Evolving Normative Systems: Adapt or Become Irrelevant

Marina De Vos

Department of Computer Science
University of Bath
cssmdv@bath.ac.uk

art-ai

Acknowledgments

- Andreasa Morris Martin
- Julian Padget

Related papers:

- Morris-Martin, De Vos, and Padget "Norm emergence in multiagent systems: a viewpoint paper", JAAMAS, 2019
- Morris-Martin, Vos, et al. "Agent-directed Runtime Norm Synthesis", AAMAS, 2023

History Part 1

- Modelling open multi-agent systems like human societies
  - Autonomy
  - Guidance
- $\Rightarrow$ Norms
- External to the agent
- Institutions/Normative Frameworks

History Part 2

- Institutions:
  - formal model
  - computation representation
  - InstAI + ASP
- Design time vs run-time
- Combining and interaction between institutions
- Debugging

Present

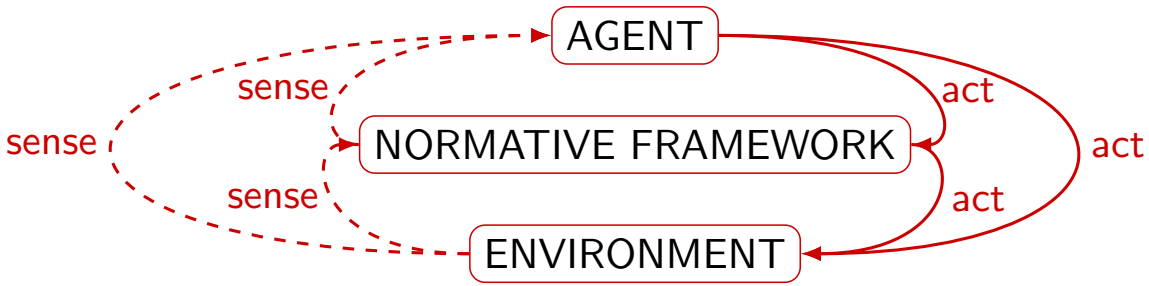
- Circumstances change
- How to adapt
- Stakeholders

What is a norm?

- What is a norm? Informal or formal constraint on action
- *Definition:* a principle of right action binding upon the members of a group and serving to guide, control, or regulate proper and acceptable behaviour [Merriam-Webster dictionary]

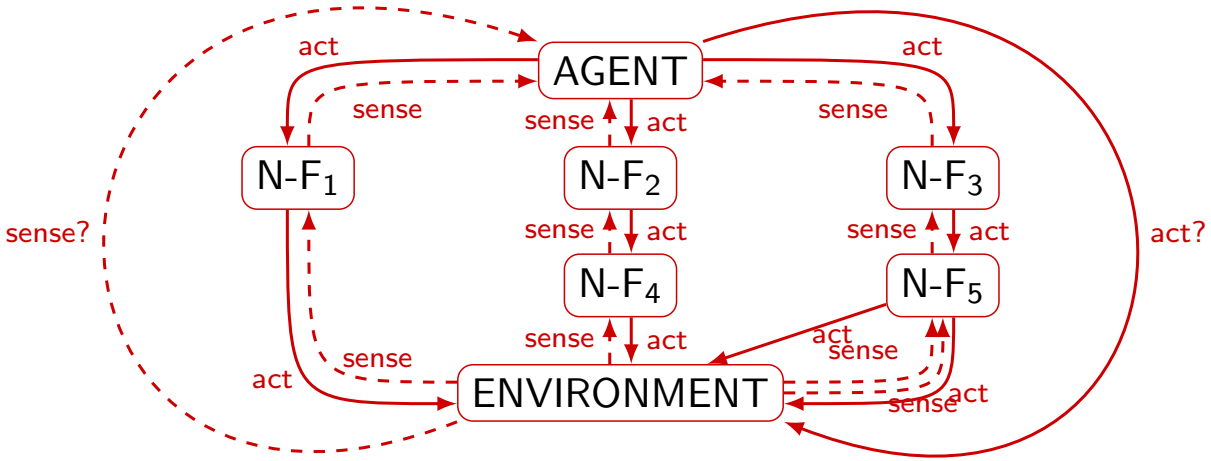
What is a normative framework?

