Mixed induction-coinduction at work for Coq

Keiko Nakata, Tarmo Uustalu
Institute of Cybernetics at Tallinn University of Technology

We present tricks to implement mixed induction-coinduction in Coq using Mendler-style coinduction. We demonstrate the tricks by formalizing termination-sensitive weak bisimilarity on interactive input-output resumptions with explicit internal actions (delays). Intuitively a resumption represents the denotation of a reactive program as a tree[1]. The tree branches on inputs, each edge corresponding to each possible input, and has infinitely deep paths if the program may diverge. Formally *resumptions* $r : res$ are defined coinductively by the rules[2]

$$\begin{array}{c} \sigma : state \\ \hline\hline ret\ \sigma : res \end{array} \quad \begin{array}{c} f : Int \to\ res \\ \hline\hline in\ f : res \end{array} \quad \begin{array}{c} v : Int \quad r : res \\ \hline\hline out\ v\ r : res \end{array} \quad \begin{array}{c} r : res \\ \hline\hline \delta\ r : res \end{array}$$

so a resumption either has terminated with some final state, $ret\ \sigma$, takes an integer input v and evolves into a new resumption $f\ v$, $in\ f$, outputs an integer v and evolves into r, $out\ v\ r$, or performs an internal action (observable at best as a delay) and becomes r, $\delta\ r$.

We also define *(strong) bisimilarity* of two resumptions, $r \approx r_*$, coinductively by

$$\begin{array}{c} \\ \hline\hline ret\ \sigma \approx ret\ \sigma \end{array} \quad \begin{array}{c} \forall v.\, f\ v \approx f_*\ v \\ \hline\hline in\ f \approx in\ f_* \end{array} \quad \begin{array}{c} r \approx r_* \\ \hline\hline out\ v\ r \approx out\ v\ r_* \end{array} \quad \begin{array}{c} r \approx r_* \\ \hline\hline \delta\ r \approx \delta\ r_* \end{array}$$

Bisimilarity is straightforwardly seen to be an equivalence.

Two resumptions are *weakly bisimilar*, $r \cong^\circ r_*$, if they are bisimilar modulo collapsing finite sequences of delay steps between observable actions. Weak bisimilarity allows us to talk about observable behavior disregarding finite delays. For instance, to guarantee correctness of a compiler optimization, we may want to prove that the optimization does not change the observable behavior of the source program, including termination and divergence behaviors, but the optimized code may perform fewer internal steps and thus be faster. We therefore formalize *termination-sensitive* weak bisimilarity, which distinguishes termination and silent divergence.

Technically, getting the definition of weak bisimilarity right is not straightforward, especially not in a constructive setting. It requires both induction and coinduction: we need to collapse a *finite* number of delay steps between observable actions possibly *infinitely*. To do so, we first define $\downarrow X \downarrow$ inductively in terms of X, for any setoid relation X with bisimilarity as the equivalence relation. We then define $\cong^\circ$ coinductively in terms of $\downarrow \cong^\circ \downarrow$. For binary relations X, Y, $X \subseteq Y$ denotes $\forall x, x_*.\, x\, X\, x_* \to x\, Y\, x_*$.

$$\frac{}{ret\ \sigma \downarrow X \downarrow ret\ \sigma} \quad \frac{r\ X\ r_*}{out\ v\ r \downarrow X \downarrow out\ v\ r_*} \quad \frac{\forall v.\, f\ v\ X\ f'\ v}{in\ f \downarrow X \downarrow in\ f'} \quad \frac{r \downarrow X \downarrow r_*}{\delta\ r \downarrow X \downarrow r_*} \quad \frac{r \downarrow X \downarrow r_*}{r \downarrow X \downarrow \delta\ r_*}$$

$$\begin{array}{c} X \subseteq \cong^\circ \quad r \downarrow X \downarrow r_* \\ \hline\hline r \cong^\circ r_* \end{array} \quad \begin{array}{c} r \cong^\circ r_* \\ \hline\hline \delta\ r \cong^\circ \delta\ r_* \end{array}$$

Intuitively, $r \downarrow X \downarrow r_*$ means that r and r_* converge to resumptions related by X.

