A Coalgebraic Treatment of Context-free Languages

Joost Winter

April 7, 2011

What is coalgebra?

Given a functor F, an F-coalgebra consists of a tuple (X, f):

- X is a set, the *carrier set*.
- f is a function from X to FX.

Diagrammatically:

$$\begin{array}{c} X \\ \downarrow{\scriptstyle f} \\ FX \end{array}$$

Coalgebras representing languages (1)

In the remainder of this talk, we will be concerned with coalgebras over the functor $\mathcal{D} := 2 \times (-)^A$. But what does this mean?

- 2 is the set $\{0, 1\}$.
- $\times$ is the cartesian product.
- A is a finite set called the *alphabet*.
- $(-)$ is a placeholder for the carrier set.
- X^A denotes the function space from A to X.

Coalgebras representing languages (2)

So: a coalgebra (X, f) over the functor $\mathcal{D}$ consists of a set X and a function f that maps every $x \in X$ to an element $f(x) \in 2 \times X^A$.

In this talk, we will use the following notation:

- Given $x \in X$, $o(x)$ (called the *output value of* x) will be the first component of $f(x)$.
- Given $x \in X$, x_a (called the *a-derivative of* x) will be the second component of $f(x)$, applied to a.

So: for every x, $o(x)$ is either 0 or 1, and for every x and a, x_a is an element of X again.

Coalgebras representing languages (3)

We can extend the notion of derivatives from alphabet symbols (i.e. elements of A) to words (i.e. elements of A^*) inductively:

- $x_\lambda = x$
- $x_{a \cdot w} = (x_a)_w$.

Homomorphisms and bisimulations

Given two $\mathcal{D}$-coalgebras (X, f) and (Y, g), a function $h : X \to Y$ is a *homomorphism* if the following hold:

1. For every $x \in X$, $o(x) = o(h(x))$.
2. For every $x \in X$ and $a \in A$, $h(x_a) = (h(x))_a$.

Given two $\mathcal{D}$-coalgebras (X, f) and (Y, g), a relation $R \subseteq X \times Y$ is a *bisimulation* if the following hold:

1. If $(x, y) \in R$, then $o(x) = o(y)$.
2. If $(x, y) \in R$, then for all $a \in A$, $(x_a, y_a) \in R$.

The final coalgebra of all languages

Consider the $\mathcal{D}$-coalgebra $(\mathcal{L}, l)$ defined as follows:

- $\mathcal{L}$ is the set of all languages over the alphabet A, i.e. $\mathcal{P}(A^*)$.
- For any $L \in \mathcal{L}$:
  - $o(L) = 1$ iff the empty word is in L.
  - $L_a = \{w \in A^* \mid a \cdot w \in L\}$.

This is a *final coalgebra*: for every $\mathcal{D}$-coalgebra (X, f), there is a *unique* homomorphism h from (X, f) to $(\mathcal{L}, l)$.
Given a $\mathcal{D}$-coalgebra (X, f), and an element $x \in X$, we let $[\![x]\!]$ denote the value of x under this unique homomorphism.

The coalgebra of regular expressions

The set $\mathcal{E}$ of *regular expressions* over an alphabet A can be defined as follows:

$$t ::= a \in A \mid 0 \mid 1 \mid t + t \mid t \cdot t \mid t^*$$

We can assign a $\mathcal{D}$-coalgebra structure to this set of regular expressions by specifying the output values and derivatives for each expression, giving us a $\mathcal{D}$-coalgebra $(\mathcal{E}, e)$:

t	$o(t)$	t_a
0	0	0
1	1	0
b	0	if $b = a$ then 1 else 0
$u + v$	$o(u) \vee o(v)$	$u_a + v_a$
$u \cdot v$	$o(u) \wedge o(v)$	$u_a \cdot v + o(u) \cdot v_a$
u^*	1	$u_a \cdot u^*$

Kleene's theorem, coalgebraically

- For any language $L \in \mathcal{L}$, we define the subcoalgebra generated by $\mathcal{L}$ as
$$\langle L \rangle := \{L_w \mid w \in A^*\}$$
It is easy to see that this indeed generates a subcoalgebra: given any $K \in \langle L \rangle$, it is easy to see that for every $a \in A$, also $K_a \in \langle L \rangle$. In other words, $\langle L \rangle$ is closed under taking derivatives to alphabet symbols.
- Kleene's theorem, coalgebraically (Rutten, 1998): *For any $L \in \mathcal{L}$, $\langle L \rangle$ is finite iff there is a regular expression t such that $L = [\![t]\!]$.*

Introduction: context-free grammars and languages

- The 'next step up' from regular expressions and languages, and finite automata, in the Chomsky hierarchy, are the context-free languages and grammars, and pushdown automata.
- We will present a format of coinductively defined systems of equations: it turns out that these systems of equations chracterize precisely the context-free languages.