- A set of rules:
  - capable of describing correct
  - and incorrect action,
  - obligations acquired through correct action
  - and sanctions levied for incorrect action
  - while maintaining a record through its internal state.
- set of rules that interprets some but not necessarily all of an agent's actions as correct or incorrect within that context: the norm-regulated agent.
- also referred to as an institution



What is a *Multi*-normative framework?

- But there is not just one normative framework
- An agent acts in several normative frameworks, concurrently, even simultaneously
- An normative framework has restricted competence; aggregation provides complex legal and/or social contexts
- Thus: a multi-normative framework is a combination of normative frameworks providing the complete interpretation of an agent's actions.



Representing State

- capture (in)formal contextualized expectations of behaviour
- Three kinds of normative facts:
  1. `perm(doX(agent1,...))`
  2. `obl(doX(agent1,...),beforeY,otherwiseZ)` or `obl(achieveX(agent1,...),beforeY,otherwiseZ)`
  3. `pow(doX,(agent1,...))`

  where
  - `achieveX` is a condition over normative facts
  - `beforeY` is either an event or a condition and
  - `otherwiseZ` is a violation event
- Domain facts depending on model

Actions change the (normative) world

- counts-as: $\mathcal{G}$: external action $\xrightarrow[\text{if(conditions)}]{\text{generates}}$ normative action
- normative facts represented by fluents
  1. fluent $\Rightarrow$ true if present, false otherwise

     $\mathcal{C}^{\uparrow}$: action $\xrightarrow[\text{if(conditions)}]{\text{initiates}}$ fluent
  2. inertial fluent:

     $\mathcal{C}^{\downarrow}$: action $\xrightarrow[\text{if(conditions)}]{\text{terminates}}$ ~~fluent~~

     good for facts true for a period with start and finish actions
  3. non-inertial fluent: $\mathcal{C}^{\mathcal{N}}$: $\xrightarrow[\text{if(conditions)}]{}$ fluent

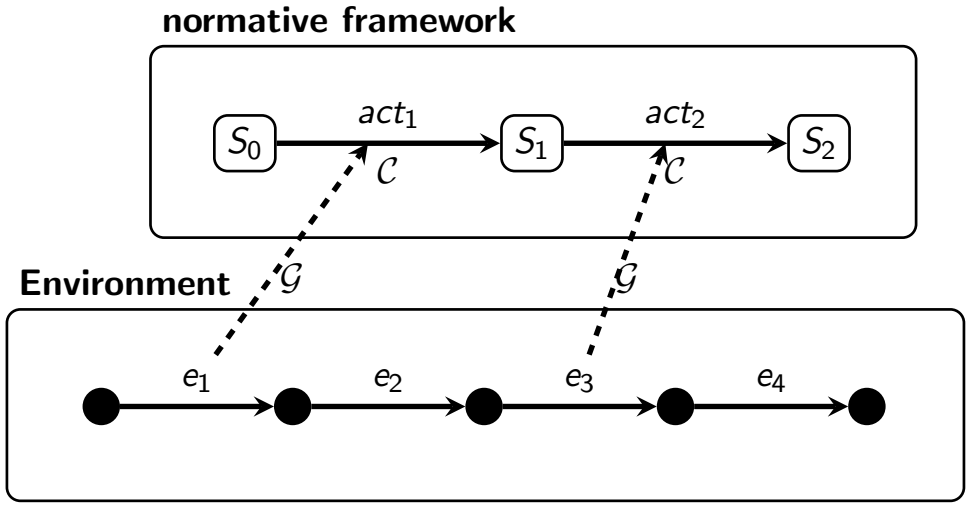
     good for facts expressed as a combination of inertial + non-inertial fluents

Making it work

- mathematical model:
  sets + relations $(\mathcal{G}, \mathcal{C}) \rightsquigarrow$ labelled transition system

$$\Delta \xrightarrow{e_1} S_1 \xrightarrow{e_2} S_2 \xrightarrow{e_3} \ldots$$

- conceptual model:

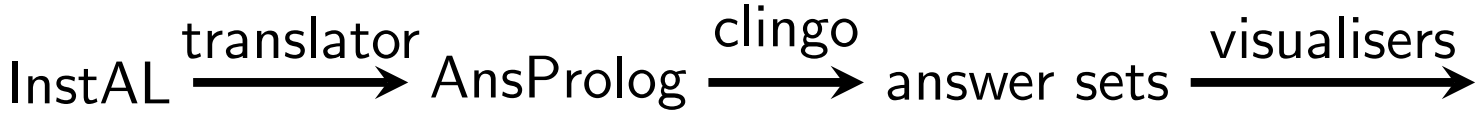


InstAL rules

1. `type AType;`: AType specifies the variables we use, e.g. Article
2. `fluent/nonintertial` *f* `;`: definition of (non inertial) fluent *f*
3. *x* `generates` *y* `[ if` *z* `];`: *x* an event (action), *y* one or more events; conditional *z*
4. *x* `initiates` *y* `[ if` *z* `];`: *x* an event, *y* one or more fluents; add to model state conditional on *z*
5. *x* `terminates` *y* `[ if` *z* `];`: as initiates, but remove fluents conditional on *z*
6. *x* `when` *y*: *x* non-inertial fluent, *y* a condition over the model state
7. `initially` *x* `[ if` *y* `];` : *x* one or more fluents in the initial model state, conditional on *y*

Implementation

- computational model:



- python front-end
- compiler: InstAL to Answer Set Programming
- python API to Clingo (answer set solver, C++)
- answer sets delivered in JSON
- visualization tools generate images from traces

Example

```
1  Instal
2
3  % the sale proceeds if sale is empowered and both
       participants have the correct assets
4  sale(Seller,Buyer) generates intSale(Seller,Buyer) if
       hasGood(Seller), hasMoney(Buyer);
5
6  % sale split between buyer and seller responsibilities
7  intSale(Seller,Buyer) generates transferReq(Seller),
       paymentReq(Buyer);
8  sale(Seller,Buyer) initiates contract(Seller,Buyer);
9
10 ASP
11 occurred(intSale(Seller,Buyer),I) :- occurred(sale(Seller,
       Buyer),I),holdsat(hasGood(Seller),I),holdsat(hasMoney(
       Buyer),I),holdsat(pow(legal,intSale(Seller,Buyer)),I),
       instant(I), agent(Seller),agent(Buyer).
12
13 occured(transferReq(Seller),I) :- occured(intSale(Seller,
       Buyer),I),holdsat(pow(legal,transferReq(Seller)),I),
       instant(I),agent(Seller),agent(Buyer).
14 occured(paymentReq(buyer),I) :-          occured(intSale(
       Seller,Buyer),I),holdsat(pow(legal,paymentReq(Buyer)),I)
       ,instant(I),agent(Seller),agent(Buyer).
15
16
17 initiated(contract(Seller,Buyer),I) :-
18 occured(sale(Seller,Buyer),I),holdsat(live(legal),I),instant
       (I),agent(Seller),agent(Buyer).
```

Background

- Two distinct perspectives of norms in the literature
  - emergent norms vs. prescriptive norms
    - distinct notion of norms
    - representation of norms
    - norm life cycle
- Opportunity for the two perspectives to complement each other
  - emergence of norms in the prescriptive perspective
  - agents as a source of norms
  - agents as stakeholder
  - lending itself to dynamic specifications
  - agents as norm synthesis actor

(Morris-Martin, De Vos, and Padget 2019)

