2008 Formosan Summer School on Logic, Language, and Computation

Program Construction and Reasoning
Exercises for Day 2

Shin-Cheng Mu

July 7th, 2008

1 In-Class Exercises

1.1 Folds and Fold-Fusion

1. Given functions $f :: \alpha \to \beta$ and $g :: \alpha \to \gamma$, $split\, f\ g :: \alpha \to (\beta, \gamma)$ is a function defined by:

$$split\, f\ g\ a = (f\ a, g\ a).$$

Recall the definition of *steep* and *sum*. The definition of *steepsum* can be re-written as:

$$steepsum \quad = \quad split\ steep\ sum.$$

Also recall that the identity function *id* on lists is a fold: $id = foldr\,(:)\,[\,]$. Use the fold-fusion theorem to fuse $steepsum \cdot id$ into one fold.

Ans:

We reason:

$$\begin{aligned}
& steepsum \\
= \quad & \{ \text{ since } f = f \cdot id \ \} \\
& steepsum \cdot id \\
= \quad & \{ \text{ since } id = foldr\,(:)\,[\,] \ \} \\
& steepsum \cdot foldr\,(:)\,[\,] \\
= \quad & \{ \ foldr\text{-fusion, see below } \} \\
& foldr\ step\ (true, 0).
\end{aligned}$$

To perform *foldr*-fusion, we construct a function *step* such that:

$$steepsum\,((:)\ x\ xs) \quad = \quad step\ x\ (steepsum\ xs).$$

We reason:

$$
\begin{array}{cl}
 & \mathit{steepsum}\,(x{:}xs) \\
= & \quad \{\ \text{def. of } \mathit{steepsum} \text{ and } \mathit{split}\ \ \} \\
 & (\mathit{steep}\,(x{:}xs), \mathit{sum}\,(x{:}xs)) \\
= & \quad \{\ \text{def. of } \mathit{steep} \text{ and } \mathit{sum}\ \} \\
 & (\mathit{steep}\,xs\ \wedge\ x > \mathit{sum}\,xs, x + \mathit{sum}\,xs) \\
= & \quad \{\ \text{introducing local identifiers}\ \} \\
 & \mathbf{let}\,(st, ss) = (\mathit{steep}\,xs, \mathit{sum}\,xs) \\
 & \mathbf{in}\,(st\ \wedge\ x > ss, x + ss) \\
= & \quad \{\ \text{let } \mathit{step}\,x\,(st, ss) = (st\ \wedge\ x > ss, x + ss)\ \} \\
 & \mathit{step}\,x\,(\mathit{steep}\,xs, \mathit{sum}\,xs) \\
= & \quad \{\ \text{def. of } \mathit{steepsum}\ \} \\
 & \mathit{step}\,x\,(\mathit{steepsum}\,xs).
\end{array}
$$

We have thus derived:

$$
\begin{array}{l}
\mathit{steepsum} = \mathit{foldr}\ \mathit{step}\,(\mathit{true}, 0) \\
\qquad \mathbf{where}\ \mathit{step}\,x\,(st, ss) = (st\ \wedge\ x > ss, x + ss).
\end{array}
$$

2. Recall the definition of *scanr* from the lecture:

$$\mathit{scanr}\,f\,e \quad = \quad \mathit{map}\,(\mathit{foldr}\,f\ e) \cdot \mathit{tails}$$

and its implementation as a fold:

$$
\begin{array}{l}
\mathit{scanr}\,f\ e = \mathit{foldr}\,(\mathit{sc}\,f)\,[e] \\
\qquad\qquad \mathbf{where}\ \mathit{sc}\,f\ x\,(y{:}ys) = f\ x\ y : y : ys
\end{array}
$$

(a) Expand $\mathit{scanr}\,(+)\,0\,[1, 2, 3]$ step by step:

$$
\begin{array}{cl}
 & \mathit{scanr}\,(+)\,0\,[1, 2, 3] \\
= & \mathit{foldr}\,(\mathit{sc}\,(+))\,[0]\,[1, 2, 3] \\
= & \ldots
\end{array}
$$

Ans:

$$
\begin{array}{cl}
= & \mathit{sc}\,(+)\,1\,(\mathit{foldr}\,(\mathit{sc}\,(+))\,[0]\,[2, 3]) \\
= & \mathit{sc}\,(+)\,1\,(\mathit{sc}\,(+)\,2\,(\mathit{foldr}\,(\mathit{sc}\,(+))\,[0]\,[3])) \\
= & \mathit{sc}\,(+)\,1\,(\mathit{sc}\,(+)\,2\,(\mathit{sc}\,(+)\,3\,(\mathit{foldr}\,(\mathit{sc}\,(+))\,[0]\,[\,]))) \\
= & \mathit{sc}\,(+)\,1\,(\mathit{sc}\,(+)\,2\,(\mathit{sc}\,(+)\,3\,[0])) \\
= & \mathit{sc}\,(+)\,1\,(\mathit{sc}\,(+)\,2\,[3, 0]) \\
= & \mathit{sc}\,(+)\,1\,[5, 3, 0] \\
= & [6, 5, 3, 0]
\end{array}
$$

(b) Derive the implementation of $\mathit{scanr}\, f\ e$ by fusing $\mathit{map}\,(\mathit{foldr}\, f\ e) \cdot \mathit{tails}$ into one fold.