Systems of equations (1)

We will use terms t specified as follows:

$$t ::= a \in A \,|\, x \in X \,|\, 0 \,|\, 1 \,|\, t + t \,|\, t \cdot t$$

where X is a finite set of variables, and A, as before, is a finite alphabet. Given X, we let TX denote the set of terms over X. A well-formed system of equations, for a set of variables X, consists of:

1. For every $x \in X$, exactly one equation of the form $o(x) = v$, where $v \in \{0, 1\}$.
2. For every $x \in X$ and $a \in A$, exactly one equation of the form $x_a = t$, where $t \in TX$.

Systems of equations (2)

Alternatively, we can consider a well-formed system of equations as a mapping

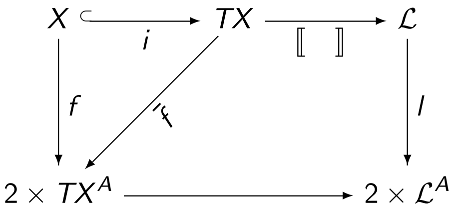
$$f : X \to 2 \times TX^A$$

We can extend such a mapping f to the $\mathcal{D}$-coalgebra $(TX, \bar{f})$ *generated by* (X, f) as follows:

t	$o(t)$	t_a
x	$o(x)$	x_a (as specified by f)
0	0	0
1	1	0
b	0	if $b = a$ then 1 else 0
$u + v$	$o(u) \vee o(v)$	$u_a + v_a$
$u \cdot v$	$o(u) \wedge o(v)$	$u_a \cdot v + o(u) \cdot v_a$

Systems of equations (3)

- This construction can be summarized diagrammatically:



- Proposition: *A language $\mathcal{L}$ is context-free iff there is a well-formed system of equations (X, f) and an $x \in X$, such that $[\![x]\!] = \mathcal{L}$ w.r.t. the coalgebra $(TX, \bar{f})$ generated by it.*

From CFGs to systems of equations (1)

- We say a context-free grammar is in weak Greibach normal form, if every production rule has a right hand side either equal to the empty word λ, or of the form $a \cdot t$.
- As the name implies, this is a weakening of the more familiar Greibach normal form. As a direct result, every CFG can be represented in weak Greibach normal form.

From CFGs to systems of equations (2)

We transform a CFG G in weak Greibach normal form into a system of equations as follows:

- We let the set X of variables be equal to the set of nonterminals in the grammar.
- Given a $x \in X$, we set $o(x) = 1$ iff the grammar contains a production rule $x \longrightarrow \lambda$.
- Given a $x \in X$ and an $a \in A$, we set

$$x_a = \sum \{ w \mid x \longrightarrow a \cdot w \}$$

Given an initial symbol $x_0 \in X$, we now have $(x_0)_w = 1$ (and, hence, $w \in [\![x_0]\!]$) iff w is in the language generated by G.

From systems of equations to CFGs

Conversely, given a system of equations, we can construct a CFG in weak Greibach normal form:

- We first transform the system of equations to a new, equivalent, system, in which all derivatives are in disjunctive normal form, and do not contain any superfluous 0s or 1s.
- Derivatives in this new system are disjunctions of sequences of alphabet symbols and variables.
- We let the grammar include a rule $x \to \lambda$ whenever $o(x) = 1$.
- We let the grammar include a rule $x \to a \cdot w$, whenever w is a sequence of alphabet symbols and variables occurring as a disjunct in x_a.

Wrap-up

- There is a very neat coalgebraic representation of regular expressions, and Kleene's theorem can be expressed succinctly in a coalgebraic fashion.
- We have extended this work towards context-free languages and grammars, and provided a coalgebraic characterization using systems of equations.
- Future work: extend the above work to other functors. This is likely to be a successful enterprise: we already discovered some nice examples of context-free streams.
- More future work: a coalgebraic account of pushdown-automata.

Bibliography

- [Jacobs/Rutten, 1997] Bart Jacobs, Jan Rutten, *A Tutorial on (Co)Algebras and (Co)Induction*
- [Rutten, 1998] Jan Rutten, *Automata and Coinduction (An Exercise in Coalgebra)*
- [Rutten, 2005] Jan Rutten, *A Coinductive Calculus of Streams*
- [Silva, 2010] Alexandra Silva, *Kleene Coalgebra*
- [Winter/Bonsangue/Rutten, 2011] Joost Winter, Marcello Bonsangue, Jan Rutten, *Context-free Languages, Coalgebraically*