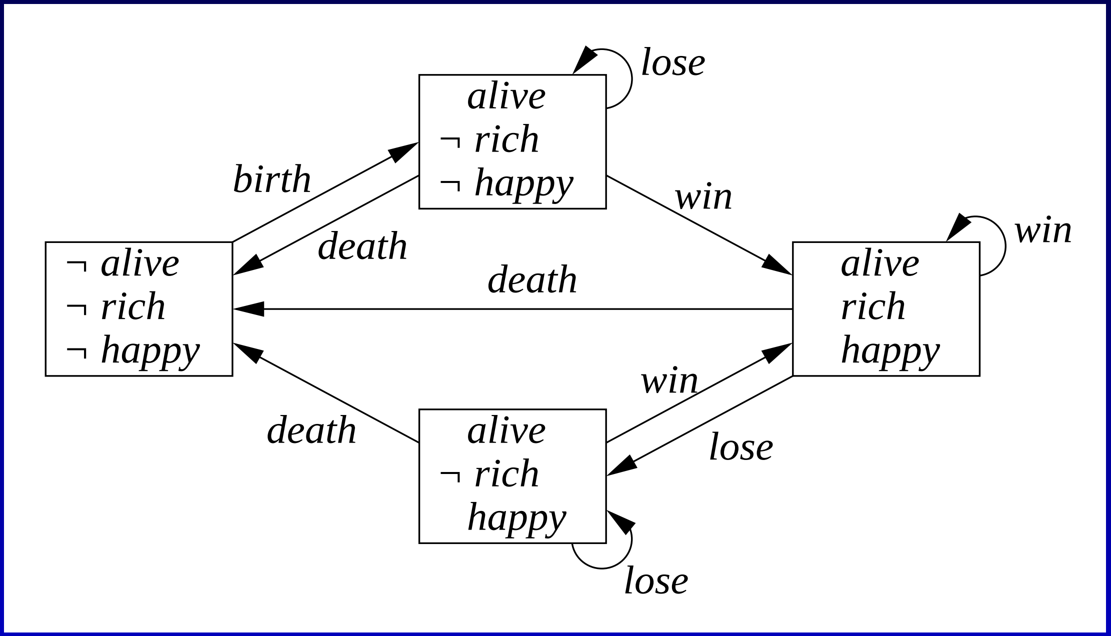
$death$ causes $\neg alive$
nonexecutable $death$ if $\neg alive$

win causes $rich$
nonexecutable win if $\neg alive$

$lose$ causes $\neg rich$
nonexecutable $lose$ if $\neg alive$

$happy$ if $rich$
$\perp$ if $rich \land \neg alive$
$\perp$ if $happy \land \neg alive$

nonexecutable $birth, death$
nonexecutable $birth, win$
nonexecutable $birth, lose$
nonexecutable $win, lose$



Semantics: states and actions

- A state is:
  - an interpretation of $\sigma^f \cup \sigma^{rigid}$ (the fluent and rigid constants) that is
  - closed under $G \rightarrow F$ for every static law F if G
- an action is
  - an interpretation of σ^a (the action constants)

Example of an action: $\{aim(x), shoot, \neg birth(y)\}$.

Semantics: transitions

$$
\begin{aligned}
\mathsf{T}_{static}(s) \quad &=_{def} \quad \{\, F \mid F \text{ if } G,\, s \models G \,\} \\
E(s, \alpha, s') \quad &=_{def} \quad \{\, F \mid F \text{ if } G \text{ after } \psi,\, s' \models G,\, s \cup \alpha \models \psi \,\}
\end{aligned}
$$

$\langle s, \alpha, s' \rangle$ is a transition iff:

- $s' \models \mathsf{T}_{static}(s')$
- $s' \models E(s, \alpha, s')$
- there is no other state s'' such that $s'' \models \mathsf{T}_{static}(s')$, $s'' \models E(s, \alpha, s')$

For a definite action description, $\langle s, \alpha, s' \rangle$ is a transition iff:

- $s' = \mathsf{T}_{static}(s') \cup E(s, \alpha, s')$

Important

$\mathcal{C}$/$\mathcal{C}+$ is a language for

- defining transition systems,

not

- a logic of transition systems.

Other languages can be interpreted on these structures:

- temporal
- epistemic (cf. ‘interpreted systems’)
- deontic (after some extension)
- narratives and planning
  - e.g. as supported by the ‘causal calculator’ CCALC

'Causal theories'

Rules of the form: $F \Leftarrow G$ (if G then F is 'caused')

For Γ a causal theory and X an interpretation of its signature:

$$\Gamma^X =_{def} \{\, F \mid F \Leftarrow G \text{ is a rule in } \Gamma \text{ and } X \models G \,\}$$

X is a model of Γ iff X is the unique (classical) model of Γ^X.

Example: defaults

$$p \Leftarrow p$$
$$\neg p \Leftarrow \ldots \textit{exceptions}$$

$\mathcal{C}/\mathcal{C}+$ action descriptions are translated to causal the

Action description D is translated to causal theory Γ_m^D

F if G	$i\!: F \Leftarrow i\!: G$
F if G after ψ	$i{+}1\!: F \Leftarrow i{+}1\!: G \wedge i\!: \psi$

Models of Γ_m^D $\overset{1-1}{\Leftrightarrow}$ paths/histories of length m in D

Translation

F if G	$i\colon F \Leftarrow i\colon G$
F if G after ψ	$i{+}1\colon F \Leftarrow i{+}1\colon G \wedge i\colon \psi$
inertial F	$i{+}1\colon F \Leftarrow i{+}1\colon F \wedge i\colon F$
A causes F if G	$i{+}1\colon F \Leftarrow i\colon A \wedge i\colon G$
never F	$\bot \Leftarrow i\colon F$
always F	$\bot \Leftarrow \neg i\colon F$
nonexecutable A if ψ	$\bot \Leftarrow i\colon A \wedge i\colon \psi$
default F if G	$i\colon F \Leftarrow i\colon F \wedge i\colon G$
A may cause F if G	$i{+}1\colon F \Leftarrow i{+}1\colon F \wedge i\colon A \wedge i\colon G$

Literal completion

For a definite causal theory Γ, translate to set of (classical) formulas $comp(\Gamma)$:

$$\left.\begin{array}{c} F \Leftarrow G_1 \\ \vdots \\ F \Leftarrow G_n \end{array}\right\} \quad \text{becomes} \quad F \leftrightarrow G_1 \vee \cdots \vee G_n$$

Models of Γ are the (classical) models of the formulas $comp(\Gamma)$.

