A tribute to attributes

Lambert Meertens Jaap van der Woude

February 28, 1991

1 Introduction

A common phenomenon in programming is "flushing of results", the ingredients of which are built up in some structure (e.g. a stack). In many simple problems those ingredients are plain values and recipes to combine them (operators). This structure depends on the structure of the given data and it may be viewed as an instance of an attribute-decorated parse tree.
We shall give another example of this phenomenon where the values may be functions. The example is a stepping stone towards catamorphising attribute grammar problems in general. Two aspects of this example are especially worth mentioning:
- Parameterised algebras and their catamorphisms; we propose a general expression.
- Lifting of operators, or rather algebras. Too bad, but we have to postpone that subject.

This note depends heavily on notation and calculation techniques that can be found for instance in [2] and, in a slightly different setting, in [0]. The reader is advised to consult these references (not merely as references, they are atleast as interesting as the note here).

A remark on the priorities of the occurring operators may be in order. The general rule is that the precedence of the operator is antiproportional to its size. In weakening order we have for instance:

$$,\ .\ ,\ \circ\ ,\ \{\times,\triangle\}\ ,\ \{+,\triangledown\}\ ,\ \rightarrow$$

2 Depth in the leaves

The example we address may be formulated losely as follows: Given a tree, replace the values in the leaves by the depth of the given tree.
For a brief sketch of the setting of the problem:
Let $\dagger$ be the type functor for trees over A. Its unary leftsection $A\dagger$ is given by:

$$A\dagger X = A + X \times X \quad \text{and} \quad A\dagger f = id_A + f \times f$$

We fix $(Atree\ ,\ \tau_A\ \triangledown\ \#\ )$ to be the initial $A\dagger$-algebra, i.e.

$$\tau_A\ \triangledown\ \#\ : A + Atree \times Atree \longrightarrow Atree$$

The two concerns, depth and replacement, are given by the $A\dagger$-catamorphism

(1) $$\delta\ =\ (\!|1^\bullet\ \triangledown\ \ddagger|\!) : Atree \to \mathbb{N} \quad \text{where} \quad \ddagger.(m,n)\ =\ 1 + m \uparrow n$$

and the *parameterised* $A\dagger$-catamorphism $\rho : \mathbb{N} \to (Atree \to \mathbb{N}tree)$

(2) $$\rho.n\ =\ (\!|(\tau_{\mathbb{N}}\ \triangledown\ \#\ ) \circ (n^\bullet + I \times I)|\!)\ =\ (\!|\tau_{\mathbb{N}} \circ n^\bullet\ \triangledown\ \#\ |\!)$$

In both (1) and (2) $v^\bullet$ denotes the constant v function. The function we are to express in terms of catamorphisms is $\Omega.t\ =\ \rho.(\delta.t).t$. One could defend that by (1) and (2) we are done; however, what we want as a solution is the equivalent of a one pass algorithm. In order to reformulate the requested function slightly, we need the usual apply (@) and the argument-swap "$\overline{\ }$", defined by

(3) $$\overline{\phi}.\gamma.z\ =\ \phi.z.\gamma$$

For Ω we calculate

$$\begin{array}{l} \quad \Omega.t \\ = \{\ \text{def. } \Omega\ \} \\ \quad \rho.(\delta.t).t \\ = \{\ (3)\ \} \\ \quad \overline{\rho}.t.(\delta.t) \\ = \{\ @\ \} \\ \quad @.(\overline{\rho}.t\ ,\ \delta.t) \\ = \{\ \Delta\ \} \\ \quad (@ \circ (\overline{\rho}\ \Delta\ \delta)).t \end{array}$$

Hence we are looking for $\Omega\ =\ @ \circ (\overline{\rho}\ \Delta\ \delta)$.

