Appendix A

MFP-I/0102: Addenda to the Lectures Notes

A.1 T 2001.10.11 — Cata-fusion/reflexion

Given the following VDM-SL definition of list inversion

```
invl[@A] : seq of @A -> seq of @A
invl(l) == if l = [] then l else invl[@A](tl l) ^ [hd l] ;
```

we want to prove $invl$ involution:

$$invl \cdot invl \quad = \quad id$$

We express $invl$ as a catamorphism

$$invl \quad \stackrel{\mathrm{def}}{=} \quad (\!|\underbrace{[\,\underline{[\,]}, \hat{} \cdot swap \cdot (singl \times id)\,]}_{g}|\!) \tag{A.1}$$

and assume the following properties:

$$invl \cdot \hat{} \quad = \quad \hat{} \cdot (invl \times invl) \cdot swap \tag{A.2}$$
$$invl \cdot singl \quad = \quad singl \tag{A.3}$$
$$\hat{} \cdot (singl \times id) \quad = \quad cons \tag{A.4}$$

Then we base our reasoning on properties available from the *cata-toolbox* (no need for induction):

$$invl \cdot invl = id$$
$$\equiv \qquad \{ \text{ cata-reflexion (2.58) e (A.1)} \}$$
$$invl \cdot (\!|g|\!) = (\!|in|\!)$$

$$
\begin{array}{cl}
\Leftarrow & \quad \{ \text{cata-fusion (2.61)} \} \\
& invl \cdot g = in \cdot (id + id \times invl) \\
\equiv & \quad \{ \text{expanding } g \text{ (A.1)} \} \\
& invl \cdot [\, \underline{[\,]}, \hat{} \cdot swap \cdot (singl \times id) \,] = in \cdot (id + id \times invl) \\
\equiv & \quad \{ \text{+-fusion (1.40)} \} \\
& [\, invl \cdot \underline{[\,]}, invl \cdot \hat{} \cdot swap \cdot (singl \times id) \,] = in \cdot (id + id \times invl) \\
\equiv & \quad \{ \text{by (A.2)} \} \\
& [\, \underline{[\,]}, \hat{} \cdot (invl \times invl) \cdot swap \cdot swap \cdot (singl \times id) \,] = in \cdot (id + id \times invl) \\
\equiv & \quad \{ swap \text{ involution and } \times\text{-bifunctor} \} \\
& [\, \underline{[\,]}, \hat{} \cdot (invl \cdot singl \times invl) \,] = in \cdot (id + id \times invl) \\
\equiv & \quad \{ \text{by (A.3)} \} \\
& [\, \underline{[\,]}, \hat{} \cdot (singl \times invl) \,] = in \cdot (id + id \times invl) \\
\equiv & \quad \{ \text{by (A.4) and } \times\text{-bifunctor} \} \\
& [\, \underline{[\,]}, cons \cdot (id \times invl) \,] = in \cdot (id + id \times invl) \\
\equiv & \quad \{ \text{(reverse) +-absorption (1.41)} \} \\
& [\, \underline{[\,]}, cons \,] \cdot (id + id \times invl) = in \cdot (id + id \times invl) \\
\equiv & \quad \{ \text{definition of } in \} \\
& TRUE
\end{array}
$$

A.2 T 2001.10.18 - Cata-absorption

Proof (deduction) by cata-absorption:

$$
\begin{array}{cl}
& sum \cdot \underline{1}^\star \\
= & \quad \{ sum \text{ is a catamorphism} \} \\
& (\!|\, [\, \underline{0}, add \,] \,|\!) \cdot \underline{1}^\star \\
= & \quad \{ \text{cata-absorption (2.67) for the } \_^\star \text{ functor} \} \\
& (\!|\, [\, \underline{0}, add \,] \cdot (id + \underline{1} \times id) \,|\!) \\
\equiv & \quad \{ \text{by } +\text{-}\textit{fusion} \text{ (1.40)} \} \\
& (\!|\, [\, \underline{0}, add \cdot (\underline{1} \times id) \,] \,|\!) \\
= & \quad \{ \text{by (1.22)} \}
\end{array}
$$

$$
\begin{array}{ll}
& (\![\, \underline{0}, add \cdot (\langle \underline{1}, \pi_2 \rangle)\,]\!) \\
= & \quad \{ \text{ definition of } suc \} \\
& (\![\, \underline{0}, suc \cdot \pi_2 \,]\!) \\
= & \quad \{ \text{ definition of } length \} \\
& length
\end{array}
$$