Norm emergence framework

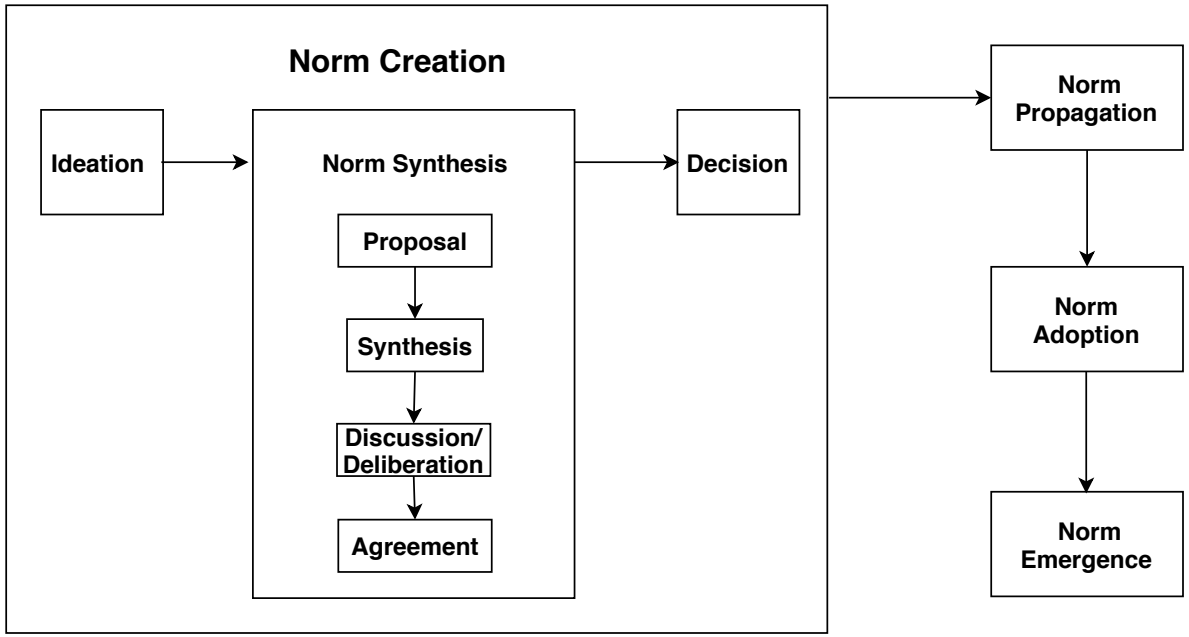


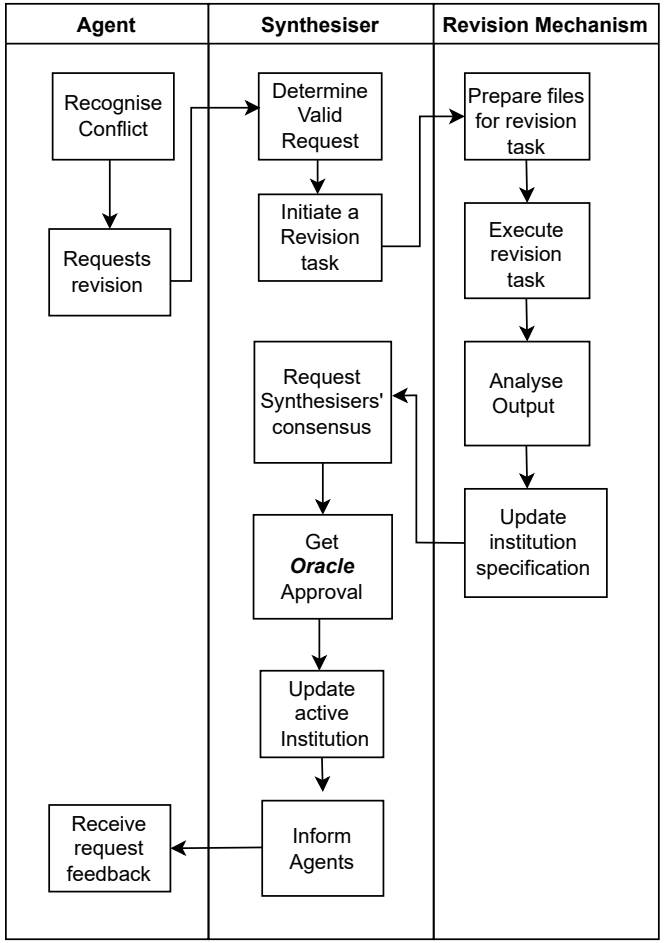
Figure: A summary of the conceptualisation of norm emergence for normative MAS

Agent-directed norm synthesis framework

- Aligns with two of Ostrom's eight principles (Ostrom 1990)
- Implements the norm creation stage of the framework
- Stages:
  1. ideation
  2. norm synthesis
     a. proposal
     b. synthesis
     c. deliberation
     d. agreement
  3. decision
- Two types of actors:
  - Participating agents
  - Synthesiser agents
- *Oracle* provides a system and functional boundary for the revision process.

(Morris-Martin, De Vos, and Padget 2020)

