An Assessment of Electronic Publishing

Linnea Lärd
Linus Erudit
Abstract

The paper analyzes and compares methods for ramification, or for update, which are based on minimization of change while respecting static domain constraints. The main results are:

- An underlying semantics for propagation oriented ramification, called causal propagation semantics.
- Comparative assessments of a number of previously proposed entailment methods for ramification, in particular, methods based on minimization of change after partitioning of the state variables (fluents). This includes the methods previously proposed by del Val and Shoham, by Kartha and Lifschitz, and by our group.
- Assessment of two ranges of sound applicability for one of those entailment methods, MSCC, relative to causal propagation semantics.
- Identification of an essential feature for minimization-based approaches to ramification, namely changeset-partitioning.

Authors’ Affiliations

Linnea Lärd, Linus Erudit
Department of Computer and Information Science
Linköping University
SE-58183 Linköping, Sweden
E-mail: {linla, liner}@ida.liu.se

Acknowledgement

This work was supported by the Swedish Research Council (Vetenskapsrådet) on grant 2004-012.
1 Topic and approach

1.1 The ramification problem

The ramification problem for reasoning about actions and change has been described as follows by Ginsberg and Smith, [?]:

The difficulty is that it is unreasonable to explicitly record all of the consequences of an action, even the immediate ones. For example [...] For any given action there are essentially an infinite number of possible consequences that depend upon the details of the situation in which the action occurs.

More concretely, the difference between ramification and the simpler case of strict inertia is that strict inertia assumes action laws to specify all the effects of the action explicitly, whereas in ramification the action law only needs to specify some of the effects; the others are to be deduced. On the other hand, the logic cannot do magic; those other effects must also be declared somehow. One can think of two specific reasons why ramification is useful:

1. In order to avoid duplication of the same information on the consequent side of several action laws. Suppose, for a trivial example, that both alive(x) and dead(x) occur as state variables. Under strict inertia, every action that causes a change in alive(x) must also specify the corresponding change in dead(x).

2. In order to avoid configuration-dependent action laws. Suppose the scenario world consists of a number of interconnected objects, where a change in one of them may cause changes in neighboring objects, and different scenarios in the same world contain different configurations. If the same action law is going to be used in all scenarios, it must apply regardless of the choice of configuration, and then it is not possible to represent the secondary changes explicitly in the action law.

We shall refer to these as the redundancy reason and the propagation reason for ramification, respectively.