Exercise 1.1 Adapt the previous calculation to that of counting the number of leaves of a leaf-tree. (Indeed, you can generalize the calculation to an arbitrary, polynomial data type.)
□

Exercise 1.2 For $2 \xleftarrow{p} A$ a predicate, define the "p-filter" operator as follows

$$filter\, p \quad \stackrel{\text{def}}{=} \quad dconc \cdot (p \rightarrow singl, \underline{[\,]})^{\star}$$

where

$$
\begin{array}{rcl}
dconc & = & (\![\, \underline{[\,]}, conc \,]\!) \\
conc(a, b) & = & a \mathbin{\hat{}} b
\end{array}
$$

1. Proceed by cata-absorption to calculate a pointwise definition of $filter\, p$
   `NB:` assume the following fact:
   $$(p \rightarrow f, g) \times h \quad = \quad p \cdot \pi_1 \rightarrow f \times h, g \times h$$
2. Write it down in (pointwise) VDM-SL notation.

□

A.3 2001.10.18 — Mutual recursion

Consider mutually-dependent f and g as follows:

```
f: nat -> nat
f(n) == if n = 0 then 0 else g(n - 1);

g: nat -> nat
g(n) == if n = 0 then 1 else f(n - 1) + g(n - 1);
```

How we reason about mutually-dependent functions?

The situation is handled by the so-called *mutual-recursion law*, also called "Fokkinga law":

$$
\begin{array}{c}
f \cdot in = h \cdot \mathsf{F}\, \langle f, g \rangle \\
\wedge \\
g \cdot in = k \cdot \mathsf{F}\, \langle f, g \rangle
\end{array}
\quad \Rightarrow \quad \langle f, g \rangle = (\![\langle h, k \rangle]\!) \tag{A.5}
$$

In terms of diagrams: from

$$
\begin{tikzcd}
\mathsf{T} \arrow[d, "f"'] & \mathsf{F}\,\mathsf{T} \arrow[l, "in"'] \arrow[d, "\mathsf{F}\,\langle f,g\rangle"] \\
A & \mathsf{F}\,(A \times B) \arrow[l, "h"]
\end{tikzcd}
\qquad
\begin{tikzcd}
\mathsf{T} \arrow[d, "g"'] & \mathsf{F}\,\mathsf{T} \arrow[l, "in"'] \arrow[d, "\mathsf{F}\,\langle f,g\rangle"] \\
B & \mathsf{F}\,(A \times B) \arrow[l, "k"]
\end{tikzcd}
$$

we get

$$
\begin{tikzcd}
\mathsf{T} \arrow[d, "\langle f,g\rangle"'] & \mathsf{F}\,\mathsf{T} \arrow[l, "in"'] \arrow[d, "\mathsf{F}\,\langle f,g\rangle"] \\
A \times B & \mathsf{F}\,(A \times B) \arrow[l, "\langle h,k\rangle"]
\end{tikzcd}
$$

Proof:

$$
\begin{array}{ll}
& \langle f, g\rangle \cdot in = \langle h, k\rangle \cdot \mathsf{F}\,\langle f, g\rangle \\
\equiv & \qquad \{ \text{ by } \times\text{-fusion (1.24)} \} \\
& \langle f, g\rangle \cdot in = \langle h \cdot \mathsf{F}\,\langle f, g\rangle, k \cdot \mathsf{F}\,\langle f, g\rangle\rangle \\
\equiv & \qquad \{ \text{ by hypothesis} \} \\
& \langle f, g\rangle \cdot in = \langle f \cdot in, g \cdot in\rangle \\
\equiv & \qquad \{ \text{ by (reverse) } \times\text{-fusion (1.24)} \} \\
& \langle f, g\rangle \cdot in = \langle f, g\rangle \cdot in \\
\equiv & \qquad \{ \text{ equality is reflexive} \} \\
& TRUE
\end{array}
$$

