Exploiting Associativity*

Maarten M Fokkinga

It is well-known that for the standard implementation of cons-lists, as in Miranda and Lisp, the reduction to canonical form of $x \mathbin{+\!\!+} y$ (" x join y ") takes $\mathrm{size}\, x$ steps, given that x and y are in canonical form. This is because the join operation for cons-list is defined by induction on the structure of its left argument as a repeated cons. So, although $(x \mathbin{+\!\!+} y) \mathbin{+\!\!+} z = x \mathbin{+\!\!+} (y \mathbin{+\!\!+} z)$, reduction of the left-hand side takes $\mathrm{size}\, x + \mathrm{size}\, (x \mathbin{+\!\!+} y) = 2 \times \mathrm{size}\, x + \mathrm{size}\, y$ steps, but only $\mathrm{size}\, x + \mathrm{size}\, y$ steps for the right-hand side. It is therefore more efficient to compute the value of an expression $x = x_1 \mathbin{+\!\!+} x_2 \mathbin{+\!\!+} \ldots \mathbin{+\!\!+} x_n$ (with some parenthezation) by evaluating $x' = x_1 \mathbin{+\!\!+} (x_2 \mathbin{+\!\!+} (\ldots \mathbin{+\!\!+} x_n))$ instead (with the parentheses grouped to the right). Less than four years ago I gave a formal treatment of this phenomenom in a seven page note ("Elimination of Left-nesting: an example of the style of functional programming"). Nowadays, in the current status of Constructive Algorithmics and Squiggol Notation, developed by Lambert Meertens and others, it is hardly more than a simple exam question; see Theorem (1.3) below. As a show of what I have learned from Lambert —and maybe of what I still have to learn— I will discuss a generalisation of the elimination of left-nesting in full.

Preliminaries

Notation Function application is denoted by a low dot, $f.a.b = (f.a).b$. Function composition is denoted $\circ$, but when both arguments are present we also write $f \cdot g$ where the dot $\cdot$ has *lowest* binding strength and will *not* be used as an argument of another operation or function. For functions f, g and arbitrary $a : A$ we define

$$\begin{array}{lcl} f \times g.\, (x,y) & = & (f.x, g.y)\,, \\ f \vartriangle g.\, x & = & (f.x, g.x)\,, \\ a^\bullet.\, x & = & a\,. \end{array}$$

For $\oplus : A \times B \to C$ and $f : A \to (B \to C)$ we define

$$\begin{array}{lcll} a \oplus b & = & \oplus.\,(a,b)\,, & \text{i.e., conventional infix notation} \\ \hat{\oplus}.a.b & = & \oplus.\,(a,b)\,, & \text{i.e., } \hat{\oplus} : A \to (B \to C) \ \ \text{(currying)} \\ f\check{}.(a,b) & = & f.a.b\,, & \text{i.e., } f\check{} : A \times B \to C \ \ \text{(uncurrying)} \\ a\oplus & = & \oplus \cdot a^\bullet \vartriangle \mathsf{id}\,, & \text{so that } a\oplus.\, b = a \oplus b \ \ \text{(left section)} \\ \oplus b & = & \oplus \cdot \mathsf{id} \vartriangle b^\bullet\,, & \text{so that } \oplus b.\, a = a \oplus b \ \ \text{(right section)} \end{array}$$

The properties of this notation will be referred to as "calculus". Using the notation we may express associativity of $\oplus$, and 'being the identity of $\oplus$', in a variable free form:

*Published (with a entertaining lay-out) in:
Liber Amicorem Lambert Meertens, 25 years (1966–1991) at MC & CWI, January 1991, pages 24–27

$$
\begin{array}{lcl}
assoc(\oplus) & \equiv & \hat{\oplus}\cdot\oplus \;=\; \circ\cdot\hat{\oplus}\times\hat{\oplus}\,, \\
unit(e,\oplus) & \equiv & \oplus e \;=\; \oplus\cdot \mathsf{id}\vartriangle e^{\bullet} \;=\; \mathsf{id} \;=\; \oplus\cdot e^{\bullet}\vartriangle \mathsf{id} \;=\; e\oplus\,.
\end{array}
$$

The reader may check $assoc(\oplus)$ by applying the functions to $((x,y),z)$.