The ‘Causal Calculator’

The ‘Causal Calculator’ CCALC

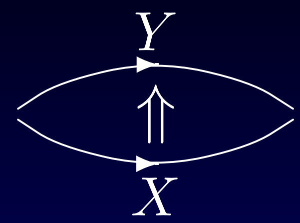
- does the translation of D to Γ_m^D,
- constructs $comp(\Gamma_m^D)$,
- invokes a standard propositional sat-solver to find (classical) models of $comp(\Gamma_m^D)$.

It provides

- a language for specifying the action signature
- a language for asserting narratives and for expressing queries.

Example: Romeo and Juliet

($\mathcal{C}$/$\mathcal{C}$+)$^{+}$: Action generation and 'counts as'



The extended transition system is:

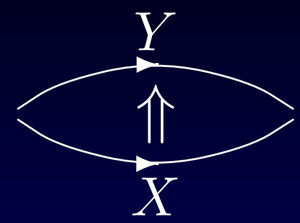
$$\langle \mathbf{F}, \mathbf{A}, \mathbf{T}, S, R, C_I, \tau \rangle$$

C_I is the counts as relation

$$C_I \subseteq S \times \mathbf{T} \times \mathbf{T} \times S$$

- $C_I(s, X, X, s')$
- if $C_I(s, X, Y, s')$ and $C_I(s, Y, Z, s')$ then $C_I(s, X, Z, s')$
- if $C_I(s, X, Y, s')$ and $\langle s, \alpha, s' \rangle \in R$ then, if $\alpha \models_\tau X$ then $\alpha \models_\tau Y$

$(\mathcal{C}/\mathcal{C}+)^{+}$: Simplification and syntax



$$\langle \mathbf{F}, \mathbf{A}, \mathbf{T}, S, R, C_I, \tau \rangle$$

C_I is the counts as relation

$$C_I \subseteq S \times \mathbf{T} \times \mathbf{T} \times S$$

$$C_I \subseteq S \times \wp(Atoms(\sigma^{\mathrm{a}})) \times Atoms(\sigma^{\mathrm{a}}) \times S$$

A special form of fluent:

$a_1, a_2, \ldots, a_m$ counts_as a'

a counts_as a'

Example

Instead of

$$birth\,(X) \text{ causes } happy\,(Y) \text{ if } parent\,(X,Y)$$
$$birth\,(X) \text{ causes } citizen\,(X) \text{ if } parent\,(X,Y),\, citizen\,(Y)$$

Separate the 'brute facts' from the 'institutional facts':

$$birth\,(X) \text{ causes } happy\,(Y) \text{ if } parent\,(X,Y)$$
$$birth\,(X) \text{ counts\_as } acquires\_cit\,(X) \text{ if } parent\,(X,Y),\, citizen\,(Y)$$
$$acquires\_cit\,(X) \text{ causes } citizen\,(X)$$

Example

Furthermore (let us say):

not-permitted $birth\,(X)$ if $parent\,(X, Y), parent\,(Z, Y), X \neq Z$

What does this imply about, e.g., $acquires\_cit\,(X)$ permitted / not-permitted?

Example: an auction protocol

Distinguish between

$$X \textit{ signals raise } (N)$$

and

$$X : \textit{raise } (N)$$

$$\text{pow}(X, A) =_{def} (X \textit{ signals } A) \text{ counts\_as } (X : A)$$

Example: an auction protocol

Alternatively (sometimes more convenient) use:

$$X : A$$

and

$$valid(X : A)$$

$$\text{pow}(X, A) =_{def} (X : A) \text{ counts\_as } valid(X : A)$$

Example: an auction protocol

pow($X, open\,(N)$) if

$\quad player\,(X), \neg withdrawn\,(X),$

$\quad max\_raise{=}Max, 0 < N \leq Max,$

$\quad current\_bid{=}none$

pow($X, raise\,(N)$) if

$\quad player\,(X), \neg withdrawn\,(X),$

$\quad max\_raise{=}Max, 0 < N \leq Max,$

$\quad \neg current\_bidder{=}X$

pow($X, withdraw$) if

$\quad player\,(X), \neg withdrawn\,(X),$

$\quad \neg current\_bidder{=}X$

$X:\ open\,(N)$ causes $current\_bidder{=}X$

$X:\ open\,(N)$ causes $current\_bid{=}N$

$X:\ raise\,(N)$ causes $current\_bidder{=}X$

$X:\ raise\,(N)$ causes $current\_bid{=}Current + N$ if

$\quad current\_bid{=}Current$

$X:\ withdraw$ causes $withdrawn\,(X)$

Example: 'Society Visualiser'

A. Artikis, M.J. Sergot, and J. Pitt.

AAMAS'02

AOSE'02 ▶

ICAIL'03

$(\mathcal{C}/\mathcal{C}+)^+$: *Implementation*

Translation to causal theories

$i\colon a_1$ counts_as $a_2 \Leftarrow i\colon a_1$ counts_as $a_2, i\colon a_2$ counts_as a_3 $\quad (i \in 0..n$

$$\bot \Leftarrow i\colon a \wedge \neg\, i\colon a' \wedge i\colon a \text{ counts\_as } a' \qquad (i \in 0..m{-}1)$$

(assuming a and a' are both ‘exogenous’ action constants)

Implementation in CCALC:

Adjustments to the front end

($\mathcal{C}$/$\mathcal{C}+$)$^+$: *Approximation by* $\mathcal{C}$/$\mathcal{C}+$

a counts_as a' approximated by nonexecutable $a \wedge \neg a'$

(assuming a and a' are both 'exogenous' action constants)

Or for more flexibility, add dynamic laws

nonexecutable $a \wedge \neg a'$ if a counts_as a'