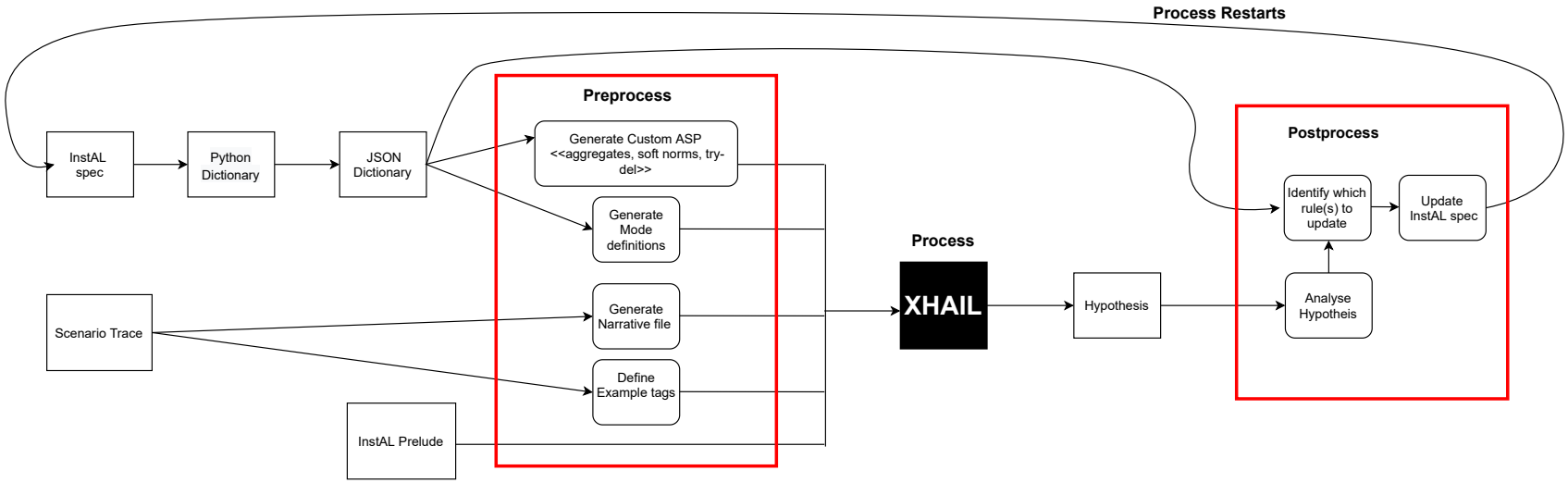
The agent-directed norm synthesis framework



Revision stages

1. Initiating
   - Select wanted/unwanted (elements of) traces
2. Guiding
   - Annotate the specification
   - Define the examples
   - Define the format of the output
3. Completing
   - Select an output
   - Translate it to InstAL
   - Update normative framework
4. Repeat

Revision Cycle



XHAIL

- Ray "Nonmonotonic abductive inductive learning",2009
- ILP system
- revision logic theories using +/- examples
- base theory
- mode declarations to constrain search
- meta-predicates try, use or del and exception to simulate causal refinement operator
- used for institutional debugging (Li et al. 2013; Corapi et al. 2011)

Case study

Case Study : Rooms Case study inspired by Sergot's room scenario (Sergot 2007).

- a set of disconnected rooms in a building
- agents can enter and leave rooms once they have the necessary permissions to do so
- rooms have a maximum capacity and some rooms have special purposes

