Schemata for Proofs by Coinduction

Anton Setzer
Swansea University
With contributions by Andreas Abel, Ulrich Berger,
Peter Hancock, Brigitte Pientka, David Thibodeau

Bergerfest and PCC, LMU Munich, 5 May 2016

Happy Birthday

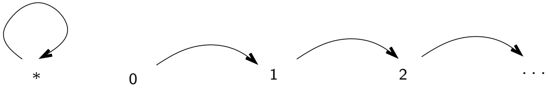
(Co)Iteration – (Co)Recursion – (Co)Induction

Schemata for Corecursive Definitions and Coinductive Proofs

$\mathbb{N}^\infty$, CoEven, CoOdd

Desired Coinductive Proof

- We want to have coinductive proof which are similar to inductive proofs
- Consider an unlabelled Transition system:



- A proof of $\forall n \in \mathbb{N}. * \sim n$ by coinduction could be as follows:
  - We show $\forall n \in \mathbb{N}. * \sim n$ by coinduction on $\sim$.
    - Assume $* \longrightarrow x$. We need to find y s.t. $n \longrightarrow y$ and $x \sim y$. Choose $y = n + 1$. By **co-IH** $* \sim n + 1$.
    - Assume $n \longrightarrow y$. We need to find x s.t. $* \longrightarrow x$ and $x \sim y$. Choose $x = *$. By **co-IH** $* \sim n + 1$.
- In essence same proof, but hopefully easier to teach and use.

Introduction/Elimination of Inductive/Coinductive Sets

- Introduction rules for the **inductive set** of natural numbers means that we have

$$0 \in \mathbb{N}$$
$$\mathrm{S} : \mathbb{N} \to \mathbb{N}$$

so we have an $\mathbb{N}$-algebra

$$(\mathbb{N}, 0, \mathrm{S}) \in (X \in \mathrm{Set}) \times X \times (X \to X)$$

- Dually, **coinductive sets** are given by their elimination rules i.e. by **observations** or **eliminators**.
  As an example we consider Stream:

$$\begin{array}{lcl} \mathrm{head} & : & \mathrm{Stream} \to \mathbb{N} \\ \mathrm{tail} & : & \mathrm{Stream} \to \mathrm{Stream} \end{array}$$

  We obtain a Stream-coalgebra

$$(\mathrm{Stream}, \mathrm{head}, \mathrm{tail}) \in (X \in \mathrm{Set}) \times (X \to \mathbb{N}) \times (X \to X)$$

Unique Iteration

- That $(\mathbb{N}, 0, \mathrm{S})$ are minimal can be given by:
  - Assume another $\mathbb{N}$-algebra (X, z, s), i.e.

$$\begin{array}{l} z \in X \\ s : X \to X \end{array}$$

  - Then there exist a **unique homomorphism** $g : (\mathbb{N}, 0, \mathrm{S}) \to (X, z, s)$, i.e.

$$\begin{array}{lcl} g : \mathbb{N} \to X & & \\ g(0) & = & z \\ g(\mathrm{S}(n)) & = & s(g(n)) \end{array}$$

  - This is the same as saying $\mathbb{N}$ is an initial $\mathrm{F}_{\mathbb{N}}$-algebra.
  - This means we can define uniquely

$$\begin{array}{lcll} g : \mathbb{N} \to X & & & \\ g(0) & = & x & \text{for some } x \in X \\ g(\mathrm{S}(n)) & = & x' & \text{for some } x' \in X \text{ depending on } g(n) \end{array}$$

  - This is the principle of **unique iteration**.
  - Definition by **pattern matching**.

