Algebras and Coalgebras in dependent type theory

Anton Setzer

Swansea University

12 April 2012

From the infamous "TPL Book"

Alternatively, an algebra may be displayed expansively in the following way:

algebra	A
carriers	$\ldots, A_s, \ldots$
constant	
	$\vdots$
	$a :\to A_s$
	$\vdots$
operations	
	$\vdots$
	$f : A_{s(1)} \times \cdots \times A_{s(n)} \to A_s$
	$\vdots$

Notations

- We use functional notation $f\ a_1 \cdots a_n$ instead of $f(a_1, \ldots, a_n)$.
- $A_1 + \cdots + A_n$ is the disjoint union of A_i.

  If there is a natural index i for A_i (usually an operation) let

  $$\mathrm{in}_i : A_i \to A_1 + \cdots + A_n$$

  be the injection.
- We write $a : A$ for a is of type A.
- Set denotes the type of sets,

  $S \to \mathrm{Set}$ is the type of S-indexed sets.

Dependent Function/Sum Type

- We define the **dependent function type**

$$(a : A) \to B[a]$$

  of functions mapping $a : A$ to an element of $B[a]$.
- We define the **dependent sum type**

$$(a : A) \times B[a]$$

  consisting of $\langle a, b \rangle$ s.t. $a : A$, $b : B[a]$.

From Algebras to Indexed Inductive Definitions

Indexed Inductive-Recursive Definitions

Induction-Induction

Coalgebras

Single Sorted Case (Omit S)

- Let for $a :\to A$, $X : \mathrm{Set}$,

$$F_a(X) := \{*\}$$

  So essentially

$$a : F_a(A) \to A$$

- Let for $f : A^n \to A$, $X : \mathrm{Set}$

$$F_f(X) := X^n$$

  So

$$f : F_f(A) \to A$$

- Let

$$F(X) := F_{f_1}(X) + \cdots + F_{f_l}(X)$$

  So

$$\vec{f} := [f_1, \ldots, f_n] : F(A) \to A$$

- $(A, \vec{f})$ is an F-**algebra**.

Multi Sorted Case, Restricted Version

- Let $S := \{s_1, \ldots, s_n\}$ be the set of all sorts.
- Consider A as of type $A : S \to \mathrm{Set}$.
- Assume now $X : S \to \mathrm{Set}$
  - If $a :\to A_s$, let
    $$F_a(X) := \{*\}$$
  - If $f : A_{s(1)} \times \cdots \times A_{s(n)} \to A_s$, let
    $$F_a(X) := X_{s(1)} \times \cdots \times X_{s(n)}$$
  - Let for $s : S$
    $$F^s(X) := F_{f_{i_1}}(X) + \cdots + F_{f_{i_l}}(X)$$
    where $f_{i_1}, \ldots, f_{i_l}$ are the functions with target type A_s.
  - Define
    $$\vec{f} : (s : S) \to F^s(A) \to A_s \quad ,$$
    $$\vec{f}\ s\ (\mathrm{in}_a\ *) = a \quad ,$$
    $$\vec{f}\ s\ (\mathrm{in}_f\ \langle x_1, \ldots, x_n \rangle) = f(x_1, \ldots, x_n)$$

Multi Sorted Case, Generalised Version

- Let S, A, F_a, F_f as before
- Let

$$F(X) := F_{f_1}(X) + \cdots + F_{f_n}(X)$$

  where $f_1, \ldots, f_l$ are all the constants and operations of A.
- Define

$$\mathrm{index} : F(X) \to S$$

  where if $f : A_{s_1} \times \cdots \times A_{s_n} \to A_s$, then

$$\mathrm{index}\ (\mathrm{in}_{f_i}\ \langle x_1, \ldots, x_n \rangle) = s$$

- Define

$$\begin{aligned} \vec{f} &: (a : F(A)) \to A_{\mathrm{index}\ s}\ , \\ \vec{f}\ (\mathrm{in}_a\ *) &= a \\ \vec{f}\ s\ (\mathrm{in}_f\ \langle x_1, \ldots, x_n \rangle) &= f\ \langle x_1, \ldots, x_n \rangle \end{aligned}$$

- Then $(A, \vec{f})$ is a **generalised F-algebra** because of the type of $\vec{f}$.

Generalisation

- Up to now $F(X)$ is the disjoint union of products of X_i.
- We can throw in as basic sets as well some $B : \mathrm{Set}$ defined before. These will be called "**non-inductive arguments**".
  - Used when forming algebras referring to other algebras.
- We can refer to many arguments of X_s simultaneously. So we have arguments of type

$$(b : B) \to X_{s(b)}$$

where $B : \mathrm{Set}$, $s : B \to S$.
These arguments are called "**inductive arguments**".