Ans:

$$
\begin{aligned}
& \mathit{map}\,(\mathit{foldr}\, f\ e) \cdot \mathit{tails} \\
= & \quad \{ \text{ since } \mathit{tails} \text{ is a fold } \} \\
& \mathit{map}\,(\mathit{foldr}\, f\ e) \cdot \mathit{foldr}\ \mathit{til}\,[[\,]] \\
= & \quad \{ \mathit{foldr}\text{-fusion, see below } \} \\
& \mathit{foldr}\,(\mathit{sc}\, f)\,[[e]].
\end{aligned}
$$

Recall the definition of *til*:

$$\mathit{til}\, x\,(\mathit{ys}{:}\mathit{yss}) = (x : \mathit{ys}) : \mathit{ys} : \mathit{yss}.$$

This fusion condition is proved below:

$$
\begin{aligned}
& \mathit{map}\,(\mathit{foldr}\, f\ e)\,(\mathit{til}\, x\,(\mathit{ys} : \mathit{yss})) \\
= & \quad \{ \text{ def. of } \mathit{til} \ \} \\
& \mathit{map}\,(\mathit{foldr}\, f\ e)\,((x : \mathit{ys}) : \mathit{ys} : \mathit{yss}) \\
= & \quad \{ \text{ def. of } \mathit{map} \ \} \\
& \mathit{foldr}\, f\ e\,(x : \mathit{ys}) : \mathit{foldr}\, f\ e\ \mathit{ys} : \mathit{map}\,(\mathit{foldr}\, f\ e)\ \mathit{yss} \\
= & \quad \{ \text{ def. of } \mathit{foldr} \ \} \\
& f\ x\,(\mathit{foldr}\, f\ e\ \mathit{ys}) : \mathit{foldr}\, f\ e\ \mathit{ys} : \mathit{map}\,(\mathit{foldr}\, f\ e)\ \mathit{yss} \\
= & \quad \{ \text{ introducing local identifiers } \} \\
& \mathbf{let}\,(\mathit{ys}, \mathit{yss}) = (\mathit{foldr}\, f\ e\ \mathit{ys}, \mathit{map}\,(\mathit{foldr}\, f\ e)\ \mathit{yss}) \\
& \mathbf{in}\, f\ x\ \mathit{ys}{:}\ \mathit{ys}{:}\ \mathit{yss} \\
= & \quad \{ \text{ let } \mathit{sc}\, f\ x\,(\mathit{ys}{:}\mathit{yss}) = f\ x\ \mathit{ys} : \mathit{ys} : \mathit{yss} \ \} \\
& \mathit{sc}\, f\ x\,(\mathit{fold}\, f\ e\ \mathit{ys} : \mathit{map}\,(\mathit{foldr}\, f\ e)\ \mathit{yss}) \\
= & \quad \{ \text{ def. of } \mathit{map} \ \} \\
& \mathit{sc}\, f\ x\,(\mathit{map}\,(\mathit{foldr}\, f\ e)\,(\mathit{ys} : \mathit{yss})).
\end{aligned}
$$

We have therefore derived:

$$\mathit{scanr}\, f\ e = \mathit{foldr}\,(\mathit{sc}\, f)\,[[e]],$$

where $\mathit{sc}\, f\ x\,(\mathit{ys}{:}\mathit{yss}) = f\ x\ \mathit{ys} : \mathit{ys} : \mathit{yss}$.

2 Take-Home Exercise (Due Date: July 10th)

You do not have to do the exercises below if you have completed any of the exercises from Day 1. Exercise 1 is worth 40 points while exercise 2 is worth 50 points.

1. The function $\mathit{filter}\, p$ selects from a list all elements satisfying a predicate p. For example, $\mathit{filter}\ \mathit{even}\,[1, 2, 3, 4] = [2, 4]$.

(a) Give a recursive definition of *filter*:

$$\begin{array}{lcl} \mathit{filter}\ p\ [\,] & = & \ldots \\ \mathit{filter}\ p\ (x{:}xs) & = & \ldots \end{array}$$

Ans:

$$\begin{array}{lcl} \mathit{filter}\ p\ [\,] & = & [\,] \\ \mathit{fliter}\ p\ (x{:}xs) & = & \mathbf{if}\ \ p\ x\ \mathbf{then}\ x{:}\mathit{filter}\ p\ xs\ \mathbf{else}\ \mathit{filter}\ p\ xs \end{array}$$

(b) Define *filter* p in terms of *foldr*.

