Homework IX Sample Solutions

Problem 1.

(a) inconsistency (for specificity Item is taken to be Int, the usual integers)

Note that by "errors propagate" INSERT(CREATE, $\text{UNDEFINED}_{\text{Int}}$) = $\text{UNDEFINED}_{\text{CL}}$ and there are undefined elements of all the sorts. Consider the term T = INSERT($\text{UNDEFINED}_{\text{CL}}$, 2). By equation 14, VALUE(T) = 2, but by "errors propagate", T = $\text{UNDEFINED}_{\text{CL}}$ and so VALUE(T) = $\text{UNDEFINED}_{\text{Int}}$. Therefore 2 = $\text{UNDEFINED}_{\text{Int}}$ and the specification is inconsistent.

(b) correcting the specification with exceptions included

The operations CREATE, INSERT, and $\text{UNDEFINED}_{\text{CL}}$ may be taken as constructors. Operations therefore are described by their behavior in each of these three cases. First add the operation OK: Circular LIst -> Boolean, with equations (not errors propagate)

OK(CREATE) = true
OK($\text{UNDEFINED}_{\text{CL}}$) = false
OK(INSERT(c,i)) = OK(c) $\wedge$ OK(i)

The "errors propagate" equations are assumed for all the operations except OK without explicitly writing them.

Next, change the given equations into conditional equations as follows:

10. ISEMPTY(INSERT(c,i)) = **if** OK(c) $\wedge$ OK(i) **then false else** $\text{UNDEFINED}_{\text{Bool}}$.
12. DELETE(INSERT(c,i)) = **if** OK(c) $\wedge$ OK(i) **then** c **else** $\text{UNDEFINED}_{\text{CL}}$.
14. VALUE(INSERT(c,i)) = **if** OK(c) $\wedge$ OK(i) **then** i **else** $\text{UNDEFINED}_{\text{Int}}$.
17. RIGHT(INSERT(INSERT(c,i), i1)) = **if** OK(c) $\wedge$ OK(i) $\wedge$ OK(i1)
    then INSERT(RIGHT(INSERT(c,i1)), i)
    else $\text{UNDEFINED}_{\text{CL}}$.

19, JOIN(c, INSERT(c1,i)) = **if** OK(c) $\wedge$ OK(c1) $\wedge$ OK(i)
    then INSERT(JOIN(c, c1), i)
    else $\text{UNDEFINED}_{\text{CL}}$.

Now it remains to argue that this new specification is consistent and sufficiently complete. For sufficient completeness, we first note that in any term whose outermost operation is either RIGHT or JOIN, that operation can be removed by repeatedly moving the operation inward using its equation with respect to INSERT, until either CREATE OR UNDEFINED is encountered and then the operation is eliminated by one of the other equations. Therefore, we need only consider applying selectors to constructor terms. And the equations for applying either of the selectors ISEMPTY or VALUE to a constructor term either reduces to applying it to a simpler term or immediately eliminates the selector operation. Hence in all cases, a term involving only operations in one of the pre-defined sorts is obtained and the specification is sufficiently complete. For instance,

VALUE(JOIN(INSERT(CREATE,1), INSERT(CREATE,1+1))) =
VALUE(INSERT(JOIN(INSERT(CREATE,1), CREATE), 1+1)) =
VALUE(INSERT(INSERT(CREATE,1),1+1))) = 1+1

Lastly, for consistency, it would be very helpful to use an animation to test a large variety of terms to see that expected results are obtained, and this is the recommended approach. Analytically, the intention of changing the key equations to conditional form blocks multiple rewritings of terms such as VALUE(INSERT(UNDEFINED$_{CL}$,2)) (i.e., can't get result 2) and ensures consistency.

The analysis considers the TOI equivalence classes which consist of
 [CREATE],
 [UNDEFINED$_{CL}$], and
 [INSERT(...INSERT(CREATE,i1),i2),...in)] for OK integers i1, i2, … , in.
Terms in these classes are disjoint from every other class (e.g., INSERT(UNDEFINED$_{CL}$,2) is in the UNDEFINED$_{CL}$ class and is not equivalent to terms in any other classes. Selectors VALUE and ISEMPTY provide the expected results on terms in each of these classes.