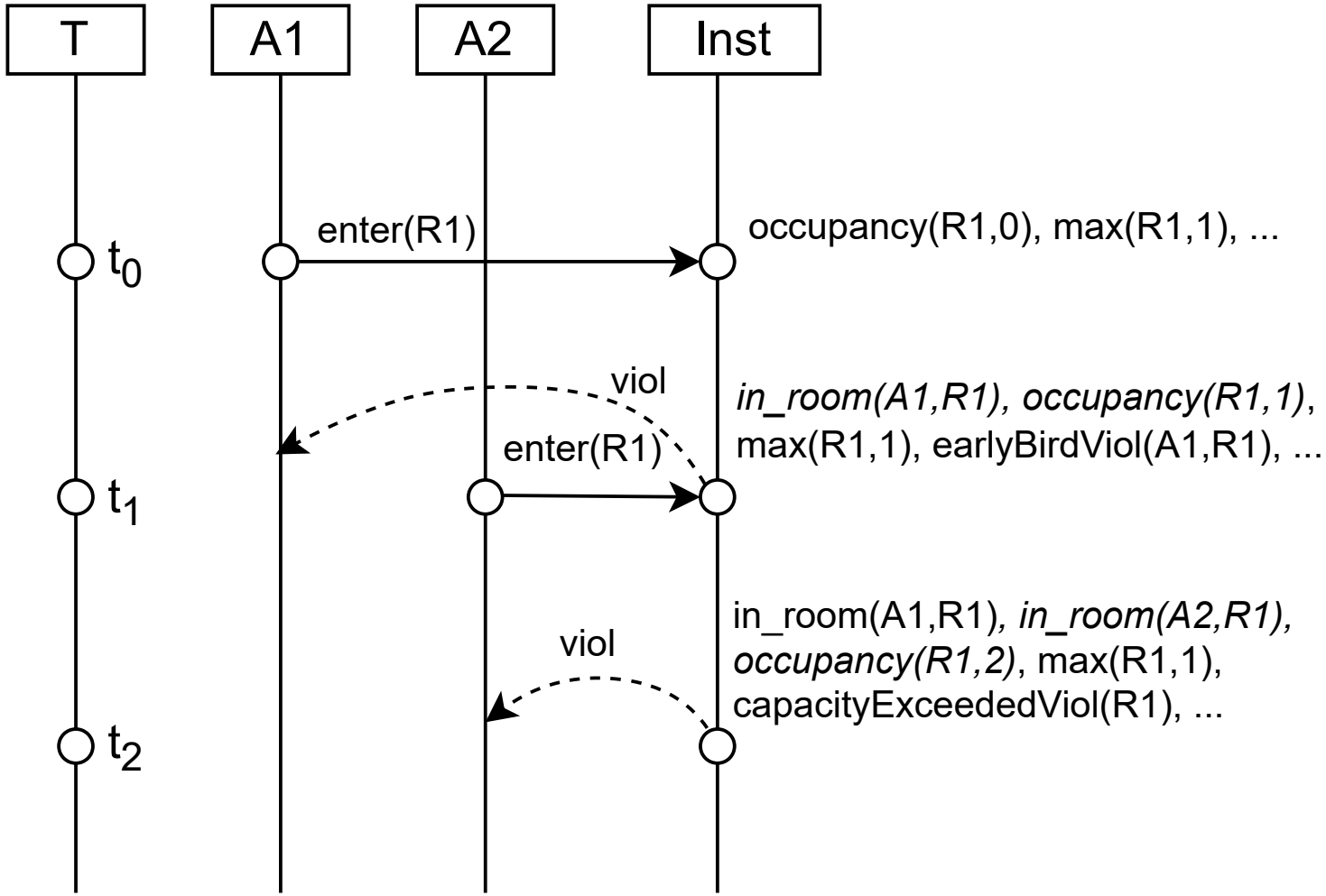
Code snippet

```
1  fluent in_room(Person,Location);
2  noninertial fluent occupancy(Location,Number);
3  noninertial fluent capacityExceededViol(Location);
4
5  exogenous event enter(Person,Location);
6  inst event arrive(Person,Location);
7
8  enter(P,L) generates arrive(P,L);
9  enter(P,L) generates deniedEntry(P,L) if not permEntry(P,L);
10 arrive(P,L) initiates in_room(P,L), perm(leave(P,L));
11 exit(P,L) terminates perm(leave(P,L)), in_room(P,L);
12 permEntry(P,L) when not in_some_room(P);
```

Introduced Issue

```
1  earlyBirdViol(P,L) when occupancy(L,X), equal(X,1), in_room(
       P,L);
2
3  capacityExceededViol(L) when occupancy(L,X), max(L,Y),
       bigger(X,Y);
```

Violation


T
A1
A2
Inst
t0
t1
t2
enter(R1)
occupancy(R1,0), max(R1,1), ...
viol
enter(R1)
in_room(A1,R1), occupancy(R1,1),
max(R1,1), earlyBirdViol(A1,R1), ...
viol
in_room(A1,R1), in_room(A2,R1),
occupancy(R1,2), max(R1,1),
capacityExceededViol(R1), ...

Annotating the Revision

```
1  earlyBirdViol(P,L) when occupancy(L,X), equal(X,1), in_room(
       P,L), revise;
```

```
1  % Rule ID: 30
2  holdsat(earlyBirdViol(P,L),rooms,I) :-
3     try(30, 1, holdsat(occupancy(L,X),rooms,I)),
4     try(30, 2, holdsat(equal(X,1),rooms,I)),
5     try(30, 3, holdsat(in_room(P,L),rooms,I)),
6     not exception(30,holdsat(earlyBirdViol(P,L),rooms,I)),
7     person(P),
8     location(L),
9     number(X),
10    instant(I).
11
12 try(30, 1, holdsat(occupancy(L,X),rooms,I)) :- not del(30,
       1), holdsat(occupancy(L,X),rooms,I), location(L), number
       (X), instant(I).
13
14 ...
```