Structures Let A be a type. The data type of *structures over* A is given by

type: $A*$

operations: $\tau : A \to A*$, $\quad +\!\!\!+ : A*\times A* \to A*$, $\quad \Box : A*$

laws: $unit(\Box, +\!\!\!+), \ldots$.

(τ yields a "singleton" structure, $+\!\!\!+$ "joins" two structures, and $\Box$ is the "empty" structure.) With only the law $unit(\Box, +\!\!\!+)$ the structures are called *trees*. If in addition $assoc(+\!\!\!+)$ is postulated, then the structures are called *join-lists*. We shall use postfix symbol $_{\mathrm{T}}$ and $_{\mathrm{L}}$ to distinguish between trees and lists; if no subscript is used we mean generically either of them.

For $f : A \to B$ and $\oplus : B \times B \to B$ the function $(\!|f,\oplus|\!) : A* \to B$, called *catamorphism*, is defined to be the unique function satisfying

$$
\begin{array}{lcl}
(\!|f,\oplus|\!)\cdot\tau & = & f \\
(\!|f,\oplus|\!)\cdot +\!\!\!+ & = & \oplus\cdot(\!|f,\oplus|\!)\times(\!|f,\oplus|\!) \\
(\!|f,\oplus|\!).\,\Box & = & \text{identity of } \oplus\,.
\end{array}
$$

(We assume that any binary operation has an identity.) Thus the effect of $(\!|f,\oplus|\!)$ might be described as the systematic replacement $\tau, +\!\!\!+, \Box \;:=\; f, \oplus, \mathrm{unit}(\oplus)$. It is easy to see that $assoc(+\!\!\!+)$ implies associativity of $\oplus$ on the range of $(\!|f,\oplus|\!)$, so that $(\!|f,\oplus|\!)_{\mathrm{L}}$ does not make sense when $\oplus$ is not associative on its range. Specific functions are

$$
\begin{array}{lcll}
\oplus/ & = & (\!|\mathsf{id},\oplus|\!) : A* \to A & \text{for } \oplus : A\times A \to A \quad \text{reduce} \\
\overline{\oplus} & = & (\!|\hat{\oplus},\circ|\!)^{\smile} : A*\times B \to B & \text{for } \oplus : A\times B \to B \quad \text{right-reduce.}
\end{array}
$$

For example, for $x = (\tau.a +\!\!\!+ \tau.b) +\!\!\!+ \tau.c$ we have $\oplus/.x = (a\oplus b)\oplus c$ with the same parenthezation as in x, and $x\,\overline{\oplus} = (a\oplus\cdot b\oplus)\cdot c\oplus$ so that $x\,\overline{\oplus}\,d = a\oplus(b\oplus(c\oplus d))$ with the parentheses grouped to the right! Part of the observation of the introduction amounts to the assertion that for any tree x we have $+\!\!\!+_{\mathrm{L}}/.x = x\,\overline{+\!\!\!+_{\mathrm{L}}}\,\Box_{\mathrm{L}}$. It is this claim, as well as a generalisation, that we shall prove below. In the proof we use two important properties of catamorphisms:

$$
\begin{array}{lclll}
f\cdot(\!|g,\oplus|\!) & = & (\!|f\cdot g,\;\otimes|\!) & \text{if}\quad f\cdot\oplus = \otimes\cdot f\times f & \text{Promotion} \\
f\cdot\overline{\oplus} & = & \overline{\otimes}\cdot\mathsf{id}\times f & \text{if}\quad f\cdot\oplus = \otimes\cdot\mathsf{id}\times f & \text{r-reduce Prom.}
\end{array}
$$

The reader may prove these properties by induction on the structure of the argument; (actually they are a simple consequence of the definition of the notion of data type — which we have not given here).

Proving the claims

Lemma 1 *For associative* $\oplus$ *we have* $\quad \hat{\oplus}\cdot(\!|f,\oplus|\!) = (\!|\hat{\oplus}\cdot f,\;\circ|\!)$.