Ans:

$$\begin{array}{lcl} \mathit{fliter}\ p\ xs & = & \mathit{foldr}\ (\mathit{flt}\ p)\ [\,]\ xs \\ \mathit{flt}\ p\ x\ ys & = & \mathbf{if}\ \ p\ x\ \mathbf{then}\ x{:}ys\ \mathbf{else}\ ys \end{array}$$

(c) Prove, by fold-fusion, that

$$\mathit{filter}\ p \cdot \mathit{map}\ f \quad = \quad \mathit{map}\ f \cdot \mathit{filter}\ (p \cdot f).$$

Hint: apply fold-fusion on both sides, and show that they are equal to the same fold.

Ans:

Consider the left-hand side:

$$\begin{array}{ll} & \mathit{filter}\ p \cdot \mathit{map}\ f \\ = & \quad \{\ \text{since } \mathit{map} \text{ is a fold}\ \} \\ & \mathit{filter}\ p \cdot \mathit{foldr}\ \mathit{mp}\ [\,]\ \mathbf{where}\ \mathit{mp}\ x\ xs = f\ x{:}xs \end{array}$$

We now attempt to construct a function *fltf* that satisfies the fusion condition:

$$\mathit{filter}\ p\ (\mathit{mf}\ f\ x\ xs) \quad = \quad \mathit{fltf}\ x\ (\mathit{filter}\ p\ xs)$$

We reason:

$$\begin{array}{ll} & \mathit{fliter}\ p\ (f\ x{:}xs) \\ = & \quad \{\ \text{def. of } \mathit{filter}\ \} \\ & \mathbf{let}\ ys = \mathit{filter}\ p\ xs\ \mathbf{in}\ \mathbf{if}\ \ p\ (f\ x)\ \mathbf{then}\ f\ x{:}ys\ \mathbf{else}\ ys \\ = & \quad \{\ \text{let } \mathit{fltf}\ x\ ys = \mathbf{if}\ \ p\ (f\ x)\ \mathbf{then}\ f\ x{:}ys\ \mathbf{else}\ ys\ \} \\ & \mathit{fltf}\ x\ (\mathit{filter}\ p\ xs) \end{array}$$

We have thus shown that:

$$\mathit{filter}\ p \cdot \mathit{map}\ f = \mathit{foldr}\ \mathit{fltf}\ (\mathit{filter}\ p\ [\,]) = \mathit{foldr}\ \mathit{fltf}\ [\,]$$

where *fltf* is defined by $\mathit{fltf}\ x\ ys = \mathbf{if}\ \ p\ (f\ x)\ \mathbf{then}\ f\ x{:}ys\ \mathbf{else}\ ys$.

Now consider the right-hand side:

$$\begin{array}{ll} & \mathit{map}\ f \cdot \mathit{filter}\ (p \cdot f) \\ = & \quad \{\ \text{write } \mathit{fliter} \text{ as a fold}\ \} \end{array}$$

$$
\begin{array}{ll}
& \mathit{map}\, f \cdot \mathit{foldr}\, (\mathit{fltf}\, (p \cdot f))\, [\,] \\
= & \quad \{ \text{ fold fusion (see below) and } \mathit{map}\, f\, [\,] = [\,] \ \} \\
& \mathit{foldr}\, \mathit{fltf}\, [\,]
\end{array}
$$

This fold fusion condition is proved below:

$$
\begin{array}{ll}
& \mathit{map}\, f\, (\mathit{fltf}\ x\ \mathit{ys}) \\
= & \quad \{ \text{ def. of } \mathit{fltf} \ \} \\
& \mathit{map}\, f\, (\mathbf{if}\ \ p\, (f\ x)\ \mathbf{then}\ x{:}\mathit{ys}\ \mathbf{else}\ \mathit{ys}) \\
= & \quad \{ \text{ since } \mathit{map} \text{ distributes into } \mathbf{if} \ \ \} \\
& \mathbf{if}\ \ p\, x\ \mathbf{then}\ \mathit{map}\, f\, (x{:}\mathit{ys})\ \mathbf{else}\ \mathit{map}\, f\ \mathit{ys} \\
= & \quad \{ \text{ def. of } \mathit{map} \ \} \\
& \mathbf{if}\ \ p\, (f\ x)\ \mathbf{then}\ f\ x : \mathit{map}\, f\ \mathit{ys}\ \mathbf{else}\ \mathit{map}\, f\ \mathit{ys} \\
= & \quad \{ \text{ def. of } \mathit{fltf} \ \} \\
& \mathit{fltf}\ x\, (\mathit{map}\, f\ \mathit{ys})
\end{array}
$$