In the latest version of $\mathcal{C}$/$\mathcal{C}+$, can add 'action dynamic laws'

a' if $a \wedge$ a counts_as a'

(which is equivalent to nonexecutable $a \wedge \neg a'$ if a counts_as a' when a and a' are both 'exogenous' action constants)

$(\mathcal{C}/\mathcal{C}+)^+_{defeasible}$

$$\frac{a_1 \text{ counts\_as } a_2 \qquad a_2 \text{ causes } F}{a_1 \text{ causes } F}$$ *implies (in effect)*

$$\frac{a_1 \text{ counts\_as } a_2 \qquad \text{nonexecutable } a_2}{\text{nonexecutable } a_1}$$ *implies (in effect)*

$(\mathcal{C}/\mathcal{C}+)^{+}{}_{defeasible}$

$$\frac{a_1 \text{ counts\_as } a_2 \qquad \text{nonexecutable } a_2}{\text{nonexecutable } a_1} \quad \textit{implies (in effect)}$$

$$\frac{raise\_hand \text{ counts\_as } make\_bid \qquad \text{nonexecutable } make\_bid}{\text{nonexecutable } raise\_hand} \quad \textit{implies (in effect)}$$

$\mathcal{(C/C+)}^{+}_{defeasible}$

$$\frac{\begin{array}{c} raise\_hand \text{ counts\_as } make\_bid \\ \text{nonexecutable } make\_bid \end{array}}{\text{nonexecutable } raise\_hand} \quad \textit{implies (in effect)}$$

Appropriate ?

— Yes, if the action description is a system specification.

— No, if the action description is a representation of some existing norm system.

For the latter case, need to adjust to make the effects of counts_as defeasible — several options. Details omitted here.

$(\mathcal{C}/\mathcal{C}+)^{++}$: Permission

An extended transition system of the form:

$$\langle \mathbf{F}, \mathbf{A}, \mathbf{T}, S, R, C_I, \tau, G_S, G_R \rangle$$

where the new components are

- $G_S \subseteq S$, the set of 'green' ('permitted','acceptable', 'ideal', 'legal') states
- $G_R \subseteq R$, the set of 'green' ('permitted','acceptable', 'ideal', 'legal') transitions

plus the constraint

If $\langle s, \alpha, s' \rangle \in G_R$ and $s \in G_S$ then $s' \in G_S$

We'll call this the green-green-green constraint.

The 'green-green-green' constraint

If $\langle s, \alpha, s' \rangle \in G_R$ and $s \in G_S$ then $s' \in G_S$

possible

not possible

possible

*possible**

*possible**

possible

*possible**

possible

*contra J-J Meyer *Dynamic Deontic Logic*

Syntax

State permission law: not-permitted F

Action permission laws: not-permitted A if F after G

Everything else is permitted by default.
Or: one can have the opposite default if desired.

Note:
($\mathcal{C}$/$\mathcal{C}+$)$^{++}$ is a language for
defining (extended) transition systems.

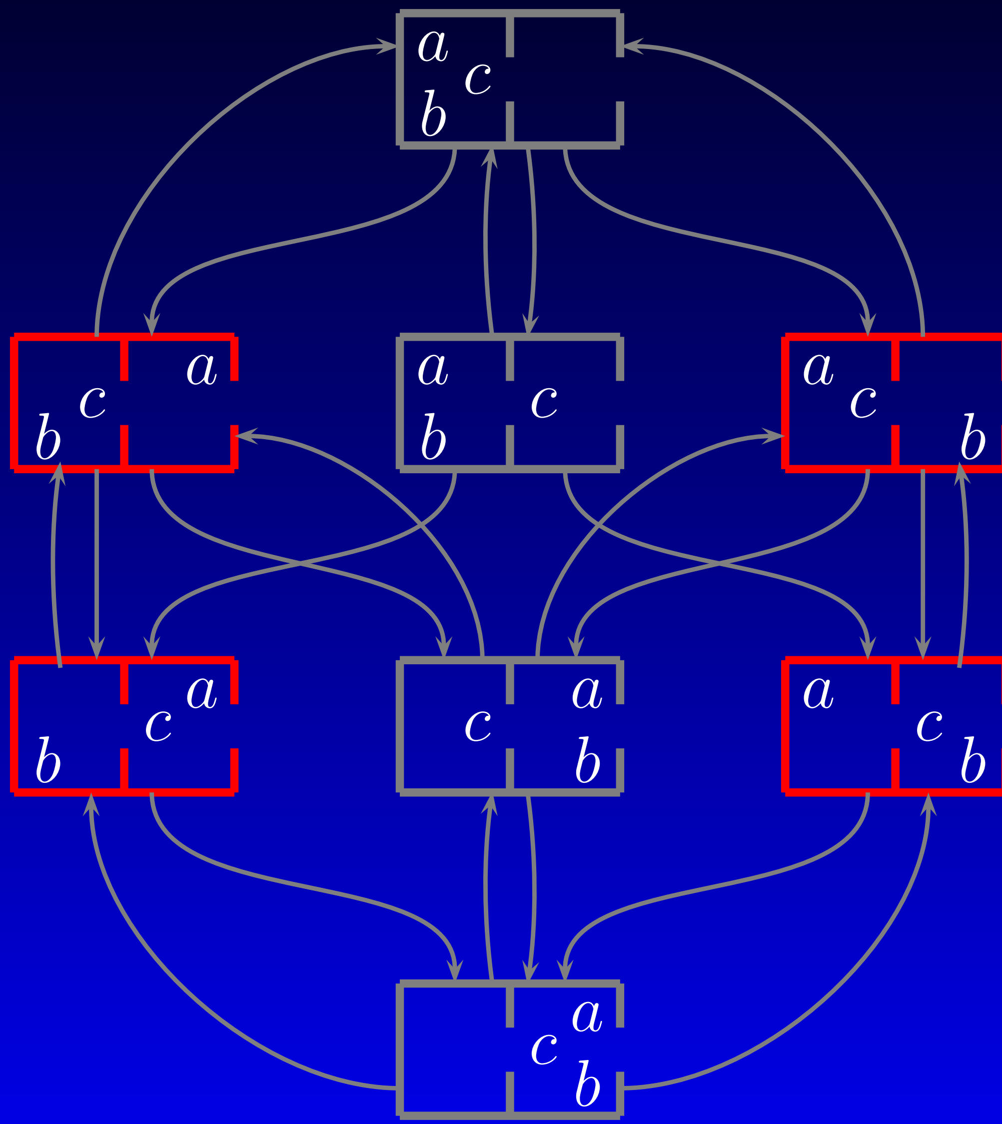
Example

It is forbidden for a man and a woman to be alone together in a room.