The proof is trivial indeed:

$$
\begin{array}{ll}
 & \hat{\oplus} \cdot (\!|f, \oplus|\!) = (\!|\hat{\oplus} \cdot f,\ \circ|\!) \\
\Leftarrow & \qquad \text{Promotion} \\
 & \hat{\oplus} \cdot \oplus = \circ \cdot \hat{\oplus} \times \hat{\oplus} \\
\equiv & \qquad \mathit{assoc}(\oplus) \\
 & \mathsf{true}\ .
\end{array}
$$

Teorem 1 *Suppose* $\mathit{assoc}(\oplus)$ *and* $\mathit{unit}(e, \oplus)$. *Then*

$$
\begin{array}{llll}
1. & (\!|f, \oplus|\!) & = & \overline{(\hat{\oplus} \cdot f)}\, e \\
2. & \oplus/ & = & \overline{\oplus}\, e \\
3. & +\!\!+_{\mathrm{L}}/ & = & \overline{+\!\!+_{\mathrm{L}}}\, \square_{\mathrm{L}}
\end{array}
$$

To prove Part 1 we argue

$$
\begin{array}{ll}
 & (\!|f, \oplus|\!) = \overline{(\hat{\oplus} \cdot f)} e \\
\equiv & \qquad \text{lhs: } \mathit{unit}(e, \oplus)\text{ ; rhs: unfold and calculus} \\
 & \oplus \cdot \mathsf{id} \vartriangle e^{\bullet} \cdot (\!|f, \oplus|\!) = (\!|\hat{\oplus} \cdot f,\ \circ|\!)^{\smile} \cdot \mathsf{id} \vartriangle e^{\bullet} \\
\Leftarrow & \qquad \text{Leibniz (and calculus)} \\
 & \hat{\oplus} \cdot (\!|f, \oplus|\!) = (\!|\hat{\oplus} \cdot f,\ \circ|\!) \\
\equiv & \qquad \text{preceding lemma, } \mathit{assoc}(\oplus) \\
 & \mathsf{true}\ .
\end{array}
$$

Part 2 follows from Part 1 by substituting $f := \mathsf{id}$. Part 3 is an instantiation of Part 2.

* * *

In general, when $\mathit{assoc}(\oplus)$ holds and $\oplus/$ is to be computed, one might wish to restructure the parenthezation of the argument by a specific transformation ε and then evaluate $\oplus/\ \cdot\ \varepsilon$. To this end define the *listifying* function

$$
\mathit{lfy} \quad = \quad (\!|\tau_{\mathrm{L}}, +\!\!+_{\mathrm{L}}|\!) \quad : A* \to A *_{\mathrm{L}} \ .
$$

Then $\mathit{lfy}.x$ is the list of tip-values of x in "left to right" order, irrespective of the parenthezation within x ; and $\mathit{lfy}.x = x$ for any list x .

Teorem 2 *Suppose* $\mathit{lfy} \cdot \varepsilon = \mathit{lfy}$ *and* $\mathit{assoc}(\oplus)$. *Then* $(\!|f, \oplus|\!) \cdot \varepsilon = (\!|f, \oplus|\!)$.

To prove this we first observe that $(\!|f, \oplus|\!)_{\mathrm{L}}$ makes sense since $\oplus$ is associative, and that

$$
\begin{array}{ll}
 & (\!|f, \oplus|\!)_{\mathrm{L}} \cdot \mathit{lfy} = (\!|f, \oplus|\!) \\
\Leftarrow & \qquad \text{unfold } \mathit{lfy}\text{, Promotion} \\
 & (\!|f, \oplus|\!)_{\mathrm{L}} \cdot \tau_{\mathrm{L}} = f \qquad \text{and} \qquad (\!|f, \oplus|\!)_{\mathrm{L}} \cdot +\!\!+_{\mathrm{L}} = \oplus \cdot (\!|f, \oplus|\!)_{\mathrm{L}} \times (\!|f, \oplus|\!)_{\mathrm{L}} \\
\equiv & \qquad \text{definition catamorphism} \\
 & \mathsf{true}\ .
\end{array}
$$

Now we calculate

$$
\begin{array}{ll}
& (\!|f,\oplus|\!) \cdot \varepsilon \\
= & \qquad \text{above observation (right to left)} \\
& (\!|f,\oplus|\!)_{\mathrm{L}} \cdot \mathit{lfy} \cdot \varepsilon \\
= & \qquad \text{premiss} \\
& (\!|f,\oplus|\!)_{\mathrm{L}} \cdot \mathit{lfy} \\
= & \qquad \text{above observation again} \\
& (\!|f,\oplus|\!)
\end{array}
$$

as desired.