2. Given two functions h_1 and h_2, the function $\mathit{split}\, h_1\, h_2$ computes the pair of their results:

$$\mathit{split}\, h_1\, h_2\, \mathit{xs} \quad = \quad (h_1\, \mathit{xs}, h_2\, \mathit{xs}).$$

In the special case when both h_1 and h_2 are defined by *foldr*:

$$
\begin{array}{lll}
h_1 & = & \mathit{foldr}\, f_1\, e_1, \\
h_2 & = & \mathit{foldr}\, f_2\, e_2,
\end{array}
$$

the following "banana-split" rule allows us to express $\mathit{split}\, h_1\, h_2$ using one single *foldr*:

$$
\begin{array}{l}
\mathit{split}\, h_1\, h_2 = \mathit{foldr}\, g\, (e_1, e_2), \\
\qquad\qquad \mathbf{where}\ g\, x\, (y, z) = (f_1\ x\ y, f_2\ x\ z).
\end{array}
$$

It optimises two traversal through the list to only one traversal. It is called "banana-split" because folds used to be written using a notation called "banana brackets".

(a) The function *split sum length* return the pair of sum and length of the input list. Use the banana-split rule to express *split sum length* by a fold.

Ans:

Since

$$
\begin{array}{lll}
\mathit{sum} & = & \mathit{foldr}\, (+)\, 0 \\
\mathit{length} & = & \mathit{foldr}\, (\lambda x\, z\, .\, (z + 1))\, 0
\end{array}
$$

let $f_1 = (+), f_2 = (\lambda x\, z.\, (z+1))$, $e_1 = e_2 = 0$, we have:

$$\begin{array}{lcl} \mathit{split}\; \mathit{sum}\; \mathit{length} & = & \mathit{foldr}\; g\, (0,0) \\ g\, x\, (y,z) & = & (x+y, z+1) \end{array}$$

(b) Prove the banana-split rule by fold fusion. Hint: recall that $\mathit{split}\; h_1\; h_2 = \mathit{split}\; h_1\; h_2 \cdot \mathit{id}$, and id is a fold.

Ans:

$$\begin{array}{ll} & \mathit{split}\; h_1\; h_2 = \mathit{split}\; h_1\; h_2 \cdot \mathit{id} \\ = & \quad \{ \text{ write } \mathit{id} \text{ as a fold } \} \\ & \mathit{split}\; h_1\; h_2 \cdot \mathit{foldr}\; (:)\; [\,] \end{array}$$

Now we try to fuse $\mathit{split}\; h_1\; h_2 \cdot \mathit{foldr}\; (:)\; [\,]$ into one fold. We have to show that g satisfies:

$$\mathit{split}\; h_1\; h_2\; ((:)\; x\; xs) \quad = \quad g\; x\; (\mathit{split}\; h_1\; h_2\; xs)$$

which is proved below:

$$\begin{array}{ll} & \mathit{split}\; h_1\; h_2\; (x{:}xs) \\ = & \quad \{ \text{ def. of } \mathit{split} \cdot\cdot \} \\ & (h_1\; (x{:}xs), h_2\; (x{:}xs)) \\ = & \quad \{ \text{ def. of } h_1 \text{ and } h_2 \} \\ & (\mathit{foldr}\; f_1\; e_1\; (x{:}xs), \mathit{foldr}\; f_2\; e_2\; (x{:}xs)) \\ = & \quad \{ \text{ def. of } \mathit{foldr} \} \\ & (f_1\; x\; (\mathit{foldr}\; f_1\; e_1\; xs), f_2\; x\; (\mathit{foldr}\; f_2\; e_2\; xs)) \\ = & \quad \{ \text{ by def., } g\; x\; (y,z) = (f_1\; x\; y, f_2\; x\; z) \} \\ & g\; x\; (\mathit{foldr}\; f_1\; e_1\; xs, \mathit{foldr}\; f_2\; e_2\; xs) \\ = & \quad \{ \text{ def. of } h_1 \text{ and } h_2 \} \\ & g\; x\; (h_1\; xs, h_2\; xs) \\ = & \quad \{ \text{ def. of } \mathit{split} \cdot\cdot \} \\ & g\; x\; (\mathit{split}\; h_1\; h_2\; xs) \end{array}$$

Back to $\mathit{split}\; h_1\; h_2$:

$$\begin{array}{ll} & \mathit{split}\; h_1\; h_2 \\ = & \quad \{ \text{ reasoning above } \} \\ & \mathit{foldr}\; g\; (\mathit{split}\; h_1\; h_2\; [\,]) \\ = & \quad \{ \text{ def. of } h_1, h_2 \} \\ & \mathit{foldr}\; g\; (e_1, e_2) \end{array}$$