Antifounded corecursion

Tarmo Uustalu, Institute of Cybernetics, Tallinn

Theory Days, Tõrve, 7–9 October 2011

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

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

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

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

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 $f = \mathsf{qsort}$.

$(\mathsf{List}, \mathsf{qsplit})$ is <u>not</u> the inverse of the initial algebra of $1 + (-) \times \mathsf{El} \times (-)$, but we still have a unique 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

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

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

$$\begin{tikzcd}
A \times \mathrm{El} \times A \arrow[d, "f \times \mathrm{El} \times f"'] & A \arrow[l, "\alpha"'] \arrow[d, "f"] \\
\mathrm{Str} \times \mathrm{El} \times \mathrm{Str} \arrow[r, "\mathrm{smerge}"'] & \mathrm{Str}
\end{tikzcd}$$

$$\mathrm{hd}\,(\mathrm{smerge}(xs_0, x, xs_1)) = x$$
$$\mathrm{tl}\,(\mathrm{smerge}(xs_0, x, xs_1)) = \mathrm{smerge}(xs_1, \mathrm{hd}\,xs_0, \mathrm{tl}\,xs_1)$$

$(\mathrm{Str}, \mathrm{smerge})$ is not the inverse of the final coalgebra of $(-) \times \mathrm{El} \times (-)$, but a unique f still exists for any (A, α).

We say that $(\mathrm{Str}, \mathrm{smerge})$ is a *corecursive algebra* of $(-) \times \mathrm{El} \times (-)$.

[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

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

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 a unique solution of the original equation.

General case (1): Inductive domain predicate ctd

For $A = \mathsf{List}$, $\alpha = \mathsf{qsplit}$, dom is defined inductively by

$$\frac{}{\mathsf{dom}\,\mathsf{nil}} \qquad \frac{x : \mathsf{El} \quad xs : \mathsf{List} \quad \mathsf{dom}\,(xs|_{\leq x}) \quad \mathsf{dom}\,(xs|_{>x})}{\mathsf{dom}\,(\mathsf{cons}\,(x, xs))}$$

We can prove that $\forall xs : \mathsf{List}.\, \mathsf{dom}\, xs$.

Hence $(\mathsf{List}, \mathsf{qsplit})$ is recursive.

Wellfounded induction

If $A|_{\mathsf{dom}} \cong A$, the 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.

Wellfounded induction ctd

For $A = \mathsf{List}$, $\alpha = \mathsf{qsplit}$, we get this induction principle:

$$\dfrac{xs : \mathsf{List} \quad P\,\mathsf{nil} \quad \begin{array}{c} x : \mathsf{El} \quad xs' : \mathsf{List} \quad P\,(xs'|_{\leq x}) \quad P\,(xs'|_{>x}) \\ \vdots \\ P\,(\mathsf{cons}\,(x, xs')) \end{array}}{P\,xs}$$

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 is easy that $\forall a : A, b, b' : B . a \downarrow b \wedge a \downarrow b' \to b = b'$.

Moreover, $\forall a : A.\, \mathrm{Dom}\, a \leftrightarrow \mathrm{dom}\, a$.

So, Dom does not really depend on the given (B, β)!

General case (2): Inductive graph relation ctd

We know that there is $f : A|_{\mathsf{Dom}} \to B$ uniquely solving

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

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

As a matter of fact, recursiveness and wellfoundedness are equivalent exactly because $\forall a : A.\, \mathsf{Dom}\, a \leftrightarrow \mathsf{dom}\, a$.

General case (2): Inductive graph relation ctd

For $A = \mathsf{List}$, $\alpha = \mathsf{qsplit}$, $B = \mathsf{List}$, $\beta = \mathsf{app} \circ (\mathsf{List} \times \mathsf{cons})$, the relation $\downarrow$ is defined inductively by

$$\frac{}{\mathsf{nil} \downarrow \mathsf{nil}} \qquad \frac{x : \mathsf{El} \quad xs : \mathsf{List} \quad xs|_{\leq x} \downarrow ys_0 \quad xs|_{>x} \downarrow ys_1}{\mathsf{cons}\,(x, xs) \downarrow \mathsf{app}\,(ys_0, \mathsf{cons}(x, ys_1))}$$

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

Unfortunately, for our dropeven example,

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

we get $\mathsf{dom} \cong 0$!

