An "Infamous" Example

Below we have an initial algebra specification by Goguen et al (chap. 5 of Yeh book) to describe the (signed) integer ADT (with sum and product operations) given Boolean and Nat ADTs. The intuitive idea is to pair a natural number with a Boolean value that represents its sign, and then express signed arithmetic (assuming we already know unsigned arithmetic). However, it will be seen that the initial algebra of this ADT is *not* the usual algebra of the integers.

The specification

Signat signature

Pre-defined types: Boolean (with '='), and Nat ({0, 1, 2, **...**}) with operations as usual, including +, $\dot{-}$, *, ≤ (note that $\dot{-}$ is "proper subtraction", n$\dot{-}$ m yields 0 when n≤m).

Operation signatures:

PAIR: Nat, Bool → Signat

ABS: Signat → Nat

SGN: Signat → Bool

SUM: Signat, Signat → Signat

PROD: Signat, Signat → Signat

Semantic equations (for all s∈ Signat, n∈ Nat, b∈ Bool)

1. PAIR(ABS(s), SGN(s)) = s
2. ABS(PAIR(n,b)) = n
3. SGN(PAIR(n,b)) = b
4. SUM(s_1, s_2) = **if** SGN(s_1)=SGN(s_2)
   then PAIR(ABS(s_1)+ABS(s_2), SGN(s_1))
   else if ABS(s_1)≤ABS(s_2)
   then PAIR(ABS(s_2)$\dot{-}$ABS(s_1), SGN(s_2))
   else PAIR(ABS(s_1)$\dot{-}$ABS(s_2), SGN(s_1))
5. PROD(s_1,s_2) = PAIR(ABS(s_1)*ABS(s_2), SGN(s_1)=SGN(s_2))

The flaw

The fault to be found with this specification of the signed integers is that there are *two* "zeros" — PAIR(0,True) or +0, and PAIR(0,False) or –0. There are no equations that enable us to deduce two different pairs are equivalent, and so in the initial algebra view they are different. This might appear to be a minor oversight, but if fact having two distinct representations of zero causes numerous familiar identities to be invalidated.

For instance, for all x, x+0 = x in the integers. But neither of the corresponding values in this specification has this property — note that SUM(+0,–0) = –0, and SUM(–0,+0) = +0 (use the equations in the specification on the term forms of these values to confirm this). Also x*0 = 0 in the integers, but in the specification PROD(–5,+0) = –0, and PROD(–0,–0) = +0. If it is really the system of signed natural numbers we seek to specify, the specification given does not qualify.

A defective repair

In a widely circulated IBM technical report that preceded publication, the authors "corrected" the problem in the SigNat ADT described above by proposing the single additional axiom

6. PAIR(0,True) = PAIR(0,False)

to unify these two different equivalence classes (i.e., +0 = –0). While at first glance this seems like a simple and obvious solution, it is a huge blunder. If we add this axiom to those we already have for SigNat, then

True $\overset{\equiv}{3}$ SGN(PAIR(0,True)) $\overset{\equiv}{6}$ SGN(PAIR(0,False)) $\overset{\equiv}{3}$ False!

Hence an inconsistency in the pre-defined type Boolean has been introduced into the specification, and the original flawed "approximate specification" has been destroyed rather than repaired.

An actual repair

A suitable correction to the original flaw is to replace equation 3 by

3'. SGN(n,b) = **if** n=0 **then** True **else** b

Then the two representations of zero are equivalent:

PAIR(0,False) $\overset{\equiv}{1}$ PAIR(ABS(PAIR(0,False), SGN(PAIR(0,False)) $\overset{\equiv}{2}$

PAIR(0, SGN(PAIR(0,False))) $\overset{\equiv}{3'}$ PAIR(0, True).

So now the two zeros fall into the same equivalence class, but no inconsistency is introduced.