Inductive Situation Calculus

Marc Denecker [a] Eugenia Ternovska [b]

[a] Department of Computer Science, K.U.Leuven, Belgium
[b] School of Computing Science, Simon Fraser University

Inductive Situation Calculus

Marc Denecker [a] Eugenia Ternovska [b]

[a] Department of Computer Science, K.U.Leuven, Belgium
[b] School of Computing Science, Simon Fraser University

The full version of this paper appeared in the Proceedings of Ninth International Conference on Principles of Knowledge Representation and Reasoning, 2004. For the reference, see [2].

Abstract

Temporal reasoning has always been a major test case for knowledge representation formalisms. In this paper, we develop an inductive variant of the situation calculus using the Logic for Non-Monotone Inductive Definitions (NMID). This is an extension of classical logic that allows for uniform representation of various forms of definitions, including monotone inductive definitions and non-monotone forms of inductive definitions such as iterated induction and induction over well-founded posets [1].

Here, we demonstrate an application of NMID-logic. The aim is two-fold. First, we illustrate the role of NMID-logic and non-monotone inductive definitions for knowledge representation by presenting a variant of the situation calculus which we call *inductive situation calculus*. We show that ramification rules can be naturally modeled through a non-monotone iterated inductive definition. Second, we illustrate the use of our recently developed modularity techniques for NMID-logic in order to translate a theory of the inductive situation calculus into a classical logic theory of Reiter's situation calculus [3].

There are several points of interest in this experiment. The first one is our observation that complex non-monotone inductive definitions not only occur in mathematics, but also in common sense reasoning. In particular, we believe that the original Reiter-style situation calculus contains hidden forms of definitions which we explicitate in the inductive situation calculus. The second point is the fact that different forms of inductive definitions, which have a uniform representation in NMID-logic, can be formalised in classical (first- or second-order) logic as well, but not in a uniform way: different sorts of definitions require different formalisation. As a consequence, our formalisation is simpler, more uniform and more modular than Reiter-style situation calculus. The third point is that NMID-logic is closely connected to logic programming. In particular, it formally extends logic programming (LP) and abductive logic programming (ALP) under the well-founded semantics. The strong connection with LP and ALP can be exploited to build implementations of NMID-logic and of inductive situation calculus theories. Moreover, NMID-logic has an important advantage as a knowledge representation language

— contrary to logic programming, it does not automatically impose the domain closure assumption (although the domain closure axiom can be expressed if needed).

To illustrate the above ideas, let us describe a simple idealized mechanical system of two connected gear wheels. Suppose fluents T_1, T_2 represent the fact that the first (respectively, the second) wheel is turning. The external events that make a wheel start or stop turning is denoted by action $Start_1$, $Start_2$, $Stop_1$, or $Stop_2$, respectively.

The core of the representation of this example is a simultaneous definition defining the fluents and causation predicates. Each of the two fluents are defined by the following rules:

$$\begin{array}{rcl}
\forall a \forall s\ (T_1(S_0) & \leftarrow & I_{T_1}), \\
\forall a \forall s\ (T_1(do(a,s)) & \leftarrow & C_{T_1}(a,s)), \\
\forall a \forall s\ (T_1(do(a,s)) & \leftarrow & T_1(s) \wedge \neg C_{\neg T_1}(a,s)) \\
\forall a \forall s\ (T_2(S_0) & \leftarrow & I_{T_2}), \\
\forall a \forall s\ (T_2(do(a,s)) & \leftarrow & C_{T_2}(a,s)), \\
\forall a \forall s\ (T_2(do(a,s)) & \leftarrow & T_2(s) \wedge \neg C_{\neg T_2}(a,s))
\end{array}$$

In this, T_i is the fluent symbol, I_{T_i} represents T_i in the initial state, $C_{T_i}(a,s)$ represents that action a makes T_i to become true in state s and $C_{\neg T_i}(a,s)$ represents that action a makes T_i to become false in state s. The first rule of this subdefinition simply associates the fluent in the initial situation with its initial situation predicate. The second rule says that if a fluent is caused to hold in a situation, then it holds in that situation. The third rule models the inertia law.

The following rules specify direct and indirect effects of actions:

$$\begin{array}{ll}
\forall a \forall s\ (C_{T_1}(a,s) \leftarrow a = Start_1), & \forall a \forall s\ (C_{T_2}(a,s) \leftarrow a = Start_2), \\
\forall a \forall s\ (C_{\neg T_1}(a,s) \leftarrow a = Stop_1), & \forall a \forall s\ (C_{\neg T_2}(a,s) \leftarrow a = Stop_2), \\
\forall a \forall s\ (C_{T_1}(a,s) \leftarrow C_{T_2}(a,s)), & \forall a \forall s\ (C_{T_2}(a,s) \leftarrow C_{T_1}(a,s)), \\
\forall a \forall s\ (C_{\neg T_1}(a,s) \leftarrow C_{\neg T_2}(a,s)), & \forall a \forall s\ (C_{\neg T_2}(a,s) \leftarrow C_{\neg T_1}(a,s))
\end{array}$$

Each rule represents a local property of the system, namely an individual causal effect. By combining them we obtain a modular description of the entire system. Notice that the causal dependencies form a positive cycle.

All fluents are defined by simultaneous induction on the well-ordered set of situations. Ramifications describing propagation of effects of actions are modeled as monotone inductions at the level of situations. The result is an iterated inductive definition with alternating phases of monotone and non-monotone induction.

References

[1] M. Denecker and E. Ternovska. A logic for non-monotone inductive definitions and its modularity properties. In *Proc. of LPNMR-04*, 2004.

[2] Marc Denecker and Eugenia Ternovska. Inductive Situation Calculus. In *Proceedings of Ninth International Conference on Principles of Knowledge Representation and Reasoning, Delta Whistler Resort, Canada*, 2004. URL = http://www.cs.kuleuven.ac.be/cgi-bin-dtai/publ_info.pl?id=41085.

[3] R. Reiter. *Knowledge in Action: Logical Foundations for Describing and Implementing Dynamical Systems*. MIT Press, 2001.