Acting on Norm Constrained Plans

Nir Oren, Wamberto Vasconcelos, Felipe Menguzzi, Michael Luck

n.oren@abdn.ac.uk, wvasconcelos@acm.org,
meneguzz@cs.cmu.edu, michael.luck@kcl.ac.uk

University of Aberdeen
Carnegie Mellon University
King's College London

Overview

How can a BDI-like agent decide which plan to execute within an environment containing norms?

- System Components
  - Constraints
  - Actions and Plans
  - Norms, Permissions and Conflicts
- Putting it all Together
  - Environment
  - Executing Actions
  - From Plans to Norm Constrained Actions

Constraints

- We utilise constraints to describe, and restrict actions.
- A set of constraints is viewed as a conjunction of individual constraints.

$$X < 4, Z \geq Y\ M = R + 4$$

- Notation: Γ is a set of constraints. Standard definitions for unification, satisfaction etc.

Action

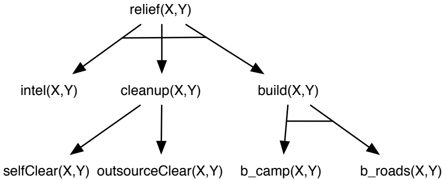
- An action ($\psi \circ \Gamma$) consists of a predicate and a constraint binding values to the variables in the predicate.
- Abstract actions have unground variables.

$$move(A, B, X, Y) \circ A = X \wedge B = Y$$

- One action, α can *entail* another, β iff whenever the constraints of α are satisfied, so are those of β.

Plans

- We treat a plan as a AND/OR tree (c.f. simple HTN planning).
- Leaf nodes represent primitive actions.



$$\textbf{andN}(\mathit{intel}(X,Y),\ \textbf{orN}(\mathit{selfClear}(X,Y),\mathit{outsourceClear}(X,Y)),\ \mathit{b\_camp}(X,Y),\mathit{b\_roads}(X,Y))$$

- Actions in plans *can* have constraints.

Obligations

- Oα, Pα, Fα where $\alpha = \psi \circ \Gamma$
- Like actions, we define entailment between norms, representing a specialisation relationship over norms.
- What does $\omega = \mathrm{O}\psi \circ \Gamma$ mean? Two choices:
  - It is obligatory to execute ψ as constrained by Γ
  - If executing action ψ, it is obligatory to adhere to constraints Γ

Permissions and Prohibitions

- Permissions are exceptions to obligations and prohibitions.
- They have no meaning in isolation.

$$O selfClear(X, Y) \circ \{X < 30, Y = 20\} \quad P selfClear(X, Y) \circ \{X < 40\}$$

- Allows X to be less than 40 when the obligation is present without violating the obligation.
- The permission thus *mitigates* the obligation/prohibition.
- Prohibitions forbid an action to take place with the values specified in the constraint.
- A set of norms is in conflict if there is no consistent way to satisfy all its constraints (given the presence of mitigating permissions).

Normative Rules

- Norms are typically situation dependent.
- A simple normative language identifies when a norm starts or ceases to exist.

$$
\boxed{
\begin{array}{rcl}
R & ::= & LHS \Rightarrow RHS \\
LHS & ::= & \alpha | \alpha \wedge NLHS | NLHS \\
NLHS & ::= & \omega | NLHS \wedge NLHS \\
RHS & ::= & RHS \wedge RHS | \oplus \omega | \ominus \omega
\end{array}
}
$$

- The language allows norm modification on action or conditional on the existence of another norm.
- Based on the work of Garcia-Camino *et. al.*

Enactment States

- The execution of an action
  - Modifies the physical environment.
  - Can cause new norms to be instantiated, or existing ones to be removed.
  - Might place constraints on future actions (via variable bindings).
- We represent the domain as a transition system between individual *enactment states*.
- Each enactment state captures the system at a single time point.

$$\Delta = (\Omega, \Gamma) \text{ where } \Omega = \{\omega_1, \ldots \omega_n\}$$

- An enactment state identifies the (hard) constraints that exist, and the norms that are in force.

Transitioning Between States

- Garcia-Camino *et. al.* defined rules for (unambiguously) transitioning between enactment states.
- Our focus is different; we want to identify the *possible* enactment states that can result from the execution of an action.

  $intel(X, Y) \Rightarrow \oplus\omega_1 \quad intel(5, 6) \Rightarrow \oplus\omega_2 \quad intel(7, 8) \Rightarrow \oplus\omega_3$
- $intel(2, 2)$ results in ω_1 within the new enactment state.
- $intel(5, 6)$ results in ω_1, ω_2 within the new enactment state.
- $intel(A, B)$?
  - $\{\omega_1\}, \{\omega_1, \omega_2\}, \{\omega_1, \omega_3\}$, constrained appropriately.