Defining the examples

- Use one or more traces
- remove unwelcome fluents: `earlyBirdViol`

```
1  #example not holdsat(earlyBirdViol(R1),rooms,1).
```

Defining the mode declarations

```
1  #modeh exception($int30,holdsat(earlyBirdViol(+person,+
       location),$inst,+instant))=1.
2  int30(30).
3  clause30(1..3).
4  #modeh del($int30,$clause30)=3.
```

Defining constraints on InstAL generated

```
1  :- initiated(F,In,E,I), occurred(viol(E),In,I),ifluent(F,In)
       ,event(E),inst(In), instant(I).
```

Completing the revision

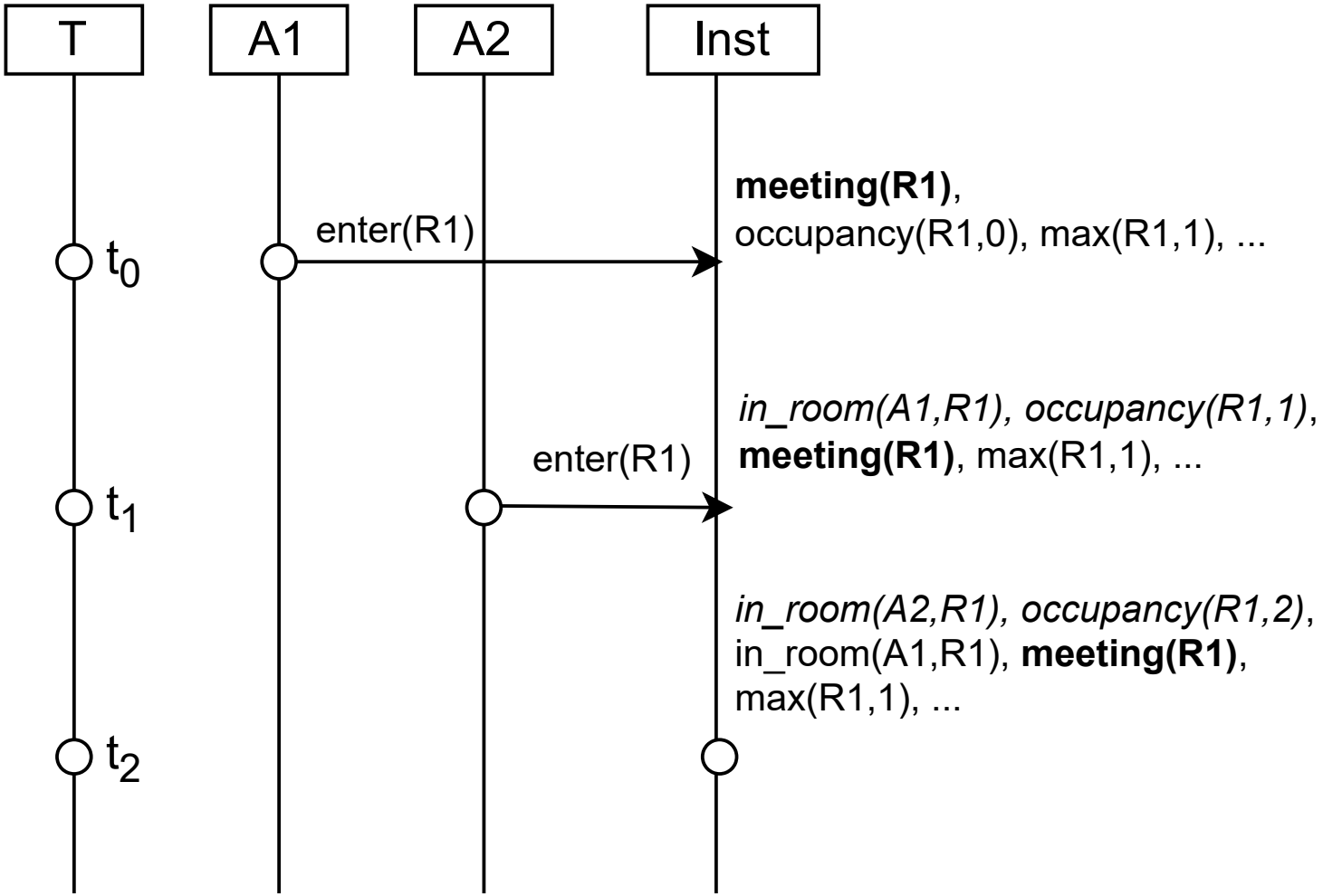
XHAIL:

```
1 exception(30,holdsat(earlyBirdViol(V1,V2),rooms,V3)).
```