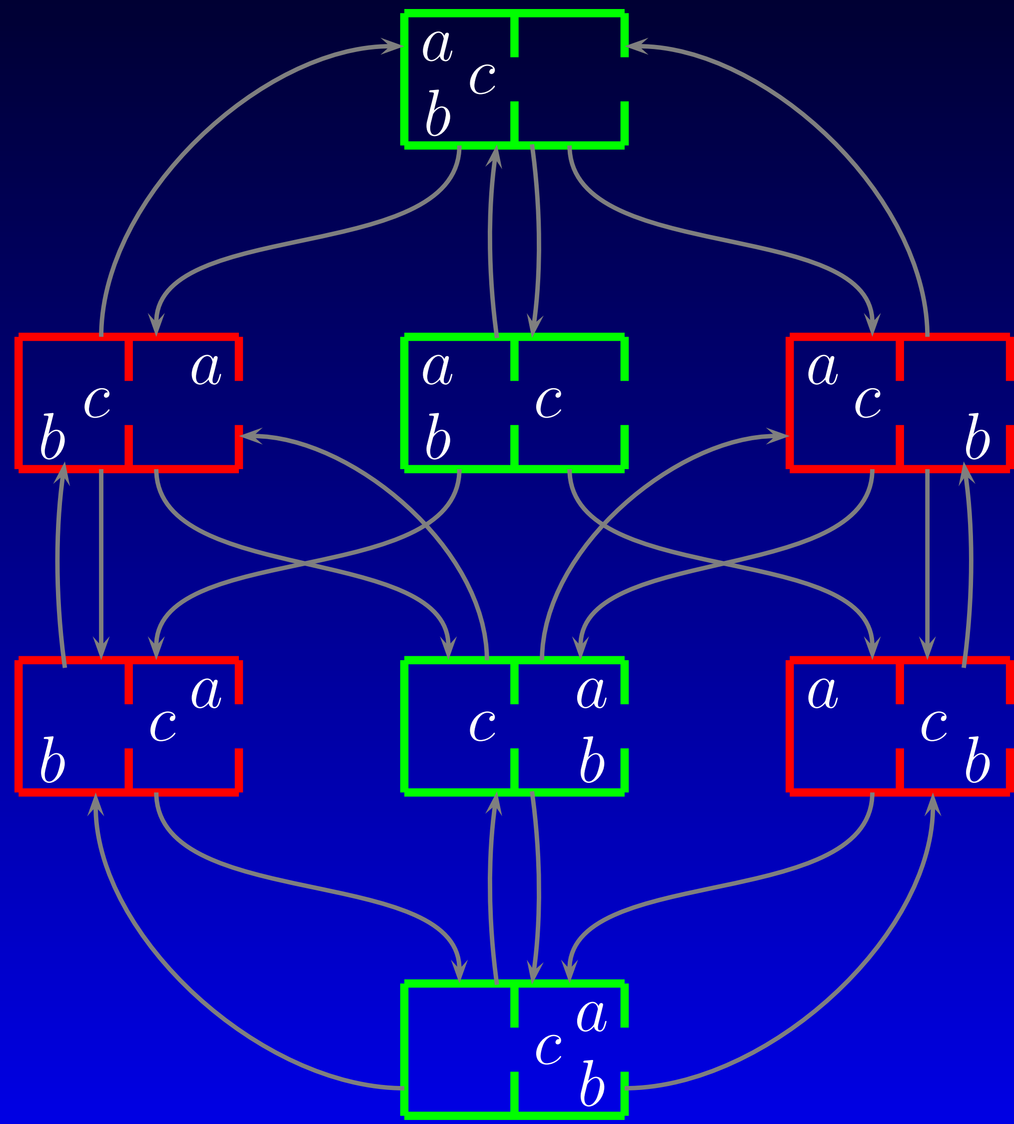
a and b are men. c is a woman.

not-permitted $loc(a){=}p \wedge loc(c){=}p \wedge \neg loc(b){=}p$
not-permitted $loc(b){=}p \wedge loc(c){=}p \wedge \neg loc(a){=}p$