Since @ is considered to be "simple", we may reformulate our aim as: express $\overline{\rho}\ \Delta\ \delta$ as a catamorphism. With the tupling construction ([1]) in mind, knowing that δ is a catamorphism, it is sufficient to express $\overline{\rho}$ as a catamorphism (but we want more: an explicit expression). Is there any chance of $\overline{\rho}$ being expressable as a catamorphism? By (2) and (3), the type is right

$$\overline{\rho} : Atree \to (\mathbb{N} \to \mathbb{N}tree)$$

So if $\overline{\rho}\ =\ (\!|\psi|\!)$, we know that ψ should have the type

$$\psi\ =\ \alpha\ \triangledown\ \beta : A + (\mathbb{N} \to \mathbb{N}tree) \times (\mathbb{N} \to \mathbb{N}tree) \to (\mathbb{N} \to \mathbb{N}tree)$$

where $\alpha : A \to (\mathbb{N} \to \mathbb{N}tree)$ and $\beta : (\mathbb{N} \to \mathbb{N}tree) \times (\mathbb{N} \to \mathbb{N}tree) \to (\mathbb{N} \to \mathbb{N}tree)$.

Indeed so, in the next section we show that (don't mind the notation)

(4) $$\overline{\rho}\ =\ (\!|\tau^\bullet\ \triangledown\ \widehat{\#}\ |\!)$$

Let us calculate $\overline{\rho}\ \Delta\ \delta$ using (4) and the tupling construction in [1]:

$$
\begin{array}{l}
\quad \overline{\rho} \vartriangle \delta \\
= \quad \{ \ (4) \ , \ (1) \ , \ [1] \ \} \\
\quad (\!| (\tau^{\bullet} \triangledown \widehat{\#}) \circ A\dagger \ll \ \vartriangle \ (1^{\bullet} \triangledown \ddagger) \circ A\dagger \gg |\!) \\
= \quad \{ \ \dagger \ , \ \triangledown \leftrightarrow + \ \text{calc} \ \} \\
\quad (\!| (\tau^{\bullet} \triangledown \widehat{\#} \circ (\ll \times \ll)) \vartriangle (1^{\bullet} \triangledown \ddagger \circ (\gg \times \gg)) |\!) \\
= \quad \{ \ \triangledown \ \text{and} \ \vartriangle \ \text{abide} \ \} \\
\quad (\!| (\tau^{\bullet} \vartriangle 1^{\bullet}) \triangledown (\widehat{\#} \circ (\ll \times \ll) \vartriangle \ddagger \circ (\gg \times \gg)) |\!) \\
= \quad \{ \ \times \leftrightarrow \vartriangle \ \text{calc.} \ , \ \infty := (\ll \times \ll) \vartriangle (\gg \times \gg) \ \} \\
\quad (\!| (\tau^{\bullet} \vartriangle 1^{\bullet}) \triangledown (\widehat{\#} \times \ddagger) \circ \infty |\!)
\end{array}
$$

Indeed, this is the form we expected and it is sufficiently neat. The solution for "catamorphising" Ω being:

$$\Omega \ = \ @ \circ (\!| (\tau^{\bullet} \vartriangle 1^{\bullet}) \triangledown (\widehat{\#} \times \ddagger) \circ \infty |\!)$$

Where ∞ denotes the "centre-swap" $(\ll \times \ll) \vartriangle (\gg \times \gg)$.
We still have to substantiate our claim (4). Knowing it is a useful claim, we roll up our sleeves.

3 Calculation of ψ

In the calculation to follow we use the polymorphic evaluation ε (a curried form of the apply function $@$, which in the literature is called evaluation frequently; so, be warned!):

$$\varepsilon : X \to ((X \to Y) \to Y)$$

defined by $(\varepsilon.z).\gamma \ = \ @.(\gamma, z) \ = \ \gamma.z$; or, equivalently, $\varepsilon \ = \ \overline{I}$.
There is a link between the evaluation and the argument-swap, as follows

$$(5) \qquad \overline{\theta}.z \ = \ \varepsilon.z \circ \theta$$

Assume $\psi : A\dagger(\mathbb{N} \to \mathbb{N}tree) \to (\mathbb{N} \to \mathbb{N}tree)$ such that $\overline{\rho} \ = \ (\!| \psi |\!)$ or rather $\rho \ = \ \overline{(\!| \psi |\!)}$.

