Reasoning about Large Taxonomies of Actions in the Situation Calculus

Yilan Gu
Department of Computer Science, University of Toronto, Canada
Mikhail Soutchanski
Department of Computer Science, Ryerson University, Canada

Motivation

The situation calculus [Reiter, 2001]

- Initially proposed by J. McCarthy, enriched by R. Reiter et al at Univ. of Toronto
- A high-level first-order logic language
- Representing and reasoning about actions in a natural and compact way

Problem: Handling effects of action functions individually and impractical to handle large scales of action functions

Our Idea

- Using common-sense knowledge: Considering one action as a specialization of another
- E.g., traveling and shipping can be considered as specializations of moving

Summary of Contributions

1. Representing large taxonomies of actions in a hierarchical way
2. Representing effects of actions much more compactly based on action hierarchies
3. Reasoning problems can be solved exponentially faster (sometimes)
4. Bringing possible knowledge engineering advantages

Background: The Situation Calculus

- Action functions: fry(x,y) (frying x using y), wash(x) (washing x)
- Situations: S0 (the initial situation), do(a,s) (the situation after doing a in s)
- Objects: persons, locations, food items, ...

Fluents

- System features whose truth values may vary from situation to situation
- Predicates or functions with the last argument of sort situation
- E.g., FoodReady(x,s) - food x is ready in the situation s

A Basic Action Theory (BAT) \( D \)

- Describing actions and their effects in a dynamic system
- A set of first-order axioms, mainly includes:
  \[ D_{\text{pre}}(\text{precondition axioms}): \text{For each } A(x), \quad \text{Pos}(A(x),s) \equiv \phi_A(x,s) \]
  \[ D_{\text{act}}(\text{successor state axioms}): \text{For each fluent } F(y), \quad F(y,s) \equiv \text{Pos}(A(x,y),s) \land \sim \text{Pos}(A(x,y),s) \]
  \[ D_{\text{acts}}(\text{an initial theory}): \text{All facts hold in the initial situation } S_0 \]
  \[ D_{\text{preAxis}}(\text{unique name axis for action names}) \]

An example of a kitchen activity domain

- Considering actions involved in a kitchen such as cook(x), fry(x,y), wash(x), ..., etc. and fluents such as FoodReady(x,s), Dirty(x,s), etc. (See actions in the digraph on the right-hand side)

A Modular Basic Action Theory \( D^N \)

- Format of actions
  \[ D_{\text{act}}^N \quad \text{and } \Sigma \text{ are as usual} \]
  \[ D_{\text{acts}}^N \quad \text{includes action hierarchy axioms } H^* \text{ and axioms for event slots in addition} \]
  \[ D_{\text{acts}}^N \quad \text{New representation of basic action theories using isA} \]
  \[ \text{E.g., } \text{Dirty}(y,s) \equiv (\exists y \text{isA}a, \text{prepFood}(x,y)) \land \text{EventSlot}(a,y,\text{Utensils}) \land \text{Dirty}(y,s) \land \sim \text{isA}(a,\text{wash}(y)) \]

Regression: operator is extended to handle isA and EventSlot

Correctness of Modular Basic Action Theories

1. For any modular BAT \( D^N \), there exists an equivalent D of Reiter’s BAT format, where equivalence means that for any first-order resolvable sentence \( W \), \( D^N \models W \iff D \models W \)
2. A regression theorem similar to Reiter’s regression theorem is proved:
   \( D^N \models W \iff D_{\text{acts}}^N \cup D_{\text{acts}} \models W \)
3. Although the formal definition of isA is second-order, the reasoning in \( D^N \) can be reduced to a first-order logic reasoning only

Advantages of the New Approach

- Computational advantages
  - When \( H \) has a tree or forest structure, then the reasoning on isA is exponentially faster than the reasoning on its equivalent clause in Reiter’s BAT representation.

Knowledge engineering advantages

- Allowing taxonomic reasoning about event slot hierarchies and action hierarchies
- Possible applications to action/service retrieval
- Easiness for system update and reuse of action hierarchies and modular BATs

Implementation (in progress): E-business domain and kitchen activity domain

References and Most Related Work

- Amir, 2000: (De)Composition of Situation Calculus Theories
- Barker, Porter & Clark, 2001: A Library of Generic Concepts for Composing KBs
- Gil, 2005: Description Logics and Planning
- Gu & Soutchanski, 2007: Modular Basic Action Theories
- Lifschitz & Ren, 2006: A Modular Action Description Language