In the first rule of $\cong^\circ$, we have used Mendler-style coinduction to make the definition of $\cong^\circ$ strictly positive, in order to enable Coq's guarded corecursion for $\cong^\circ$ (the guardedness condition for induction nested into coinduction is otherwise too weak). The natural (Park-style) rule to stipulate would have been:

$$\begin{array}{c} r \downarrow \cong^\circ \downarrow r_* \\ \hline\hline r \cong^\circ r_* \end{array}$$

[1] Our previous paper available at authors' homepage defines operational semantics of reactive While in terms of resumptions.

[2] We mark inductive definitions by single horizontal rules and coinductive definitions by double horizontal rules.

With our definition, it is derivable. We can also prove that $\downarrow X \downarrow$ is monotone in X, which allows us to recover the natural inversion principle for $\cong^{\circ}$.

We note that induction simultaneously with coinduction does not make sense, if the recursive and corecursive occurrences are covariant, as they are here[3]. So, we must have an inductive definition nested into a coinductive definition[4], or vice versa. Here, we need the former, since we want finite chunks of $\downarrow \cong^{\circ} \downarrow$ derivations to be weaved into an infinite $\cong^{\circ}$ derivation. In the current design of Agda, this form of mixing induction and coinduction is the basic form in which induction and coinduction are supported.

Weak bisimilarity is a setoid predicate and an equivalence relation. Reflexivity and symmetry are straightforward to prove by coinduction. Transitivity is more subtle and we will sketch the proof below. For binary relations X, Y, $X \circ Y$ denotes their composition. Namely, $x\,(X \circ Y)\,x'$ if there is x'' such that $x\,X\,x''$ and $x''\,Y\,x'$.

We first prove by *induction* transitivity for $\downarrow X \downarrow$.

Lemma 1. *For any resumptions r_0, r_1, r_2 and setoid relations X, Y, if $r_0 \downarrow X \downarrow r_1$ and $r_1 \downarrow Y \downarrow r_2$, then $r_0 \downarrow (X \circ Y) \downarrow r_2$.*

Lemma 2. *For any resumptions r_0, r_1 and r_2, if $r_0 \cong^{\circ} r_1$ and $r_1 \cong^{\circ} r_2$, then $r_0 \cong^{\circ} r_2$.*

Proof. By coinduction and inversion on $r_0 \cong^{\circ} r_1$ and $r_1 \cong^{\circ} r_2$. We show the main case. Suppose we have $r_0 \cong^{\circ} r_1$ and $r_1 \cong^{\circ} r_2$ because $r_0 \downarrow X \downarrow r_1$ and $r_1 \downarrow Y \downarrow r_2$ for some X and Y such that $X \subseteq \cong^{\circ}$ and $Y \subseteq \cong^{\circ}$. By Lemma 1, $r_0 \downarrow X \circ Y \downarrow r_2$. Using coinduction hypothesis, we prove $X \circ Y \subseteq \cong^{\circ}$, which closes the case. □

Mendler-style coinduction for weak bisimilarity nicely works here: the use of coinduction hypothesis to prove $X \circ Y \subseteq \cong^{\circ}$ is properly guarded. It also requires impredicativity: our relations on resumptions, including $\downarrow X \downarrow$ and $\cong^{\circ}$, must be *Prop*-valued.

We have given a general technique for mixing induction-coinduction in Coq, by nesting induction into coinduction: the inductive definition is parameterized and the coinductive definition is converted into Mendler's format. We have explained the technique by formalizing a practically useful and technically interesting notion of weak bisimilarity. The technique works nicely with Coq's syntactic guardedness approach to ensure productivity of coinduction. It also scales well to more complex examples: in our previous work, we have used the technique to formalize a delay-free big-step operational semantics for responsive programs (a program is responsive if it *always eventually* performs input or output unless it terminates) and related it to a basic delayful semantics.

Acknowledgements This research was supported by the Estonian Centre of Excellence in Computer Science, EXCS, funded by the European Regional Development Fund, and the Estonian Science Foundation grant no. 6940.

[3] It is meaningful to look for a least X and greatest Y solving a system of equations $X = F(Y, X)$, $Y = G(X, Y)$, if F and G are contravariant in their first arguments and covariant in the second arguments.

[4] a definition of the form $\nu X.\,\mu Y.\,F(X,Y)$ with F covariant in both arguments