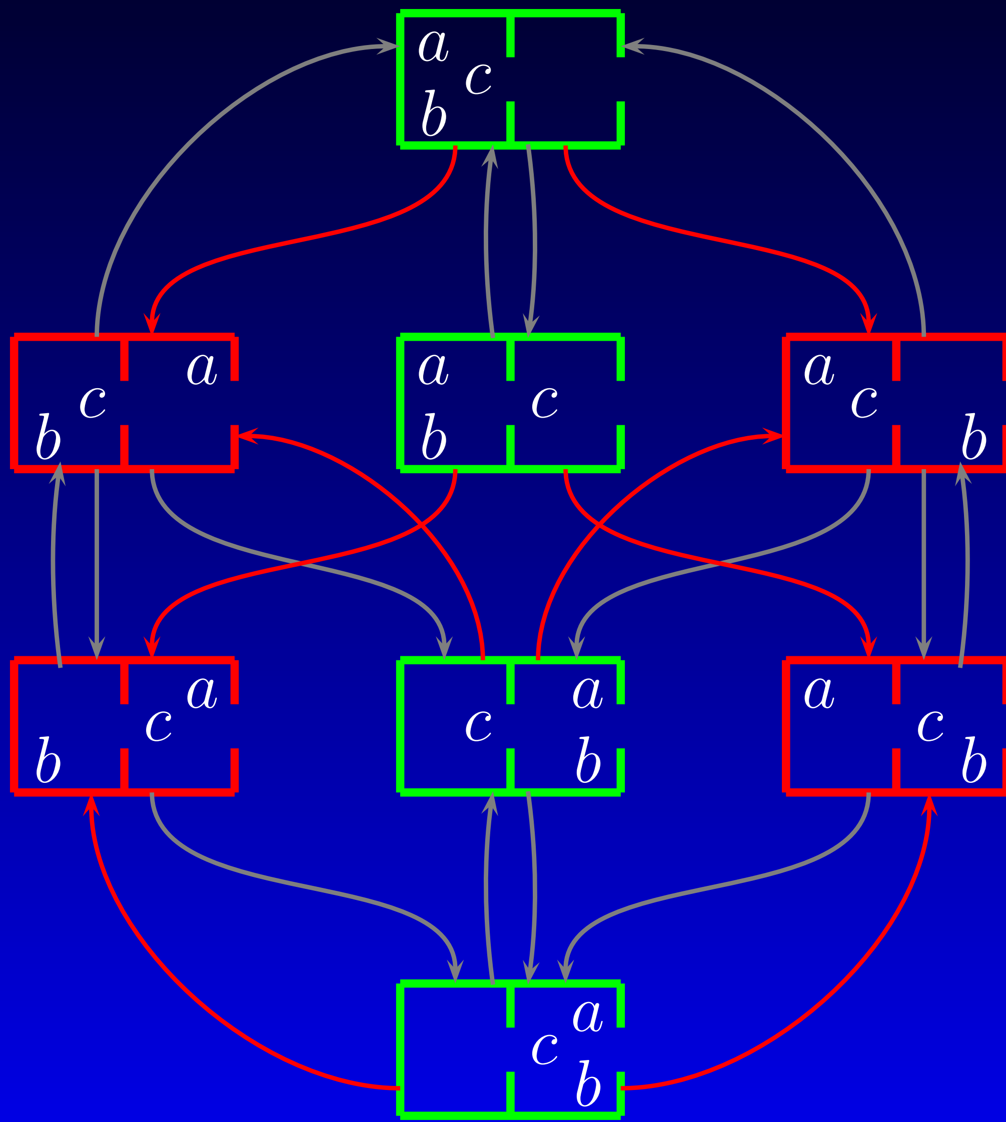
Given:



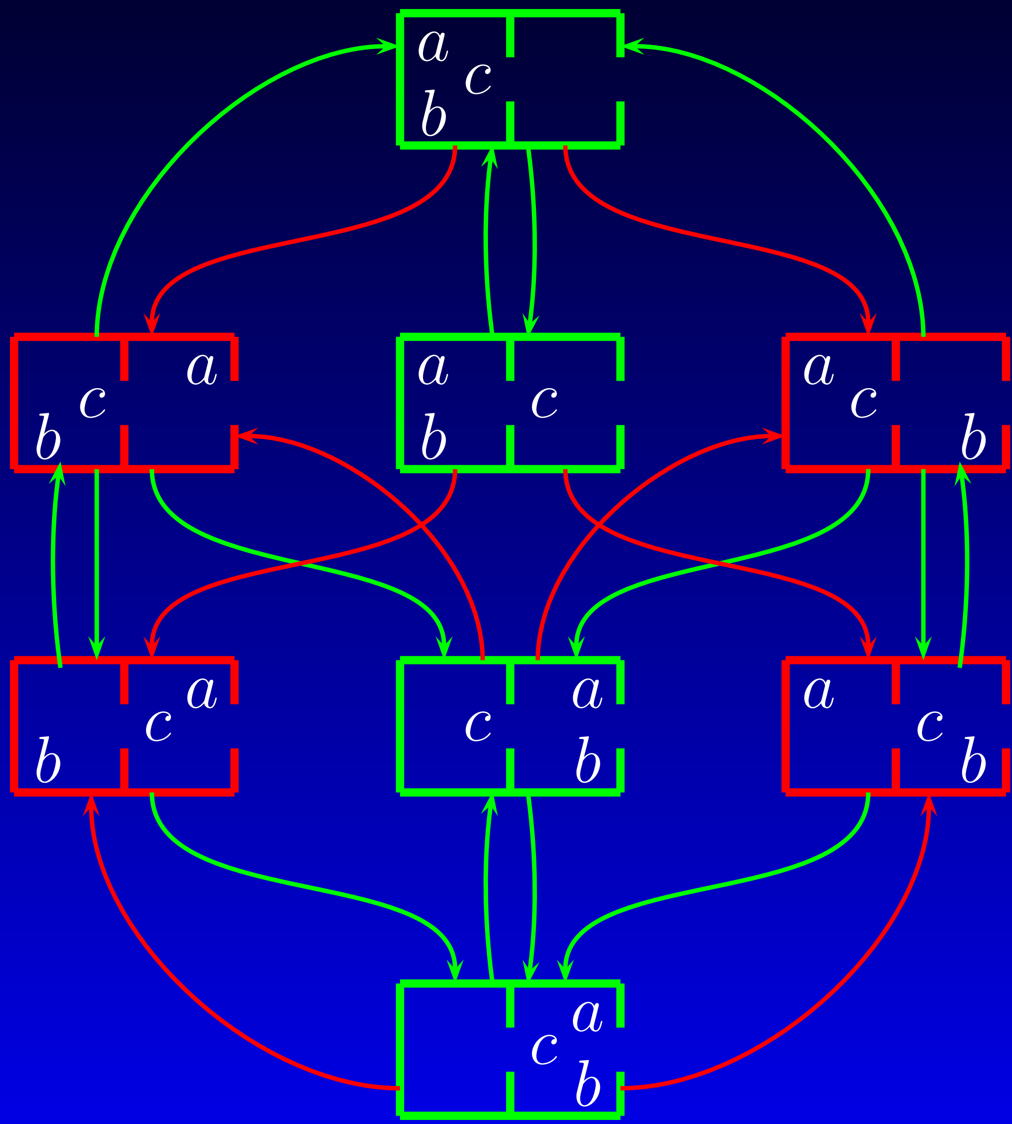
First step:



The green-green-green constraint:



Infer:



($\mathcal{C}$/$\mathcal{C}+$)$^{++}$: translation to causal theory

Special fluent symbol: status (possible values green and red)

State permission law

$$\text{not-permitted } F$$

becomes

$$i : \text{status=red} \Leftarrow i : F$$

Add

$$i : \text{status=green} \Leftarrow i : \text{status=green}$$

to get the desired default behaviour

$(\mathcal{C}/\mathcal{C}+)^{++}$: translation to causal theory

A special action symbol trans (possible values green and red)

Action permission law

$$\text{not-permitted } A \text{ if } F \text{ after } G$$

becomes

$$i\colon \text{trans}{=}\text{red} \Leftarrow i{+}1\colon F \wedge i\colon A \wedge i\colon G$$

Add

$$i\colon \text{trans}{=}\text{green} \Leftarrow i\colon \text{trans}{=}\text{green}$$

to get the desired default behaviour

$(\mathcal{C}/\mathcal{C}+)^{++}$: translation to causal theory

Finally, to capture the green-green-green constraint, add the constraint

$$i: \text{trans}{=}\text{red} \Leftarrow i: \text{status}{=}\text{green} \wedge i{+}1: \text{status}{=}\text{red}$$

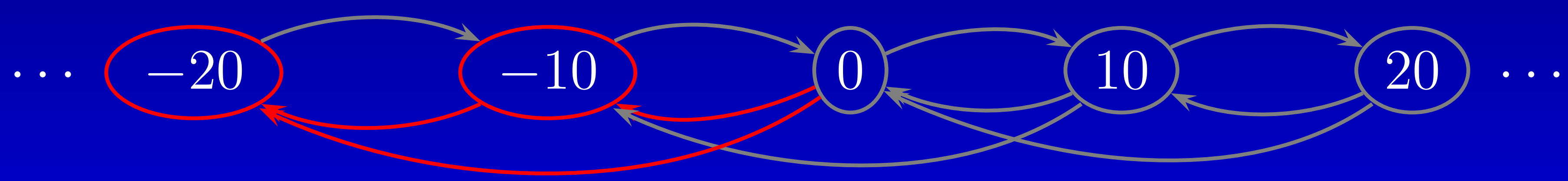
The ‘green-green-green’ constraint

Example

Three kinds of action: withdraw £10, withdraw £20, deposit £10.

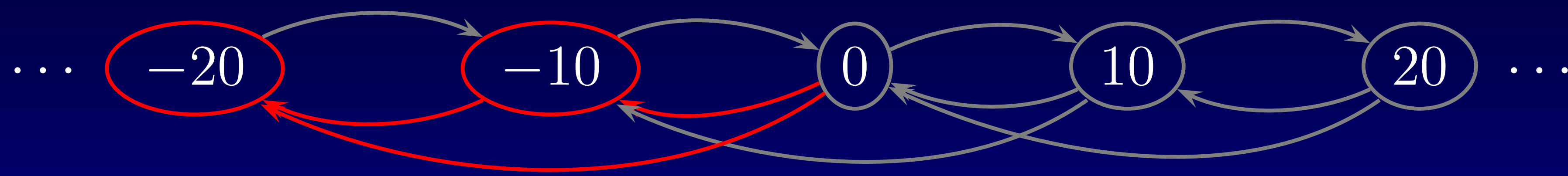
A state is forbidden (not permitted, red) if the balance is less than 0.

Suppose, for the sake of the example, that a withdrawal is forbidden (red) when the balance is zero or negative.



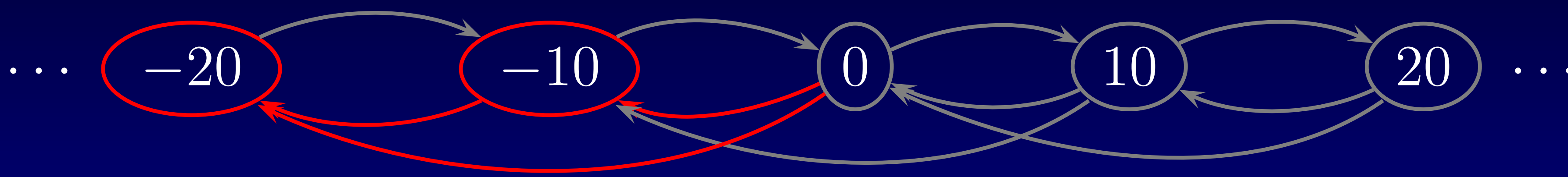
The ‘green-green-green’ constraint

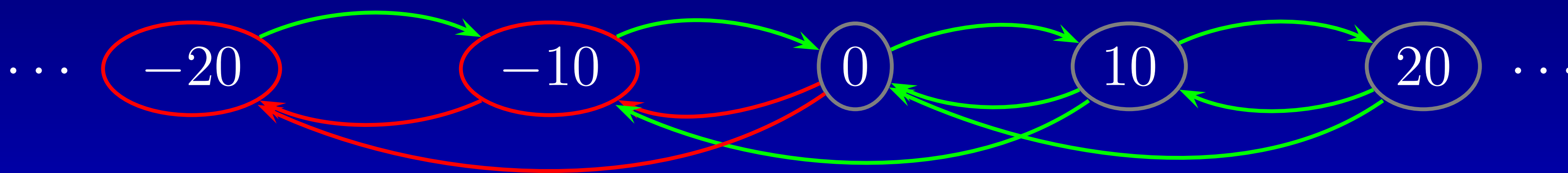
The wrong way round:



The ‘green-green-green’ constraint

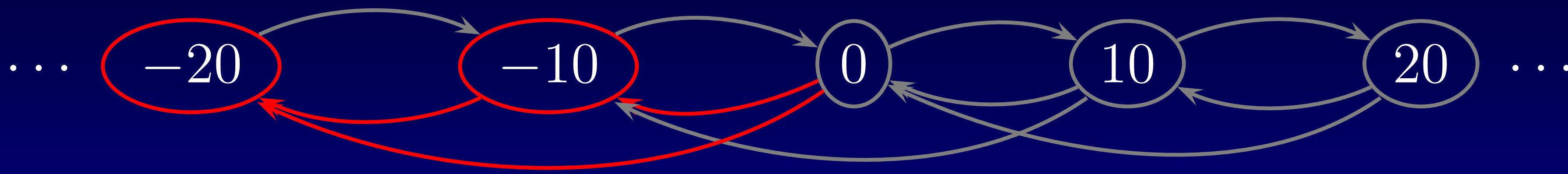
The wrong way round:

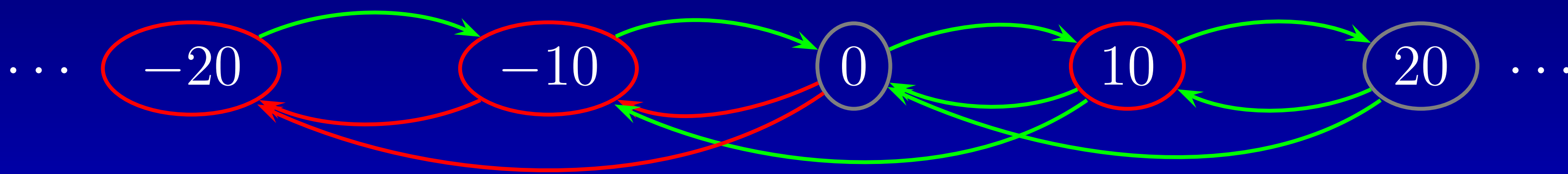




The ‘green-green-green’ constraint

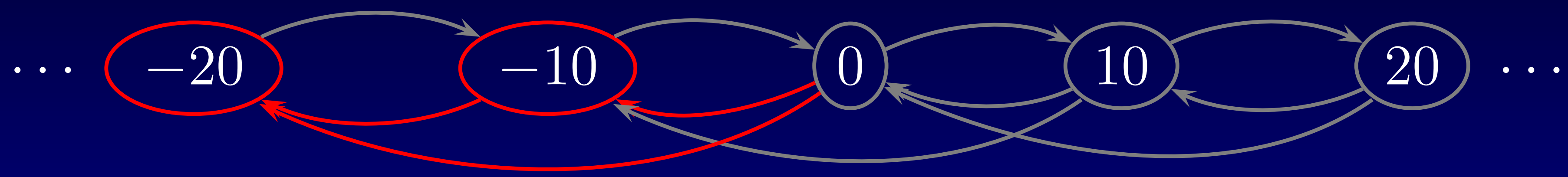
The wrong way round:

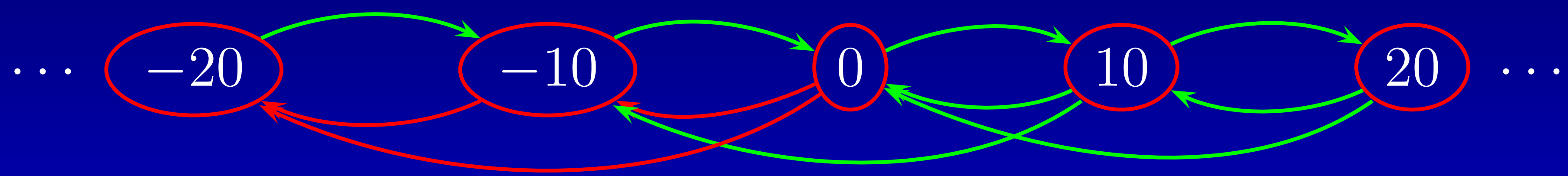




The ‘green-green-green’ constraint

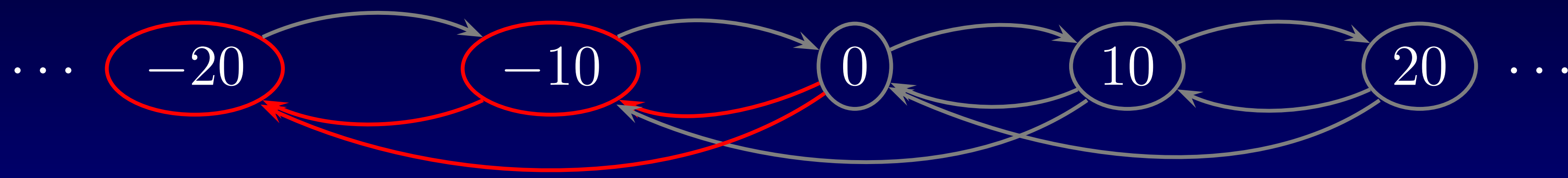
The wrong way round:





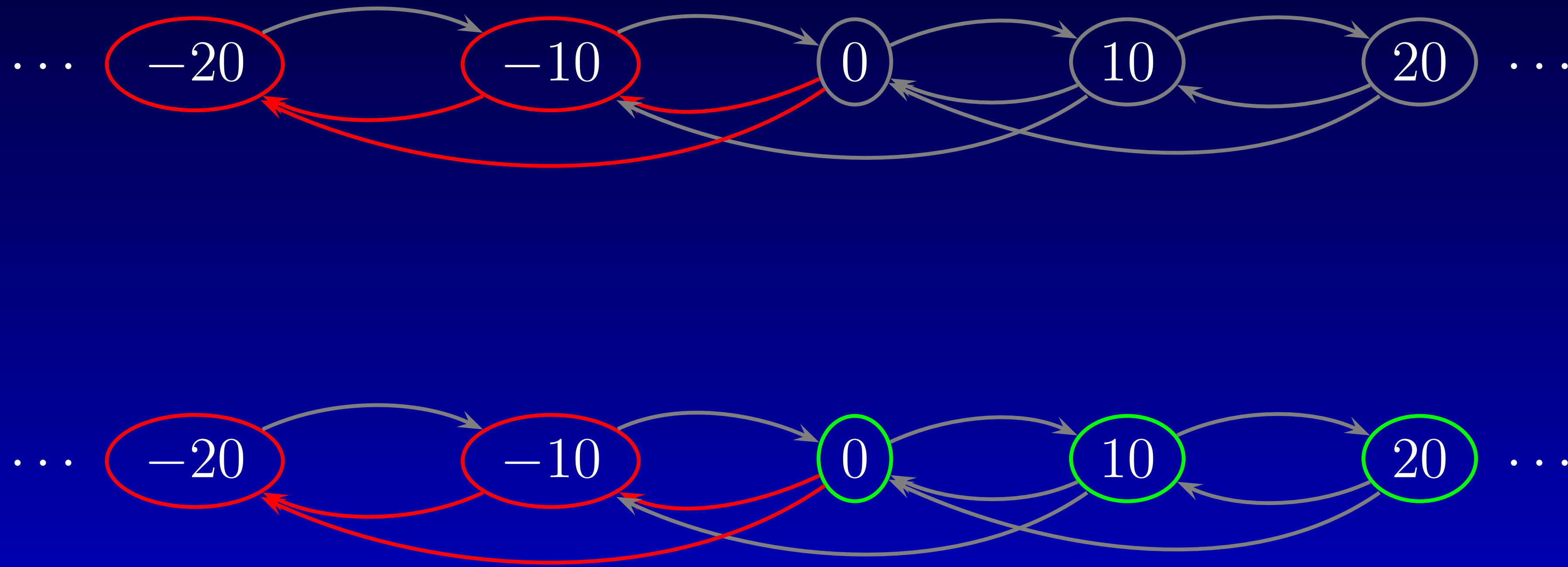
The ‘green-green-green’ constraint

The right way round:



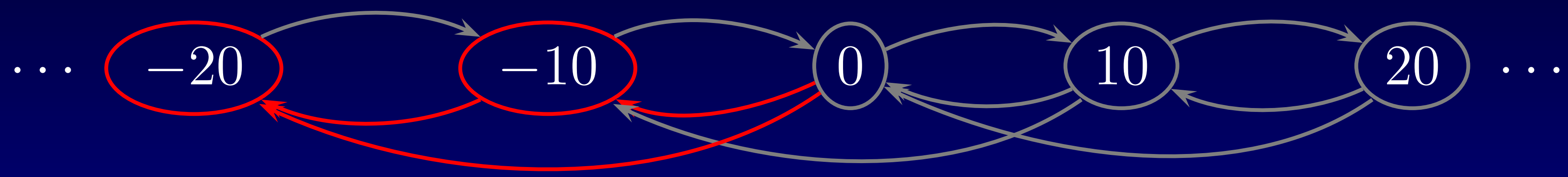
The 'green-green-green' constraint

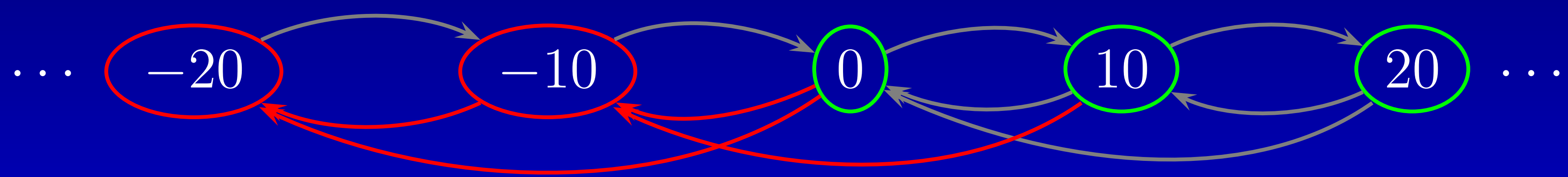
The right way round:



The ‘green-green-green’ constraint

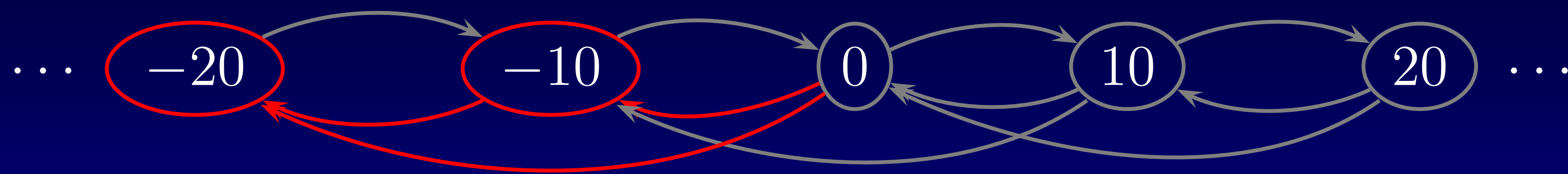
The right way round:

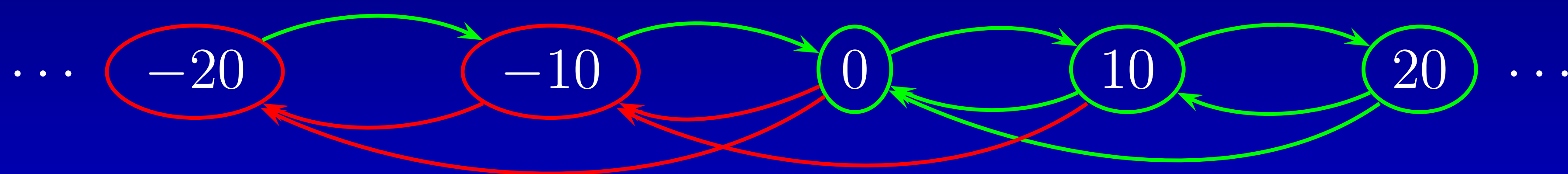




The ‘green-green-green’ constraint

The right way round:





Aims

We want a representational formalism that will support:

- computational tasks
- verification (ideally)

How far can we get by building on

- transition systems?
- the language $\mathcal{C}$/$\mathcal{C}+$ in particular?

Current work

- $(\mathcal{C}/\mathcal{C}+)^{++}$ as an input language for a model checker.
- Alternative implementation routes
  ($\mathcal{EC}+$ — 'event calculus' like computation with narratives).
- Run time architectures and implementation mechanisms.
- Further structure in states of the (extended) transition system
  - multiple 'institutions' (multiple 'counts as' relations)
  - local states for agents (and environment) for modelling multi-agent systems
- Richer structures — transition systems too restrictive.