Unique Coiteration

- Dually, that $(\mathrm{Stream}, \mathrm{head}, \mathrm{tail})$ is maximal can be given by:
  - Assume another Stream-coalgebra (X, h, t):

$$\begin{array}{lcl} h & : & X \to \mathbb{N} \\ t & : & X \to X \end{array}$$

  - Then there exist a **unique homomorphism** $g : (X, h, t) \to (\mathrm{Stream}, \mathrm{head}, \mathrm{tail})$, i.e.:

$$\begin{array}{lcl} \multicolumn{3}{l}{g : X \to \mathrm{Stream}} \\ \mathrm{head}(g(x)) & = & h(x) \\ \mathrm{tail}(g(x)) & = & g(t(x)) \end{array}$$

- Means we can define uniquely

$$\begin{array}{lcll} \multicolumn{3}{l}{g : X \to \mathrm{Stream}} & \\ \mathrm{head}(g(x)) & = & n & \text{for some } n \in \mathbb{N} \text{ depending on } x \\ \mathrm{tail}(g(x)) & = & g(x') & \text{for some } x' \in X \text{ depending on } x \end{array}$$

  This is the principle of **unique coiteration**.
- Definition by **copattern matching**.

Unique Primitive (Co)Recursion

- From unique iteration for $\mathbb{N}$ we can derive the principle of **unique primitive recursion**:
  - We can define uniquely

$$
\begin{array}{lcll}
g : \mathbb{N} \to X & & & \\
g(0) & = & x & \text{for some } x \in X \\
g(\mathrm{S}(n)) & = & x' & \text{for some } x' \in X \text{ depending on } n,\ g(n)
\end{array}
$$

- From unique coiteration we can derive the principle of **unique primitive corecursion**:
  - We can define uniquely

$$
\begin{array}{lcl}
g : X \to \mathrm{Stream} & & \\
\mathrm{head}(g(x)) & = & n \text{ for some } n \in \mathbb{N} \text{ depending on } x \\
\mathrm{tail}(g(x))) & = & g(x') \text{ for some } x' \in X \text{ depending on } x \\
 & & \text{or} \\
 & = & s \text{ for some } s \in \mathrm{Stream} \text{ depending on } x
\end{array}
$$

Induction

- Induction is essentially used to prove uniqueness of iteration and primitive recursion.

Theorem

Let $(\mathbb{N}, 0, \mathrm{S})$ *be an* $\mathbb{N}$*-algebra. The following is equivalent*

1. *The principle of unique iteration.*
2. *The principle of unique primitive recursion.*
3. *The principle of iteration* $+$ *induction.*
4. *The principle of primitive recursion* $+$ *induction.*

Coinduction

- Uniqueness in coiteration is equivalent to the principle:
  Bisimulation implies equality
- Bisimulation on Stream is the largest relation $\sim$ on Stream s.t.

$$s \sim s' \to \mathrm{head}(s) = \mathrm{head}(s') \wedge \mathrm{tail}(s) \sim \mathrm{tail}(s')$$

- Largest can be expressed as $\sim$ being an indexed coinductively defined set.
- Primitive corecursion over $\sim$ means:
  We can prove

$$\forall s, s'.X(s, s') \to s \sim s'$$

  by showing

$$\begin{array}{lcl} X(s, s') & \to & \mathrm{head}(s) = \mathrm{head}(s') \\ X(s, s') & \to & X(\mathrm{tail}(s), \mathrm{tail}(s')) \vee \mathrm{tail}(s) \sim \mathrm{tail}(s') \end{array}$$

Schema of Coinduction

- Combining
  - bisimulation implies equality
  - bisimulation can be shown corecursively

  we obtain the following principle of **coinduction**:
- We can prove
$$\forall s, s'.X(s, s') \to s = s'$$
  by showing
$$\begin{array}{rcl} \forall s, s'.X(s, s') & \to & \mathrm{head}(s) = \mathrm{head}(s') \\ \forall s, s'.X(s, s') & \to & \mathrm{tail}(s) = \mathrm{tail}(s') \end{array}$$
  where $\mathrm{tail}(s) = \mathrm{tail}(s')$ can be derived
  - directly or
  - from a proof of
$$X(\mathrm{tail}(s), \mathrm{tail}(s'))$$
    invoking the **co-induction-hypothesis** (which can be only used directly)
$$X(\mathrm{tail}(s), \mathrm{tail}(s')) \to \mathrm{tail}(s) = \mathrm{tail}(s')$$