Generalisation

- We can allow the type of later arguments depend on previous non-inductive arguments.
- We replace
$$F_{f_1}(X) + \cdots + F_{f_n}(X)$$
by
$$(f : \{f_1, \ldots, f_n\}) \times F_f(X)$$
so we need no disjoint union.
- We define polynomial functors for the general case, the restricted version is a special case of this.

Polynomial Functors

- The **set of polynomial functors** $F : (S \to \mathrm{Set}) \to \mathrm{Set}$ together with $\mathrm{index} : F\ X) \to S$ is given by
  - Base Case:
    The following is polynomial ($s : S$):

    $$F\ X = \{*\}$$
    $$\mathrm{index}\ * = s$$

  - Non-inductive Argument:
    Assume $B : \mathrm{Set}$ and that for $b : B$ we have (F_b, index_b) is polynomial.
    The following is polynomial:

    $$F\ X = (b : B) \times F_b\ X$$
    $$\mathrm{index}\ \langle b, x\rangle = \mathrm{index}_b\ x$$

Polynomial Functors

- Inductive Argument:
  Assume $B : \mathrm{Set}$, $s : B \to S$, (F', index') is polynomial.
  The following is polynomial:

$$F\ X = ((b : B) \to X_{s(b)}) \times F'\ X$$
$$\mathrm{index}\ \langle x, y \rangle = \mathrm{index}'\ y$$

Restricted/Generalised Indexed Inductive Definitions

- Restricted indexed inductive definitions are initial algebras for polynomial functors

$$F_s : (S \to \mathrm{Set}) \to \mathrm{Set} \quad ,$$
$$\mathrm{index}_s\ x = s$$

and the introduction rule has the form

$$\mathrm{intro} : (s : S) \to F_s\ A \to A_s$$

- Generalised indexed inductive definitions are initial algebras for a polynomial functor

$$F : (S \to \mathrm{Set}) \to \mathrm{Set} \quad ,$$
$$\mathrm{index} : F\ X \to S$$

and the introduction rule has the form

$$\mathrm{intro} : (x : F\ A) \to A_{\mathrm{index}\ x}$$

Asymmetry of Indexed Inductive Definitions

- Asymmetry of arguments:
  - Only dependency on non-inductive arguments not on inductive arguments.
  - Direct dependency not possible, since we don't know what X is.
- Solution: instead of defining just $X : S \to \mathrm{Set}$ define
  - $X : S \to \mathrm{Set}$ inductively together with
  - $T : (s : S) \to X_s \to D[s]$ recursively for some type $D[s]$.
  - Later arguments can depend on T applied to inductive arguments.

Generalised Indexed Inductive Definitions

- Let $\mathrm{Fam}_S(D) := (X : S \to \mathrm{Set}) \times ((s : S) \to X\ s \to D[s])$.
- So define

$$
\begin{aligned}
F &: \mathrm{Fam}_S(D) \to \mathrm{Set} \\
\mathrm{index} &: F\ X \to S \\
\mathrm{toD} &: (x : F\ X) \to D[\mathrm{index}\ x]
\end{aligned}
$$

- The formation and introduction rules are now

$$
\begin{aligned}
A &: S \to \mathrm{Set} \\
T &: (s : S) \to A\ s \to D[s]
\end{aligned}
$$

$$
\begin{aligned}
&\mathrm{intro} : (a : F\ \langle A, T \rangle) \to A_{\mathrm{index}\ a} \\
&T\ (\mathrm{index}\ a)\ (\mathrm{intro}\ a) = \mathrm{toD}\ a
\end{aligned}
$$

Polynomial Functors

- The **set of polynomial functors** with related functions

$$
\begin{aligned}
&F : \mathrm{Fam}_S(D) \to \mathrm{Set} \\
&\mathrm{index} : F\ X \to S \\
&\mathrm{toD} : (x : F\ X) \to D[\mathrm{index}\ x]
\end{aligned}
$$

  is defined as follows:
  - Base Case:
    Let $s : S$, $d : D[s]$.
    The following is polynomial:

$$
\begin{aligned}
&F\ X = \{*\} \\
&\mathrm{index} \quad * \quad = \quad s \\
&\mathrm{toD} \quad * \quad = \quad d
\end{aligned}
$$

Polynomial Functors

- Non-inductive Argument:
  Assume $B : \mathrm{Set}$ and for $b : B$ we have (F_b, index_b) is polynomial.
  The following is polynomial:

$$
\begin{array}{llll}
F\ X = (b : B) \times F_b\ X & , & & \\
\mathrm{index} \quad \langle b, x \rangle & = & \mathrm{index}_b & x \ , \\
\mathrm{toD} \quad \langle b, x \rangle & = & \mathrm{toD}_b & x \ .
\end{array}
$$