The particular parenthezation enforced by a right-reduce is given by

$$\varepsilon_r \;=\; \overline{(\hat{+\!\!+}_{\mathrm{T}} \cdot \tau_{\mathrm{T}})} \cdot \mathsf{id} \vartriangle \Box^{\bullet}_{\mathrm{T}} \;=\; \overline{(\hat{+\!\!+}_{\mathrm{T}} \cdot \tau_{\mathrm{T}})}\, \Box_{\mathrm{T}} \quad : \quad A* \to A*_{\mathrm{T}}\ .$$

This claim is formalized and proved in Part 2 of the following theorem. Part 3 shows that ε_r satisfies the condition of Theorem (2), thus enabling an alternative proof of Theorem (1.2).

Teorem 3

1. $(\!|f,\oplus|\!) \cdot \varepsilon_r \;=\; \overline{(\hat{\oplus} \cdot f)}\, e$ if $\mathit{unit}(e,\oplus)$ (not necessarily $\mathit{assoc}(\oplus)$!).
2. $\oplus/ \cdot \varepsilon_r \;=\; \overline{\oplus}\, e$ if $\mathit{unit}(e,\oplus)$.
3. ε_r satisfies $\mathit{lfy} \cdot \varepsilon_r \;=\; \mathit{lfy}$.

To prove the first part we argue

$$
\begin{array}{ll}
& (\!|f,\oplus|\!) \cdot \varepsilon_r \;=\; \overline{(\hat{\oplus} \cdot f)}\, e \\
\equiv & \qquad \text{calculus} \\
& (\!|f,\oplus|\!) \cdot \varepsilon_r \;=\; \overline{(\hat{\oplus} \cdot f)} \cdot \mathsf{id} \times \mathsf{id} \cdot \mathsf{id} \vartriangle e^{\bullet} \\
\equiv & \qquad \text{equation for catamorphism on } \Box \text{ using } \mathit{unit}(e,\oplus) \\
& (\!|f,\oplus|\!) \cdot \varepsilon_r \;=\; \overline{(\hat{\oplus} \cdot f)} \cdot \mathsf{id} \times (\!|f,\oplus|\!) \cdot \mathsf{id} \vartriangle \Box^{\bullet} \\
\Leftarrow & \qquad \text{unfold } \varepsilon_r \text{, and Promotion for right-reduce} \\
& (\!|f,\oplus|\!) \cdot \hat{+\!\!+} \cdot \tau \;=\; \hat{\oplus} \cdot f \cdot \mathsf{id} \times (\!|f,\oplus|\!) \\
\equiv & \qquad \text{equations for catamorphism on } +\!\!+ \text{ and } \tau \\
& \mathsf{true}\ .
\end{array}
$$

Instantiating Part 1 with $f := \mathsf{id}$ gives Part 2. For Part 3 we calculate

$$
\begin{array}{ll}
& \mathit{lfy} \cdot \varepsilon_r \\
= & \qquad \text{Part 1 with } f,\oplus,e := \tau_{\mathrm{L}}, +\!\!+_{\mathrm{L}}, \Box_{\mathrm{L}} \\
& \overline{(\hat{+\!\!+}_{\mathrm{L}} \cdot \tau_{\mathrm{L}})}\, \Box \\
= & \qquad \text{Theorem (1.1) noting } \mathit{assoc}(+\!\!+_{\mathrm{L}}) \text{ and } \mathit{unit}(\Box, +\!\!+) \\
& \mathit{lfy}
\end{array}
$$

as desired. ∎