Example

- Define by **primitive corecursion**

$$\begin{array}{lcl} s \in \mathrm{Stream} & & \\ \mathrm{head}(s) & = & 0 \\ \mathrm{tail}(s) & = & s \end{array} \qquad \begin{array}{lcl} s' : \mathbb{N} \to \mathrm{Stream} & & \\ \mathrm{head}(s'(n)) & = & 0 \\ \mathrm{tail}(s'(n)) & = & s'(n+1) \end{array}$$

$$\begin{array}{lcl} \mathrm{cons} : \mathbb{N} \to \mathrm{Stream} \to \mathrm{Stream} & & \\ \mathrm{head}(\mathrm{cons}(n,s)) & = & n \\ \mathrm{tail}(\mathrm{cons}(n,s)) & = & s \end{array}$$

- We show $\forall n \in \mathbb{N}.s = s'(n)$ by **coinduction**:
  Assume $n \in \mathbb{N}$. $\mathrm{head}(s) = \mathrm{head}(s'(n))$ and $\mathrm{tail}(s) = s = s'(n+1) = \mathrm{tail}(s'(n))$, where $s = s'(n+1)$ follows by the **co-IH**.
- We show $\mathrm{cons}(0,s) = s$ by coinduction:
  $\mathrm{head}(\mathrm{cons}(0,s)) = 0 = \mathrm{head}(s)$ and $\mathrm{tail}(\mathrm{cons}(0,s)) = s = \mathrm{tail}(s)$, where we did not use the co-IH.

Equivalence

Theorem

Let $(\mathrm{Stream}, \mathrm{head}, \mathrm{tail})$ *be a* Stream*-coalgebra. The following is equivalent*

1. *The principle of unique coiteration.*
2. *The principle of unique primitive corecursion.*
3. *The principle of coiteration + coinduction*
4. *The principle of primitive corecursion + coinduction*

Duality

[1]

Inductive Definition	Coinductive Definition
Determined by Introduction	Determined by Observation/Elimination
Iteration	Coiteration
Pattern matching	Copattern matching
Primitive Recursion	Primitive Corecursion
Induction	Coinduction
Induction-Hypothesis	Coinduction-Hypothesis

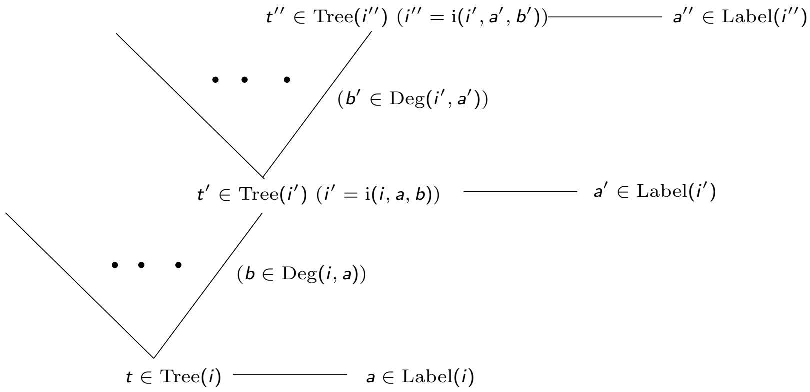
[1] This table is essentially due to Peter Hancock.