Now, surely there is a unique function from $0 \to \mathsf{Str}$. 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 (B, β) 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!

General case (3): Coinductive bisimilarity relation

For $B = \mathrm{Str}$, $\beta = \mathrm{qmerge}$, the relation $\approx$ is defined coinductively by

$$\frac{\begin{array}{ll} xs_0, xs_1 : \mathrm{Str}, & \mathrm{smerge}(xs_0, x, xs_1) \\ \quad x, x' : \mathrm{El}, xs_0', xs_1' : \mathrm{Str} & \quad \approx \mathrm{smerge}(xs_0', x', xs_1') \end{array}}{xs_0 \approx^* xs_0' \wedge x = x' \wedge xs_1 \approx^* xs_1'}$$

It turns out that $\forall xs, xs' : \mathrm{Str}.\, xs \approx xs' \to xs = xs'$.

Based on this knowledge, we know that solutions are unique, but need not exist.

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

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

Antifounded coinduction ctd

For $B = \mathsf{Str}$, $\beta = \mathsf{qmerge}$, we get this coinduction principle:

$$\dfrac{xs, xs' : \mathsf{Str} \quad xs \mathrel{R} xs' \quad \begin{array}{c} \begin{array}{ll} xs_0, xs_1 : \mathsf{Str}, & \mathsf{smerge}(xs_0, x, xs_1) \\ \quad x, x' : \mathsf{El}, xs_0', xs_1' : \mathsf{Str} & \quad R\,\mathsf{smerge}(xs_0', x', xs_1') \end{array} \\ \vdots \\ xs_0 \mathrel{R^*} xs_0' \wedge x = x' \wedge xs_1 \mathrel{R^*} xs_1' \end{array}}{xs = xs'}$$

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

$$b \equiv b' = \exists a : A.\, a \downarrow^\infty b \wedge a \downarrow^\infty b'$$

General case (4): Coinductive graph relation ctd

Now we have $f : A|_{\mathsf{Dom}^\infty} \to B/_{\equiv^*}$ uniquely solving

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

If both $\forall a : A.\, \mathsf{Dom}^\infty\, a$ and $\forall b, b' : B.\, b \equiv b' \to b = b'$,
which are the same as $A|_{\mathsf{Dom}^\infty} \cong A$ resp. $B/_{\equiv^*} \cong B$,
then f uniquely solves the original equation.

Notice, however, that we get a unique solution only for our given (A, α): We have <u>not</u> obtained that (B, β) is corecursive.

General case (4): Coinductive graph relation ctd

For $B = \mathsf{Str}$, $\beta = \mathsf{smerge}$ and any fixed A, α, the relation $\downarrow^\infty$ is defined coinductively by

$$\frac{a : A \quad xs_0 : \mathsf{Str}, x : \mathsf{El}, xs_1 : \mathsf{Str} \quad a \downarrow^\infty \mathsf{smerge}\,(xs_0, x, xs_1)}{\mathsf{fst}\, a \downarrow^\infty xs_0 \wedge \mathsf{fst}\,(\mathsf{snd}\, a) = x \wedge \mathsf{snd}\,(\mathsf{snd}\, a) \downarrow^\infty xs_1}$$

It turns out that $\forall a : A.\, \mathsf{Dom}^\infty\, a$ and $\forall xs, xs' : \mathsf{Str}.\, xs \equiv xs' \to xs = xs'$ no matter what A, α are.

So in this case we do have a unique solution f for any A, α, i.e., $(\mathsf{Str}, \mathsf{smerge})$ is corecursive.

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.