# Property Persistence

Adrian Pearce

13 July 2011

includes slides by Ryan Kelly

# Outline

## Regression: Another example

Recall our thief wants prove

$$\mathcal{D} \models (\exists a, b) : walk(a, b), enter(bank(b), Bob), crackSafe(bank(b), Bob)$$

As owners of the gold, we would like to ensure that the thief cannot steal it. Unfortunately this is not possible, as nothing prevents him from simply cracking the safe and taking the gold.

We can, however, ensure that the thief cannot steal the gold *undetected*.

$$\mathcal{D} \models (\forall s) : s_0 \leq_{Undetected} s \rightarrow \neg Stolen(s)$$

which will be true when the following is true

$$\mathcal{D}_{bg} \cup \mathcal{D}_{S_0} \models \neg Stolen(S_0) \wedge [\neg SafeOpen(S_0) \vee LightOn(S_0)]$$

# Regression: Another example (continued)

Unfortunately, this is not possible within the framework of regression presented thus far for two reasons

- $(\forall s) : s_0 \leq_{Undetected} s \supset \neg Stolen(s)$ is not a regressable formulae, since it quantifies over situations.
- The difficulty stems from the second-order induction axiom to define the set of all situations - meaning we can't easily regress over formulae that includes universal quantification.

# Property Persistence

Suppose we could "factor out" the quantification. Then we could get on with the business of doing regression.

**Definition (Persistence condition)**

The *persistence condition* $\mathcal{P}[\phi, \alpha]$ of a formula $\phi$ and action conditions $\alpha$ to mean: assuming all future actions satisfy $\alpha$, $\phi$ will remain true.

$$\mathcal{P}[\phi, \alpha](s) \equiv \forall s' : s \leq_\alpha s' \rightarrow \phi(s')$$

Like $\mathcal{R}$, the idea is to transform a query into a form that is easier to deal with.

# Property Persistence

The persistence condition can be calculated as a fixpoint:

$$\mathcal{P}^1[\phi, \alpha](s) \stackrel{\text{def}}{=} \phi(s) \wedge \forall c : \alpha(c) \rightarrow \mathcal{R}[\phi(do(c, s))]$$

$$\mathcal{P}^n[\phi, \alpha](s) \stackrel{\text{def}}{=} \mathcal{P}^1[\mathcal{P}^{n-1}[\phi, \alpha], \alpha]$$

$$(\mathcal{P}^n[\phi, \alpha] \rightarrow \mathcal{P}^{n+1}[\phi, \alpha]) \Rightarrow (\mathcal{P}^n[\phi, \alpha] \equiv \mathcal{P}[\phi, \alpha])$$

This calculation can be done using *static domain reasoning* and provably terminates in several important cases.

# Property persistence theorem

The property persistence theorem guarentees that if $\mathcal{P}^n_{\mathcal{D}}(\phi, \alpha)$ implies $\mathcal{P}^{n+1}_{\mathcal{D}}(\phi, \alpha)$, then $\mathcal{P}^n_{\mathcal{D}}(\phi, \alpha)$ is the persistence condition for $\phi$ under $\alpha$.

## Theorem

*Given a basic action theory $\mathcal{D}$, uniform formula $\phi$ and action description predicate $\alpha$, then for any n:*

$$\mathcal{D}_{bg} \models \forall s : \mathcal{P}^n_{\mathcal{D}}(\phi, \alpha)[s] \rightarrow \mathcal{P}^{n+1}_{\mathcal{D}}(\phi, \alpha)[s]$$
$$\text{iff}$$
$$\mathcal{D} - \mathcal{D}_{s_0} \models \forall s : \mathcal{P}^n_{\mathcal{D}}(\phi, \alpha)[s] \equiv \mathcal{P}_{\mathcal{D}}(\phi, \alpha)[s]$$

## Calculating Persistence

Define $\mathcal{P}^1[\phi, \alpha]$ to be "persistence to depth 1":

$$\mathcal{P}^1[\phi, \alpha](s) \stackrel{\text{def}}{=} \phi(s) \wedge \forall c.[\alpha(c, s) \Rightarrow \mathcal{R}[\phi(do(c, s))]]$$

We can assert that a property holds to depth 2, 3, ... by repeatedly applying $\mathcal{P}^1$:

$$\mathcal{P}^n[\phi, \alpha] = \mathcal{P}^1[\mathcal{P}^{n-1}[\phi, alpha], \alpha]$$

We want persistence for *any* $n$: need the least-fixed point of $\mathcal{P}^1$. Fixed-point theory guarantees we can calculate this by trans-finite iteration.

# Proving Invariants in Programs & Plans (Continued)

**Need for Cooperation:** Given an agent *agt*, goal $G$ and situation $\sigma$, establish that no sequence of actions performed by that agent can achieve the goal. Suppose we define *MyAction* to identify the agent's own actions:

$$MyAction(a, s) \stackrel{\text{def}}{=} actor(a) = agt$$

Then the appropriate query is:

$$\mathcal{D} \models \forall s : \sigma \leq_{MyAction} s \rightarrow \neg G(s)$$

If this is the case, the agent will need to seek cooperation from another agent in order to achieve its goal.

# Proving Invariants in Programs & Plans (Continued)

**Knowledge with Hidden Actions:** An agent reasoning about its own knowledge in asynchronous domains must account for arbitrarily-long sequences of hidden actions. To establish that it knows $\phi$, it must establish that $\phi$ cannot become false through a sequence of hidden actions:

$$\mathcal{D} \models \forall s : \sigma \leq_{Hidden} s \rightarrow \phi[s]$$

# References


- Ray Reiter. Proving Properties of States in the Situation Calculus. Artificial Intelligence, 64:337-351, 1993.
- Ryan F. Kelly and Adrian R. Pearce. Property Persistence in the Situation Calculus. In Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI'07), pages 1948-1953, 2007
- Ryan Kelly. "Asynchronous Multi-Agent Reasoning in the Situation Calculus", PhD Thesis, The University of Melbourne, 2009

# Summary