(Co)Iteration – (Co)Recursion – (Co)Induction

Schemata for Corecursive Definitions and Coinductive Proofs

$\mathbb{N}^\infty$, CoEven, CoOdd

Generalisation: Petersson-Synek Trees (or Fixed Points of Containers)



Petersson-Synek Trees (PST)

- Strictly positive inductive definitions can be reduced to the PSTs
- Inductive PSTs are the data types

$$
\begin{array}{l}
\mathrm{data}\ \mathrm{Tree} : \mathrm{I} \to \mathrm{Set}\ \mathrm{where} \\
\quad \mathrm{C} : (((i \in \mathrm{I}) \times (a \in \mathrm{Label}(i)) \\
\qquad \times ((b \in \mathrm{Deg}(i,a)) \to \mathrm{Tree}(\mathrm{j}(i,a,b)) \\
\qquad \to \mathrm{Tree}(i)
\end{array}
$$

- Coinductive PSTs are defined follows:

$$
\begin{array}{lll}
\multicolumn{3}{l}{\mathrm{coalg}\ \mathrm{Tree}^{\infty} : \mathrm{I} \to \mathrm{Set}\ \mathrm{where}} \\
\quad \mathrm{label} & : & ((i \in \mathrm{I}) \times \mathrm{Tree}^{\infty}(i)) \to \mathrm{Label}(i) \\
\quad \mathrm{subtree} & : & ((i \in \mathrm{I}) \times (t \in \mathrm{Tree}^{\infty}(i)) \\
 & & \ \times (b \in \mathrm{Deg}(i, \mathrm{label}(i,t)))) \\
 & & \to \mathrm{Tree}^{\infty}(\mathrm{j}(i, \mathrm{label}(i,t), b))
\end{array}
$$

Equivalence of unique (Co)induction, (Co)recursion, (Co)induction

- The notions of (co)iteration, primitive (co)recursion, (co)induction can be generalised in a straightforward way to PSTs and Co-PSTs.
- One can show the equivalence of
  - unique iteration, unique primitive recursion, iteration + induction, primitive recursion + induction
  - unique coiteration, unique primitive corecursion, coiteration + coinduction, primitive corecursion + coinduction

Schema for Primitive Corecursion

- Consider

$$\begin{array}{l} \mathrm{coalg}\ \mathrm{Tree}^{\infty} : \mathrm{I} \to \mathrm{Set}\ \mathrm{where} \\ \quad \mathrm{label} \quad : \quad ((i \in \mathrm{I}) \times \mathrm{Tree}^{\infty}(i)) \to \mathrm{Label}(i) \\ \quad \mathrm{subtree}: \quad ((i \in \mathrm{I}) \times (t \in \mathrm{Tree}^{\infty}(i)) \times (b \in \mathrm{Deg}(i, \mathrm{label}(i,t)))) \\ \qquad\qquad\qquad \to \mathrm{Tree}^{\infty}(\mathrm{j}(i, \mathrm{label}(i,t), b)) \end{array}$$

- We can define a function

$$\begin{array}{lcll} f : ((i \in \mathrm{I}) \times X(i)) \to \mathrm{Tree}^{\infty}(i) & & & \\ \mathrm{label}(i, f(i,x)) & = & a'(i,x) & \in \mathrm{Label}(i) \\ \mathrm{subtree}(i, f(i,x), b) & = & t'(i,x,b) & \in \mathrm{Tree}^{\infty}(i') \text{ with } i' := \mathrm{j}(i, a', b) \end{array}$$

  where $a'(i,x) \in \mathrm{Label}(i)$
  and $t'(i,x,b)$ can be defined
  - as an element of $\mathrm{Tree}^{\infty}(i')$ defined before
  - or corecursively defined as $\mathrm{subtree}(i, f(i,x), b) = f(i', x')$ for some $x' \in X(i')$.
    Here $f(i',x')$ will be called the **corecursion hypothesis**.

