Antifounded corecursion and coinduction in type theory

Tarmo Uustalu, Institute of Cybernetics, Tallinn

PAR 2010, Edinburgh, 15 July 2010

When does a structured recursion diagram define a function?

We are interested defining (= definitely desribing) a function $f : A \to B$ by an equation of the form:

$$\begin{tikzcd} FA \arrow[d, "Ff"'] & A \arrow[l, "\alpha"'] \arrow[d, "f"] \\ FB \arrow[r, "\beta"'] & B \end{tikzcd}$$

F – branching type of recursive call [corecursive return] trees (an endofunctor)
α – marshals arguments for recursive calls (an F-coalgebra)
β – collects recursive call results (an F-algebra)

Some good cases (1): Initial algebra

$$\begin{tikzcd}
1 + \mathsf{El} \times \mathsf{List} \arrow[d, "1+\mathsf{El}\times f"'] & & \mathsf{List} \arrow[ll, "{[\mathsf{nil}, cons]^{-1}}"'] \arrow[d, "f"] \\
1 + \mathsf{El} \times B \arrow[rr, "\beta"'] & & B
\end{tikzcd}$$

E.g., for $B = \mathsf{List}$, $\beta = \mathsf{ins}$, we get $f = \mathsf{isort}$.

A unique f exists for any (B, β) because $(\mathsf{List}, [\mathsf{nil}, \mathsf{cons}])$ is the *initial algebra* of $1 + \mathsf{El} \times (-)$.

f is the fold of (B, β).

Some good cases (2): Recursive coalgebras

$$\begin{tikzcd}
1 + \mathsf{List} \times \mathsf{El} \times \mathsf{List} \arrow[d, "1+f\times\mathsf{El}\times f"'] & & \mathsf{List} \arrow[ll, "\mathsf{qsplit}"'] \arrow[d, "f"] \\
1 + B \times \mathsf{El} \times B \arrow[rr, "\beta"'] & & B
\end{tikzcd}$$

$\mathsf{qsplit}\,\mathsf{nil} = \mathsf{inl}\,*; \quad \mathsf{qsplit}\,(\mathsf{cons}\,(x, xs)) = \mathsf{inr}\,(xs|_{\leq x}, x, xs|_{>x})$

E.g., for $B = \mathsf{List}$, $\beta = \mathsf{app} \circ (\mathsf{List} \times \mathsf{cons})$, we get $f = \mathsf{qsort}$.

We have a unique solution f for any (B, β). We say that $(\mathsf{List}, \mathsf{qsplit})$ is a *recursive coalgebra* of $1 + (-) \times \mathsf{El} \times (-)$.

[The inverse of the initial F-algebra is the final recursive F-coalgebra.]

Some good cases (3): Final coalgebra

$$\begin{tikzcd}
\mathsf{El} \times A \arrow[d, "1+\mathsf{El}\times f"'] & & A \arrow[ll, "\alpha"'] \arrow[d, "f"] \\
\mathsf{El} \times \mathsf{Str} \arrow[rr, "\langle \mathsf{hd},\mathsf{tl} \rangle^{-1}"'] & & \mathsf{Str}
\end{tikzcd}$$

E.g., for $A = \mathsf{Str}$, $\alpha = \langle \mathsf{hd}, \mathsf{tl} \circ \mathsf{tl} \rangle$, we get $f = \mathsf{dropeven}$.

A unique f exists for any (A, α) because $(\mathsf{Str}, \langle \mathsf{hd}, \mathsf{tl} \rangle)$ is the *final coalgebra* of $\mathsf{El} \times (-)$.

f is the unfold of (A, α).

Some good cases (4): Corecursive algebras

$$\begin{tikzcd}
A \times \mathsf{El} \times A \arrow[d, "f \times \mathsf{El} \times f"'] & & A \arrow[ll, "\alpha"'] \arrow[d, "f"] \\
\mathsf{Str} \times \mathsf{El} \times \mathsf{Str} \arrow[rr, "\textcolor{red}{\mathsf{smerge}}"'] & & \mathsf{Str}
\end{tikzcd}$$

$$\begin{aligned}
&\mathsf{hd}\,(\mathsf{smerge}(xs_0, x, xs_1)) = x \\
&\mathsf{tl}\,(\mathsf{smerge}(xs_0, x, xs_1)) = \mathsf{smerge}(xs_1, \mathsf{hd}\,xs_0, \mathsf{tl}\,xs_1)
\end{aligned}$$

We have a unique f for any (A, α). We therefore call $(\mathsf{Str}, \mathsf{smerge})$ a *corecursive algebra* of $\mathsf{El} \times (-) \times (-)$.

[The inverse of the final F-coalgebra is the initial corecursive F-algebra.]

General case (1): Inductive domain predicate

Bove-Capretta

For given (A, α), define a predicate dom on A inductively by

$$\frac{a : A \qquad (\tilde{F}\,\mathrm{dom})\,(\alpha\, a)}{\mathrm{dom}\, a}$$

For any (B, β), there is $f : A|_{\mathrm{dom}} \to B$ uniquely solving

$$\begin{tikzcd}
F(A|_{\mathrm{dom}}) \arrow[d, "Ff"'] & A|_{\mathrm{dom}} \arrow[l, "\alpha|_{\mathrm{dom}}"'] \arrow[d, "f"] \\
FB \arrow[r, "\beta"'] & B
\end{tikzcd}$$

If $\forall a : A.\, \mathrm{dom}\, a$, which is the same as $A|_{\mathrm{dom}} \cong A$, then f is unique solution of the original equation.

Wellfounded induction

If $A|_{\mathrm{dom}} \cong A$, then a coalgebra (A, α) is said to be *wellfounded*.