$$
\begin{array}{ll}
 & \quad \rho \ = \ \overline{(\!| \psi |\!)} \\
 & \equiv \ \{ \ \text{intro} \ n \ , \ \text{heading for fusion} \ \} \\
 & \quad \rho.n \ = \ \overline{(\!| \psi |\!)}.n \\
 & \equiv \ \{ \ (2) \ , \ (5) \ \} \\
 & \quad (\!| \tau \circ n^{\bullet} \triangledown \ \# \ |\!) \ = \ \varepsilon.n \circ (\!| \psi |\!) \\
 & \Leftarrow \ \{ \ \text{fusion} \ \} \\
 & \quad (\tau \circ n^{\bullet} \triangledown \ \# \ ) \circ A\dagger(\varepsilon.n) \ = \ \varepsilon.n \circ \psi \\
 & \equiv \ \{ \ \dagger \ , \ \triangledown \leftrightarrow + \ \text{calc.} \ \} \\
(\P) & \quad \tau \circ n^{\bullet} \triangledown \ \# \ \circ (\varepsilon.n \times \varepsilon.n) \ = \ \varepsilon.n \circ \psi
\end{array}
$$

Let us try to express the operands in the LHS of ¶ as

$$\tau \circ n^{\bullet} \ = \ \varepsilon.n \circ \alpha \quad \text{and} \quad \# \ \circ (\varepsilon.n \times \varepsilon.n) \ = \ \varepsilon.n \circ \beta$$

Indeed, $(\tau \circ n^\bullet).\gamma \;=\; \tau.n \;=\; \tau^\bullet.\gamma.n \;=\; (\varepsilon.n \circ \tau^\bullet).\gamma$
More involved is the other one:

$$
\begin{array}{l}
\quad (\; +\!\!+ \;\circ (\varepsilon.n \times \varepsilon.n)).\gamma \\
= \;\{\; \gamma \in (\mathbb{N} \to \mathbb{N}tree) \times (\mathbb{N} \to \mathbb{N}tree)\,, \text{ say } \;\gamma \;=\; (\gamma_0, \gamma_1) \;\} \\
\quad +\!\!+ .((\varepsilon.n).\gamma_0, (\varepsilon.n).\gamma_1) \\
= \;\{\; \text{def. } \;\varepsilon\;, \;\; +\!\!+ \;\; \text{as infix operator } \;\} \\
\quad \gamma_0.n \;+\!\!+\; \gamma_1.n \\
= \;\{\; \widehat{+\!\!+} \;\; \text{is lifted version of } \;\; +\!\!+ \;\;\} \\
\quad (\gamma_0 \;\widehat{+\!\!+}\; \gamma_1).n \\
= \;\{\text{def. } \;\varepsilon\;, \;\; \widehat{+\!\!+} \;\; \text{as prefix operator } \;\} \\
\quad (\varepsilon.n \circ \widehat{+\!\!+}\,).\gamma
\end{array}
$$

Hence

$$
\begin{array}{l}
\quad (\P) \\
\equiv \;\{\; \text{above } \;\} \\
\quad (\varepsilon.n \circ \tau^\bullet) \;\triangledown\; (\varepsilon.n \circ \widehat{+\!\!+}\,) \;=\; \varepsilon.n \circ \psi \\
\equiv \;\{\; \text{distribution } \;\} \\
\quad \varepsilon.n \circ (\tau^\bullet \;\triangledown\; \widehat{+\!\!+}\,) \;=\; \varepsilon.n \circ \psi \\
\Leftarrow \;\{\; \text{exit } \;n \;\} \\
\quad \tau^\bullet \;\triangledown\; \widehat{+\!\!+} \;\;=\;\; \psi
\end{array}
$$

This completes the proof of our claim (4).
Two intriguing questions are

- Can we give a generic solution for the catamorphisation of parametrised catamorphisms? Given a suitable setting we can, as will be demonstrated in the next section.
- We used $\widehat{+\!\!+}$, the lifted version of $+\!\!+$. Do there exist a setting and a theory for lifting in general? Indeed, they do; but we don't feel it is in a presentable form yet. (Coming next in this theatre?)