Transitioning Between States

- Given an enactment state, an action and a set of normative rules, we identify a set of *potentially applicable rules*. I.e. rules for which the LHS holds w.r.t the action and enactment state.
- We check for consistency in the constraints computed from the action executed, existing constraints and each element in the powerset of potentially applicable rules.
- Small subtlety: we need to include the constraints of the potentially applicable rules that are not applied.

$$r_1 = \mathit{intel}(X, Y) \circ X < 5 \Rightarrow \oplus\omega_1 \quad r_2 = \mathit{intel}(X, Y) \circ Y > 2 \Rightarrow \oplus\omega_2$$

$$\begin{array}{ll} \langle\{X < 5, Y > 2\}, \{\omega_1, \omega_2\}\rangle, & \langle\{X < 5, Y \leq 2\}, \{\omega_1\}\rangle, \\ \langle\{X \geq 5, Y > 2\}, \{\omega_2\}\rangle, & \langle\{X \geq 5, Y \leq 2\}, \{\}\rangle \end{array}$$

- We remove all enactment states obtained due to the application of non-maximally consistent sets of potentially applicable rules.

Transitioning Between States

- We place an ordering constraint on norm modification, adding norms before removing them.
- For any given path through the tree of enactment states, Γ is monotonic, tracking all constraints that have been imposed to that point in time.
- Note: constraints are only added due to the LHS of a rule, not its RHS.

Where are we?

- Given a partially or fully ground action, an enactment state and a current set of norms we can now identify all possible enactment states that can be generated from that state.
- This enactment state identifies the constraints affecting the agent, and the “active” norms at that point in time.
- How can we decide what actions to execute within some enactment state?
- A few assumptions
  - We have a plan library with each plan containing partially constrained actions.
  - Achieving a plan yields utility.
  - Violating a prohibition or an obligation, executing an action or utilising a permission, costs utility.
  - Complying with norms yields utility.

$$cost : Act \times 2^{Norms} \times 2^{Norms} \rightarrow \mathbb{R}$$

Plan Selection

- We can select a plan for execution from a set of plans by
  - Computing an enactment state tree for each possible path through the plan.
  - Identifying the tree with the maximal associated utility.
- Rather than do all of this up front, we can perform a best first incremental search in the enactment state space
  - Select a subset of norms for compliance.
  - Minimally constrain the action to comply with those norms.

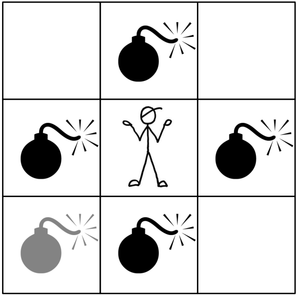
Plan Selection

- It is easy to modify the basic approach to represent a fully norm compliant agent.
- The algorithm is guaranteed to terminate and is sound and complete.
- But of exponential complexity.
- It does however have anytime properties as we always track the best action sequence to date.

Heuristics and Planning Complexity

- It's possible to modify our basic plan selection algorithm to act as an A* search. It's more difficult to find an admissible heuristic.
    - Assume no more norm violations will occur
    - That all norms will be complied with
    - Monte-Carlo plan sampling
- It's also possible to prune plans which appear bad when compared to the current best plan.
    - Removes completeness guarantee, unless the utility gain function is monotonic.

An Example



An Example

$$\textbf{andN}(\mathit{scanC}, \mathit{moveC}, \textbf{orN}(\mathit{nothing}, \mathit{pickup}, \mathit{explodeC}))$$

$$\begin{aligned}\mathit{moveC} \equiv \textbf{orN}(&\mathit{move}(X, Y, A, B) \cdot A = X \wedge B = Y, \\ &\mathit{move}(X, Y, A, B) \cdot A = X + 1 \wedge B = Y, \\ &\mathit{move}(X, Y, A, B) \cdot A = X - 1 \wedge B = Y, \\ &\mathit{move}(X, Y, A, B) \cdot A = X \wedge B = Y + 1, \\ &\mathit{move}(X, Y, A, B) \cdot A = X \wedge B = Y - 1)\end{aligned}$$

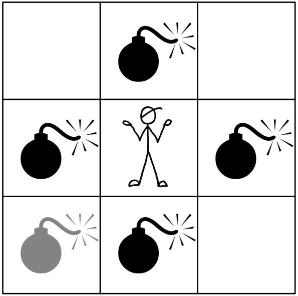
$$\mathit{pickup}(C, D), C = A \wedge D = B$$

An Example

- 10 normative rules
  - Preventing wandering out of the area
  - Stepping on dangerous bombs
  - Exploding bombs, . . .

$$
\begin{aligned}
&move(R4XO, R4YO, R4X, R4Y) \cdot \top \Rightarrow \\
&\quad \oplus \mathsf{F} explode(R4A, R4B) \cdot ((R4A = R4X \wedge R4B = R4Y) \vee \\
&\qquad (R4A = R4X - 1 \wedge R4B = R4Y) \vee \\
&\qquad (R4A = R4X + 1 \wedge R4B = R4Y) \vee \\
&\qquad (R4A = R4X \wedge R4B = R4Y - 1) \vee \\
&\qquad (R4A = R4X \wedge R4B = R4Y + 1)))
\end{aligned}
$$

Example



- We evaluated a norm aware agent, fully norm complaint agent and a norm unaware agent (following the basic plan).
- Results were unsurprising...
- One difficulty we encountered was representing the sensing action in the plan.

Future Work

- Identify the effects of heuristics on the algorithm.
- Generalise our representation of obligation.
- Enrich the language
  - Richer constraints
  - multiple actions in a rule
- Integrate sensing/action effects into the model.
- Integrate uncertainty into the approach (MDPs)?