Schema for Coinduction

- Assume

$$\begin{array}{lcl} J & \in & \mathrm{Set} \\ \widehat{i} & : & J \to \mathrm{I} \\ x_0, x_1 & : & (j \in J) \to \mathrm{Tree}^\infty(\widehat{i}(j)) \end{array}$$

We can show $\forall j \in J.x_0(j) = x_0(j')$ coinductively by showing

- $\mathrm{label}(\widehat{i}(j), x_0(j))$ and $\mathrm{label}(\widehat{i}(j), x_1(j))$ are equal
- and for all b that $\mathrm{subtree}(\widehat{i}(j), x_0(j), b)$ and $\mathrm{subtree}(\widehat{i}(j), x_0(j), b)$ are equal, where we can use either the fact that
  - this was shown before,
  - or we can use the **coinduction-hypothesis**, which means using the fact $\mathrm{subtree}(\widehat{i}(j), x_0(j), b) = x_0(j')$ and $\mathrm{subtree}(\widehat{i}(j), x_1(j), b) = x_1(j')$ for some $j' \in J$.

(Co)Iteration – (Co)Recursion – (Co)Induction

Schemata for Corecursive Definitions and Coinductive Proofs

$\mathbb{N}^\infty$, CoEven, CoOdd

Coinduction over Coinductively Defined Predicates

- When carrying out proofs over coinductively defined sets, one often proves a predicate which is defined coinductively indexed over the coinductively defined sets.
- So we have indexed coinductively defined sets, which can be introduced by corecursion.
- A proof by corecursion can be considered as a proof by coinduction.
- We consider the example of the co-natural numbers.

$\mathbb{N}^\infty$

$$
\begin{array}{l}
\text{coalg } \mathbb{N}^\infty \in \text{Set where} \\
\quad \text{shape} : \mathbb{N}^\infty \to (0 + \mathrm{S}(\mathbb{N}^\infty))
\end{array}
$$

- $\mathbb{N}^\infty$ can be reduced to non-indexed PSTs:

$$
\begin{array}{l}
\text{coalg } \mathbb{N}^\infty \in \text{Set where} \\
\quad \begin{array}{lcl}
\text{label} & : & \mathbb{N}^\infty \to \{0, \mathrm{S}\} \\
\text{subtree} & : & ((n \in \mathbb{N}^\infty) \times \mathrm{Deg}(\text{label}(n))) \to \mathbb{N}^\infty
\end{array} \\
\text{where } \begin{array}{lcl}
\mathrm{Deg}(0) & = & \emptyset \\
\mathrm{Deg}(\mathrm{S}) & = & \{*\}
\end{array}
\end{array}
$$

- Define $+$ by primitive corecursion

$$
\begin{array}{l}
\_ + \_ : (\mathbb{N}^\infty \times \mathbb{N}^\infty) \to \mathbb{N}^\infty \\
\text{shape}(n + m) = \text{case shape}(m) \text{ of} \\
\qquad \qquad \{ \begin{array}{lcl}
0 & \longrightarrow & \text{shape}(n) \\
\mathrm{S}(m') & \longrightarrow & \mathrm{S}(n + m') \ \}
\end{array}
\end{array}
$$

CoEven, CoOdd

- We define simultaneously coinductively

$$\mathrm{CoEven} : \mathbb{N}^\infty \to \mathrm{Set}$$
$$\mathrm{CoEven}(n) \to \mathrm{CoEvenCond}(\mathrm{shape}(n))$$

$$\mathrm{CoOdd} : \mathbb{N}^\infty \to \mathrm{Set}$$
$$\mathrm{CoOdd}(n) \to \mathrm{CoOddCond}(\mathrm{shape}(n))$$

where

$$\mathrm{CoEvenCond}(0) \text{ is true}$$
$$\mathrm{CoEvenCond}(\mathrm{S}(m)) = \mathrm{CoOdd}(m)$$

$$\mathrm{CoOddCond}(0) \text{ doesn't hold}$$
$$\mathrm{CoOddCond}(\mathrm{S}(m)) = \mathrm{CoEven}(m)$$

