Equality and Rewriting Variations

We have already explored CafeOBJ's use of the rewriting relation $\overset{*}{\Rightarrow}$ in place of equality, and some of the considerations that arise. CafeOBJ provides three other alternatives for tem relationships (plus another ':is' for sorts). These relations are:

- _==_ for equality.
- _==>_ for transitions, and
- _=*=_, _=b=_ for behavioral equivalence (i.e., indistinguishability).

Equality Predicate

The equality predicate (==) is a vital operation of the system. It can be reliably used to test the equality of terms of the visible sorts when the ⇒ relation is confluent and terminating.

Transition Predicate

The transition predicate (==>) is an "oriented" version of equality for visible sorts. A transition relation is reflexive and transitive, but the symmetric property of equality (X=Y implies Y=X) is omitted. Transitions are regarded as (normally) irreversible changes. However, the relation ==> can be regarded as being defined by means of the following scheme of equality rules:

for each visible sort S	eq X:S ==> X = true
for trans T => T1	eq T ==> T1 = true
for ctrans T => T1 if C	ceq T ==> T1 = true if C

and for each

op f : S1 … Sn -> S	ceq f(X1:S1, … , Xn:Sn) ==> f(Y1:S1, … , Yn:Sn) = true if X1 ==> Y1 and … and Xn ==> Yn.

This omits the transitive property. The direct way to express transitivity would be

    ceq X:S ==> X1:S = true if X ==> Y:S and Y ==> X1.

However, such a rule involves a variable in the condition that does not appear in the left-hand side of the rule and is therefore prohibited. So instead CafeOBJ defines the operator "_=(*)=>_" that is defined to mean a transition in an arbitrary number of steps, and the transitive property becomes the rule

    ceq X:S ==> Y:S = true if X =(*)=> Y.

Behavioral Equivalence Predicate

Lastly, behavioral equivalence (=*=) is defined for each hidden sort. Recall that behavioral operators (declared with **bop**) have exactly one argument with a hidden sort. The implication that two values of hidden sort H are indistinguishable is not fully captured by CafeOBJ's behavioral equivalence. The system uses only selectors (attributes as CafeOBJ calls them) with a single argument, f : H -> P, and if these are f1, f2, … fn, then

    eq X:H =*= Y:H = f1(X) == f1(Y) and … and fn(X) == fn(Y).

Evaluation alternatives

There are three evaluation commands in CafeOBJ. They differ in the rules they employ during reduction, and in where they use them. In particular

- reduce and breduce use only equations, transitions are excluded,
- execute uses all the rules, and
- reduce and execute use behavioral rules only on subexpressions of a selector operation, while breduce uses them on all subexpressions.