Polynomial Functors

- Inductive Argument:
  Assume $B : \mathrm{Set}$, $s : B \to S$.
  Assume for $t : (b : B) \to D[s\ b]$ we have

$$(F_t, \mathrm{index}_t, \mathrm{toD}_t)$$

  are polynomial.
  The following is polynomial

$$\begin{array}{lll} F\ \langle X, T\rangle = (f : (b : B) \to X_{s\ b}) \times F_{t \circ f}\ \langle X, t\rangle & & , \\ \mathrm{index} \quad \langle f, X\rangle & = \quad \mathrm{index}_{t \circ f} \quad x & , \\ \mathrm{toD} \quad \langle f, X\rangle & = \quad \mathrm{toD}_{t \circ f} \quad x & . \end{array}$$

From Algebras to Indexed Inductive Definitions

Indexed Inductive-Recursive Definitions

Induction-Induction

Coalgebras

Induction-Induction

- PhD Project of Fredrik Forsberg.
- In single sorted Induction Recursion we defined
  - $A : \mathrm{Set}$ inductively, while defining
  - $T : A \to D$ recursively.
- In Induction Induction we define
  - $A : \mathrm{Set}$ inductively, while defining
  - $B : A \to \mathrm{Set}$ inductively.

Example Surreal Numbers

- We define the surreal numbers

$$\text{Surreal} : \mathrm{Set}$$

together with relations

$$\begin{aligned} x \leq y &\quad : \quad \mathrm{Set} \\ x \not\leq y &\quad : \quad \mathrm{Set} \end{aligned}$$

for x, y : Surreal inductive-inductively.
(Size problems required modifications, see paper).

Example Surreal Numbers

- Assume

$$X_{\mathrm{L}}, X_{\mathrm{R}} : \mathcal{P}(\text{Surreal})$$
$$\forall x \in X_{\mathrm{L}}.\forall y \in X_{\mathrm{R}}.x \not\leq y$$

  Then

$$(X_{\mathrm{L}}, X_{\mathrm{R}}) : \text{Surreal}$$

- Assume

$$X = (X_{\mathrm{L}}, X_{\mathrm{R}}) : \text{Surreal}$$
$$Y = (Y_{\mathrm{L}}, Y_{\mathrm{R}}) : \text{Surreal}$$

  Assume
  - $\forall x \in X_{\mathrm{L}}.Y \not\leq x$.
  - $\forall y \in Y_{\mathrm{R}}.y \not\leq X$.

  Then $X \leq Y$.

Example Surreal Numbers

- Assume

$$X = (X_{\mathrm{L}}, X_{\mathrm{R}}) : \mathsf{Surreal}$$
$$Y = (Y_{\mathrm{L}}, Y_{\mathrm{R}}) : \mathsf{Surreal}$$

  Assume
  - $\exists x \in X_{\mathrm{L}} . Y \leq x$ or
  - $\exists y \in Y_{\mathrm{R}} . y \leq X$.

  Then $X \not\leq Y$.

From Algebras to Indexed Inductive Definitions

Indexed Inductive-Recursive Definitions

Induction-Induction

Coalgebras

Coalgebras

- Restriction to the simplest non-indexed case.
- Algebras are functions

$$f : F\ A \to A$$

  Simplest example Lists:

$$[\mathrm{nil}, \mathrm{cons}] : (\{*\} + A \times \mathrm{List}\ A) \to \mathrm{List}\ A$$

- **Coalgebras** are functions

$$f : A \to F\ A$$

- Colists are sets $\mathrm{coList}\ A : \mathrm{Set}$ together with

$$\mathrm{case} : \mathrm{coList}\ A \to (\{*\} + A \times \mathrm{List}\ A)$$

Misconception

- Often people think colists consist of

$$\mathrm{cons}\ a_1\ (\mathrm{cons}\ a_2\ \cdots (\mathrm{cons}\ a_n\ \mathrm{nil}) \cdots)$$

or infinite streams

$$\mathrm{cons}\ a_1\ (\mathrm{cons}\ a_2\ \cdots)$$

- In our setting colists are not infinite, but can be unfolded potentially infinitely many.
- Example: the increasing colist is given by

$$\begin{aligned}&\mathrm{inc} : \mathbb{N} \to \mathrm{coList}\\ &\mathrm{case}\ (\mathrm{inc}\ n) = \mathrm{inr}\ \langle n, \mathrm{inc}\ (n+1)\rangle\end{aligned}$$

Theory of Coalgebras

- Can be developed for indexed coalgebras with dependencies.
- Extensions to induction-recursion don't make sense yet.
- In type theory
  - Algebras are determined by their introduction rules, the elimination rules are "derived".
  - Coalgebras are determined by their elimination rules, the introduction rules are "derived".

Conclusion

- From algebra as in computer science to an abstract notion of algebras.
- Non-indexed, indexed inductive definitions.
- Induction recursion more symmetric.
- Induction induction seems to occur often in mathematics.
- Coalgebras as sets defined by their elimination rules.