CoEven, CoOdd as PSTs

- Define CoEven, CoOdd as one PST indexed over
  $\mathrm{I} := \{\mathrm{CoEven}, \mathrm{CoOdd}\} \times \mathbb{N}^\infty \times \mathbb{N}^\infty$

$$
\begin{array}{l}
\mathrm{coalg\ CoEvenOdd} : \mathrm{I} \to \mathrm{Set\ where} \\
\quad \begin{array}{lcl}
\mathrm{label} & : & ((i \in \mathrm{I}) \times \mathrm{CoEvenOdd}(i)) \to \mathrm{Label}(i) \\
\mathrm{subtree} & : & ((i \in \mathrm{I}) \times (p \in \mathrm{CoEvenOdd}(i)) \times \mathrm{Deg}(i, \mathrm{label}(i, p))) \\
 & & \to \mathrm{CoEvenOdd}(\mathrm{j}(i))
\end{array}
\end{array}
$$

where

$$
\begin{array}{lcl}
\mathrm{Label}(c, n, m) & = & \begin{cases} \emptyset & \text{if } \mathrm{shape}(m) = 0 \text{ and } c = \mathrm{CoOdd} \\ \{*\} & \text{otherwise} \end{cases} \\
\mathrm{Deg}(c, n, m) & = & \begin{cases} \emptyset & \text{if } \mathrm{shape}(m) = 0 \text{ and } c = \mathrm{CoEven} \\ \{*\} & \text{otherwise} \end{cases} \\
\mathrm{j}(\mathrm{CoEven}, n, m) & = & (\mathrm{CoOdd}, n, \mathrm{pred}(m)) \\
\mathrm{j}(\mathrm{CoOdd}, n, m) & = & (\mathrm{CoEven}, n, \mathrm{pred}(m))
\end{array}
$$

Closure of CoEven under $+$

- We show simultaneously

$$\forall n, m \in \mathbb{N}^\infty.\mathrm{CoEven}(n) \to \mathrm{CoEven}(m) \to \mathrm{CoEven}(n+m)$$
$$\forall n, m \in \mathbb{N}^\infty.\mathrm{CoEven}(n) \to \mathrm{CoOdd}(m) \to \mathrm{CoOdd}(n+m)$$

  by coinduction on CoEven, CoOdd

  - Assume n, m, $\mathrm{CoEven}(n)$, $\mathrm{CoEven}(m)$.
    For showing $\mathrm{CoEven}(n+m)$ we have to show $\mathrm{CoEvenCond}(\mathrm{shape}(n+m))$.
    - If $\mathrm{shape}(m) = \mathrm{zero}$ then $\mathrm{shape}(n+m) = \mathrm{shape}(n)$ and by $\mathrm{CoEven}(n)$ we have $\mathrm{CoEvenCond}(\mathrm{shape}(n))$.
    - If $\mathrm{shape}(m) = \mathrm{S}(m')$ then $\mathrm{shape}(n+m) = \mathrm{S}(n+m')$, $\mathrm{CoEvenCond}(\mathrm{shape}(n+m)) = \mathrm{CoOdd}(n+m')$ which follows by the **coIH** and $\mathrm{CoOdd}(m')$.
  - The proof of the second condition follows similarly

Conclusion

- Coiteration, primitive corecursion, coinduction are the duals of iteration, primitive recursion, induction.
- In iteration/recursion/induction, the instances of the co-IH used are restricted, but the result can be used in arbitrary functions and formulas.
- In coiteration/corecursion/coinduction, the instances of the co-IH are unrestricted, but the result can be only used directly.
- General case of indexed coinductively defined sets can be reduced to co-PSTs.
- Schemata for primitive corecursion and coinduction.
- Schemata can be applied to indexed coinductively defined sets and relations.
- Relations on coinductively defined sets seem to be often coinductively defined indexed relations and can be shown by indexed corecursion.

Happy Birthday