Bidirectional Transformation is Effectful

Faris Abou-Saleh[1], James Cheney[2], Jeremy Gibbons[1], James McKinna[2], and Perdita Stevens[2]

1 **Department of Computer Science, University of Oxford**
http://www.cs.ox.ac.uk/people/firstname.lastname/
2 **School of Informatics, University of Edinburgh**
http://www.inf.ed.ac.uk/people/staff/Firstname_Lastname.html

Abstract

Bidirectional transformations inherently involve state effects. Modelling them that way allows the incorporation of other effects too, such as I/O, non-determinism, and exceptions.

1998 ACM Subject Classification F.3.2 Semantics of Programming Languages—algebraic approaches to semantics; D.3.2 Language Classifications—specialized application languages;

Keywords and phrases monad, model-driven development, view–update, lens

Digital Object Identifier 10.4230/LIPIcs.xxx.yyy.p

1 Background

Bidirectional transformations (bx) are programs that maintain consistency between different information sources. Examples include a database and a view on it; an internal data representation and a user interface to it; and models of different aspects of a system in model-driven software development.

In each case, the problem is to propagate updates in one source to the other sources, in order to restore consistency. The challenge is that the mappings between sources are in general not functional: no source need be a master copy, and so it really is a matter of reconciling the new with the old rather than regeneration from scratch. The long-term goal is the development of bidirectional languages, allowing one to define a single declarative specification of the relationships between sources, which may be interpreted in multiple directions to achieve the necessary consistency-restoring transformations.

The original motivation comes from Bancilhon and Spyratos's 'view–update problem' in databases in the 1980s, but it has seen continued interest through the work of Schürr et al. on graph transformations since the 1990s, and Pierce et al. on lenses since the 2000s.

2 Insight

Given that some sources may contain information that is absent from the others, consistency restoration must necessarily be dependent on sources beyond the immediately changed one. The asymmetric lens approach involves a 'source' type S and a 'view' type V, with a projection function $get :: S \to V$ from source to view and a restoration function $put :: (V \times S) \to S$ from an edited view and an old source to an updated source, subject to certain coherence axioms.

Our observation is that the types of *get* and especially *put* are very reminiscent of the state monad in effectful functional programming; writing M for this monad, one might instead model the types as $get :: M\ V$ and $put :: V \to M\ ()$, reading a V from and writing it to some opaque global store.


© Faris Abou-Saleh, James Cheney, Jeremy Gibbons, James McKinna, and Perdita Stevens;
licensed under Creative Commons License CC-BY
Conference title on which this volume is based on.
Editors: Billy Editor and Bill Editors; pp. 1–2

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

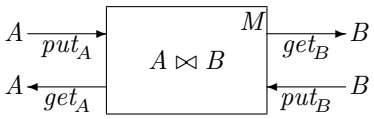

Figure 1 Effectful bx on sources A, B over monad M

This turns out to be a very fruitful insight. One consequence is that it generalizes readily to the symmetric case, which relaxes the requirement for a 'master copy' S: we have one pair of operations $get_A :: M\ A$ and $put_A :: A \to M\ ()$ on source A, and another pair $get_B :: M\ B$ and $put_B :: B \to M\ ()$ on source B, which maintain consistency between the two sources. These are the operations of a two-cell memory, but with an interestingly different equational theory: the different sources A, B are generally not independent, and so operations on distinct memory cells need not commute—one might say that the cells are 'entangled'. (The natural model of the equational theory amounts to the state monad, but rather than being arbitrary $A \times B$ pairs, the state type is the relational join $A \bowtie B$, ie the subset of such pairs 'consistent' under the bx.) See Figure 1.

A second consequence is that it paves the way towards a framework of bxs incorporating other effects, such as I/O (allowing for bxs that query the user or consult an external oracle), non-determinism (allowing for bxs with internal choices), and exceptions (allowing for bxs that may sometimes fail to restore consistency). These have hitherto been ignored, or at best given ad hoc treatments, in the literature, restricting bxs to being purely functional—yet almost all practical tools work in the context of a conventional language with a full complement of effects. We have the theory, standard combinators, and numerous examples all worked out, pending publication. The details are more subtle than one would expect, especially for sequential composition of bxs.

3 Prospects

Our interest in this effectful perspective on bx is as a means to an end, rather than an end in itself: we hope that it can serve as the basis for a unification of many different strands of work in the bx community. In one direction, it is clear that the equational theory can easily be extended from the binary case to multiple sources; this is important in model-driven development, where systems are generated from multiple overlapping high-level models. In a second but related direction, we want to accommodate bx approaches that record not only *that* two models are consistent, but also *how* they are consistent—for example, via a witness structure of links associating elements of one model with those of the other; we conjecture that such witness structures will fall naturally out of generalizing the effectful functional perspective to a dependently typed one. And in a third direction, we are studying the notion of 'least change' in consistency restoration, a major lacuna in existing formalisms; we conjecture that this in turn will depend on some formalisation of witness structures to consistency.

4 Acknowledgements

This work has been conducted within the UK EPSRC project *A Theory of Least Change for Bidirectional Transformations* at Edinburgh (grant EP/K020218/1) and Oxford (EP/K020919/1); we are very grateful for that support. A full paper *Notions of Bidirectional Computation and Entangled State Monads* will appear at *Mathematics of Program Construction 2015*.