Revision:

```
1 %OLD RULE 30
2 earlyBirdViol(P,L) when occupancy(L,X), equal(X,1), in_room(
      P,L), revise;
3 %NEW RULE 30
4 %%% DELETED RULE %%%
```

Violation Removed



Future

- norm internalisation
- normative history
- norm evolution
- self-governance

Collaborators

- Tina Balke
- Owen Cliffe
- Domenico Corapi
- Sabrina Kirrane
- Tingting Li
- Jack McKinlay
- Alessandra Mileo
- Fahid Mohammed
- Andreassa Morris Martin
- Julian Padget
- Oliver Ray
- Alessandra Russo
- Ken Satoh

References I

Corapi, Domenico et al. (July 2011). "Normative design using inductive learning". en. In: *Theory and Practice of Logic Programming* 11.4-5, pp. 783–799. ISSN: 1471-0684, 1475-3081. DOI: 10.1017/S1471068411000305. URL: https://www.cambridge.org/core/product/identifier/S1471068411000305/type/journal_article (visited on 07/07/2020).

References II

Li, Tingting et al. (2013). "Legal Conflict Detection in Interacting Legal Systems". In: *Legal Knowledge and Information Systems - JURIX 2013: The Twenty-Sixth Annual Conference, December 11-13, 2013, University of Bologna, Italy*. Vol. 259. Frontiers in Artificial Intelligence and Applications. IOS Press, pp. 107–116. DOI: 10.3233/978-1-61499-359-9-107. URL: https://doi.org/10.3233/978-1-61499-359-9-107.

References III

Morris-Martin, Andreasa, Marina De Vos, and Julian Padget (2020). "A Norm Emergence Framework for Normative MAS - Position Paper". In: *Coordination, Organizations, Institutions, Norms, and Ethics for Governance of Multi-Agent Systems XIII - International Workshops COIN 2017 and COINE 2020, Sao Paulo, Brazil, May 8-9, 2017 and Virtual Event, May 9, 2020, Revised Selected Papers*. Ed. by Andrea Aler Tubella et al. Vol. 12298. Lecture Notes in Computer Science. Springer, pp. 156–174. DOI: 10.1007/978-3-030-72376-7_9. URL: https://doi.org/10.1007/978-3-030-72376-7%5C_9.

References IV

- Morris-Martin, Andreasa, Marina De Vos, and Julian Padget (Nov. 2019). "Norm emergence in multiagent systems: a viewpoint paper". In: *Autonomous Agents and Multi-Agent Systems* 33.6, pp. 706–749. ISSN: 1573-7454. DOI: 10.1007/s10458-019-09422-0. URL: https://doi.org/10.1007/s10458-019-09422-0.
- Morris-Martin, Andreasa, Marina De Vos, et al. (2023). "Agent-directed Runtime Norm Synthesis". In: *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2023, London, United Kingdom, 29 May 2023 - 2 June 2023*. Ed. by Noa Agmon et al. ACM, pp. 2271–2279. DOI: 10.5555/3545946.3598905. URL: https://dl.acm.org/doi/10.5555/3545946.3598905.

References V

- Ostrom, Elinor (1990). *Governing the Commons. The Evolution of Institutions for Collective Action*. CUP.
- Ray, Oliver (2009). "Nonmonotonic abductive inductive learning". In: *Journal of Applied Logic* 7.3, pp. 329–340. ISSN: 1570-8683. DOI: https://doi.org/10.1016/j.jal.2008.10.007. URL: https://www.sciencedirect.com/science/article/pii/S1570868308000682.

References VI

Sergot, Marek J. (2007). "Action and Agency in Norm-Governed Multi-agent Systems". In: *Engineering Societies in the Agents World VIII, 8th International Workshop, ESAW 2007, Athens, Greece, October 22-24, 2007, Revised Selected Papers*. Ed. by Alexander Artikis et al. Vol. 4995. Lecture Notes in Computer Science. Springer, pp. 1–54. DOI: 10.1007/978-3-540-87654-0_1. URL: https://doi.org/10.1007/978-3-540-87654-0%5C_1.