4 Parametrised algebras and catamorphisms

A *parametrised* $\dagger$-algebra for some (unary) functor $\dagger$ is a map

$$\phi : Z \to (A\dagger \to A)$$

i.e. for every $z \in Z$, $\phi.z : A\dagger \to A$ is a $\dagger$-algebra. Such a $\phi.z$ induces a $\dagger$-catamorphism $(\!|\phi.z|\!) : L \to A$, where (L, in) denotes a fixed initial $\dagger$-algebra. In other words, ϕ induces a *parametrised* catamorphism

$$(\!|?|\!) \circ \phi : Z \to (L \to A)$$

Using the argument-swap "$\overline{\ \ }$", we see that

$$\overline{(\!|?|\!) \circ \phi} : L \to (Z \to A)$$

has the right type for it to be a catamorphism, say $\overline{(\!|?|\!) \circ \phi} = (\!|\psi|\!)$, where

$$\psi : (Z \to A)\dagger \to (Z \to A)$$

Let us try and calculate ψ as we did in the former sections:

$$\begin{array}{ll} & (\!|?|\!) \circ \phi = \overline{(\!|\psi|\!)} \\ \equiv & \{ \text{ intro } z \text{ , (5) } \} \\ & (\!|\phi.z|\!) = \varepsilon.z \circ (\!|\psi|\!) \\ \Leftarrow & \{ \text{ fusion } \} \\ (\P\P) & \phi.z \circ (\varepsilon.z)\dagger = \varepsilon.z \circ \psi \end{array}$$

Here we are stuck! How do we reformulate the LHS of $(\P\P)$ such that we may "exit" z? In our example $\dagger$ was polynomial and we had an expresion for it, so we could express $(\varepsilon.z)\dagger$ explicitly. I.e. there was a map Φ such that, for every f, $\Phi.f = f\dagger$. This seems a strange identity: a functor always has an arrow-part, why denote it differently? The question is: are we allowed to calculate with it inside the given category; or, is the arrow-part internalisable? A functor with the property that its arrow-part may be represented as an arrow itself is sometimes called *strong*, the theatre of operations should be a Cartesian Closed Category. For a discussion of strong functors and examples see [3]. Assuming we have such a setting and a strong functor, we resume our calculation

$$\begin{array}{ll} & (\P\P) \\ \equiv & \{ \Phi \leftrightarrow \dagger \} \{ (5) \} \\ & \phi.z \circ (\Phi \circ \varepsilon).z = \overline{\psi}.z \\ \equiv & \{ \text{ Let } \circledcirc.(f,g) = f \circ g \} \{ \vartriangle \} \\ & (\circledcirc \circ (\phi \vartriangle \Phi \circ \varepsilon)).z = \overline{\psi}.z \\ \equiv & \{ \text{ exit } z \} \{ \text{"}\overline{\ \ }\text{" is an inversion } \} \\ & \overline{\circledcirc \circ (\phi \vartriangle \Phi \circ \varepsilon)} = \psi \end{array}$$

This means that we have a general expression for ψ, provided that the functor is strong *and* assuming that $\circledcirc$ is an entity inside our categorical calculational system (strongness of the exponent functor!). The argument swap in the expression may be pushed inside, albeit (for the moment) using pointwise arguments:

$$\overline{\circledcirc \circ (\phi \vartriangle \Phi \circ \varepsilon)} = \hat{@} \circ (\phi^{\bullet} \vartriangle \overline{\Phi \circ \varepsilon})$$

here $\hat{@}$ is the lifted version of $@$: $\hat{@}.(\theta, \chi).z = @.(\theta.z, \chi.z)$. The pointwise proof is left as an easy exercise for the reader.
The extra fun of this expression is that the leftover swapped arrow $\overline{\Phi \circ \varepsilon}$ is a natural transformation that arises from a formalisation of lifting. Lifting is all over the place!
Although we didn't need this general expression in our example, it might be useful for a general theory on attribute grammar problems still to be developed.

Acknowledgements

The depth in leaves exercise, though standard, revived after a catalyzing talk on recursion of Norbert Völker on a Wednesday meeting at the Utrecht University. The material presented here benefitted from many inspiring discussions with Maarten Fokkinga and Johan Jeuring, especially with respect to catamorphising parametrised catamorphisms.

References

[0] Backhouse, R.C. and many others, A relational theory of datatypes, Notes workshop constructive algorithmics, Hollum, 1990.

[1] Fokkinga, M.M., Tupling and mutumorphisms, Squiggolist **1**, vol 4, 81-82, 1990.

[2] Fokkinga, M.M. and E. Meijer, Program calculation properties of continuous algebras, preprint 1990.

[3] Verwer, N., Categorical semantics as a basis for program transformations, in: A.J. van de Goor(ed), Proc. SION CSN90, deel 2, 539-554, 1990.