Wellfoundedness gives an induction principle on A: For any predicate P on A, we have

$$\frac{a : A \qquad \begin{array}{c} a' : A \quad (\tilde{F}\,P)\,(\alpha\,a') \\ \vdots \\ P\,a \end{array}}{P\,a}$$

We have seen that wellfoundedness suffices for recursiveness. In fact, it is also necessary.

General case (2): Inductive graph relation

Bove

For given (A, α), (B, β), define a relation $\downarrow$ between A, B inductively by

$$\frac{a : A \quad bs : FB \quad (\alpha\, a)\ (\tilde{F} \downarrow)\ bs}{a \downarrow (\beta\, bs)}$$

Further, define a predicate Dom on A by $\mathrm{Dom}\, a = \exists b : B.\, a \downarrow b$.
It turns out that $\forall a : A.\, \mathrm{Dom}\, a \leftrightarrow \mathrm{dom}\, a$.
So again, there is $f : A|_{\mathrm{Dom}} \to B$ uniquely solving

$$\begin{tikzcd}
F(A|_{\mathrm{Dom}}) \arrow[d, "Ff"'] & A|_{\mathrm{Dom}} \arrow[l, "\alpha|_{\mathrm{Dom}}"'] \arrow[d, "f"] \\
FB \arrow[r, "\beta"'] & B
\end{tikzcd}$$

And, if $\forall a : A.\, \mathrm{Dom}\, a$, which is the same as $A|_{\mathrm{Dom}} \cong A$, then f is a unique solution of the original equation.

Trouble: Inductive domain/graph don't work for corecursion

Unfortunately, for our dropeven example,

$$\begin{tikzcd} \mathsf{El} \times \mathsf{Str} \arrow[d, "1+\mathsf{El}\times\mathsf{dropeven}"'] & & \mathsf{Str} \arrow[ll, "\langle \mathsf{hd}, \mathsf{tl}\circ\mathsf{tl} \rangle"'] \arrow[d, "\mathsf{dropeven}"] \\ \mathsf{El} \times \mathsf{Str} \arrow[rr, "{\color{red}\langle \mathsf{hd}, \mathsf{tl} \rangle^{-1}}"'] & & \mathsf{Str} \end{tikzcd}$$

we get dom $\cong 0$!

Now, surely there is a unique function from $0 \rightarrow \mathsf{Str}$. But this is uninteresting!

???

General case (3): Coinductive bisimilarity relation

Capretta, Uustalu, Vene

For given (B, β), define a relation $\approx$ on B coinductively by

$$\frac{bs, bs' : FB \quad \beta\, bs \approx \beta\, bs'}{bs\ (\tilde{F} \approx^*)\ bs'}$$

If $\forall b, b' : B.\, b \approx b' \to b = b'$, which is the same as $B/_{\approx^*} \cong B$, we say that it is *antifounded*.

This does not suffice for existence of f satisfying

$$\begin{tikzcd} FA \arrow[d, "Ff"'] & A \arrow[l, "\alpha"'] \arrow[d, "f"] \\ F(B/_{\approx^*})/ \arrow[r, "\beta/_{\approx^*}"'] & B/_{\approx^*} \end{tikzcd}$$

but it suffices for uniqueness!

Antifounded coinduction

We saw that antifoundedness of (B, β) does not suffice for corecursion from A to B for any (A, α).

The converse also fails: not every corecursive algebra (B, β) is antifounded.

However, for an antifounded algebra (B, β), we do get an interesting coinduction principle on B: For any relation R on B, we have

$$\dfrac{b, b' : B \quad b\,R\,b' \quad \begin{array}{c} bs, bs' : FB \quad (\beta\,bs)\,R\,(\beta\,bs') \\ \vdots \\ bs\,(\tilde{F}\,R^*)\,bs' \end{array}}{b = b'}$$

General case (4): Coinductive graph relation

For given (A, α), (B, β), define a relation $\downarrow^\infty$ between A, B coinductively by

$$\frac{a : A \quad bs : FB \quad a \downarrow^\infty (\beta\, bs)}{(\alpha\, a)\ (\tilde{F} \downarrow^\infty)\ bs}$$

Define a predicate Dom^∞ on A by $\mathrm{Dom}^\infty a = \exists b : B.\, a \downarrow^\infty b$ and a relation $\equiv$ on B by $b \equiv b' = \exists a : A.\, a \downarrow^\infty b \wedge a \downarrow^\infty b'$.
Now we have $f : A|_{\mathrm{Dom}^\infty} \to B/_{\equiv^*}$ uniquely solving

$$\begin{tikzcd}
F(A|_{\mathrm{Dom}^\infty}) \arrow[d, "Ff"'] & A|_{\mathrm{Dom}^\infty} \arrow[l, "\alpha|_{\mathrm{Dom}^\infty}"'] \arrow[d, "f"] \\
F(B/_{\equiv^*})/ \arrow[r, "\beta/_{\equiv^*}"'] & B/_{\equiv^*}
\end{tikzcd}$$

If $\forall a : A.\, \mathrm{Dom}^\infty\, a$ and $\forall b, b' : B.\, b \equiv b' \to b = b'$, so that $A|_{\mathrm{Dom}^\infty} \cong A$ and $B/_{\equiv^*} \cong B$, then f uniquely solves the original equation.

Conclusion

There are two kinds of partiality: some arguments may be not in the domain, some values not crisp.

Bove-Capretta method extends to recursive equations where unique solvability is not due to termination, but productivity or a combination.

Instead of one condition to check by ad-hoc means, there are two in the general case.

The theory of corecursion/coinduction is not as simple and clean as that of recursion/induction — admitting coinduction is different from admitting corecursion.