Applying this to the above pair of f and g:

$$
\begin{array}{rcl}
f \cdot [\, \underline{0}, suc \,] & = & [\, \underline{0}, g \,] \\
g \cdot [\, \underline{0}, suc \,] & = & [\, \underline{1}, + \cdot \langle f, g\rangle \,]
\end{array}
$$

The mutual dependence can be made more explicit by forcing

$$
\begin{array}{rcl}
f \cdot [\, \underline{0}, suc \,] & = & [\, \underline{0}, \pi_2 \cdot \langle f, g\rangle \,] \\
g \cdot [\, \underline{0}, suc \,] & = & [\, \underline{1}, + \cdot \langle f, g\rangle \,]
\end{array}
$$

The underlying inductive type is

$$
\mathbb{N}_0 \quad \cong \quad \underbrace{1 + \mathbb{N}_0}_{\mathsf{F}\,\mathbb{N}_0} \tag{A.6}
$$

which is such that $\mathsf{F}\,f = id + f$. So we can write

$$
\begin{array}{rcl}
f \cdot in & = & [\, \underline{0}, \pi_2 \,] \cdot \mathsf{F}\,\langle f, g\rangle \\
g \cdot in & = & [\, \underline{1}, + \,] \cdot \mathsf{F}\,\langle f, g\rangle
\end{array}
$$

So we identify $h = [\,\underline{0}, \pi_2\,]$ and $k = [\,\underline{1}, +\,]$ therefore obtaining

$$
\begin{array}{ll}
\langle f, g \rangle & \\
= & \{ \text{Fokkinga law} \} \\
& \llparenthesis \langle [\,\underline{0}, \pi_2\,], [\,\underline{1}, +\,] \rangle \rrparenthesis \\
= & \{ \text{exchange law} \} \\
& \llparenthesis [\,\langle \underline{0}, \underline{1} \rangle, \langle \pi_2, + \rangle\,] \rrparenthesis
\end{array}
$$

which is easily converted into VDM-SL as follows:

```
fg: nat -> nat
fg(n) == if n = 0 then mk_(0,1)
         else let p=fg(n - 1)
              in mk_(p.#2,p.#1 + p.#2);
```

A.3.1 Example

Checking a list-invariant which ensures that a (non-empty) list is ordered:

$$
\begin{array}{l}
ordered : \; A^{+} \longrightarrow 2 \\
ordered\,[a] = TRUE \\
ordered\,(cons(a, l)) = a > (Max\, l) \wedge (ordered\, l)
\end{array}
$$

Assuming *singl* $a = [a]$ we can depict $ordered$ as follows:

$$
\begin{tikzcd}
A^{+} \arrow[d, "ordered"'] & A + A \times A^{+} \arrow[l, "{[\,singl, cons\,]}"'] \arrow[d, "id + id \times \langle Max, ordered \rangle"] \\
2 & A + A \times (A \times 2) \arrow[l, "{[\,\underline{TRUE}, \alpha\,]}"]
\end{tikzcd}
$$

where

$$\alpha(a, (m, b)) \stackrel{\mathrm{def}}{=} a > m \wedge b$$

and where

$$Max = \llparenthesis [\,id, max\,] \rrparenthesis$$

cf.

$$
\begin{tikzcd}
A^{+} \arrow[d, "Max"'] & A + A \times A^{+} \arrow[l, "{[\,singl, cons\,]}"'] \arrow[d, "id + id \times Max"] \\
A & A + A \times A \arrow[l, "{[\,id, max\,]}"]
\end{tikzcd}
$$

It is easy to check that the equation implicit in this diagram is the same as the one implicit in

$$\begin{tikzcd} A^+ \arrow[d, "Max"'] & A + A \times A^+ \arrow[l, "{[\, singl, cons \,]}"'] \arrow[d, "id + id \times \langle Max, g\rangle"] \\ A & A + A \times (A \times B) \arrow[l, "{[\, id, max \cdot (id \times \pi_1) \,]}"] \end{tikzcd}$$

for any $A^+ \xrightarrow{g} B$. For $B = 2$ and $g = ordered$ we are in position to apply Fokkinga's law and to obtain:

$$\begin{array}{rcl} \langle Max, ordered\rangle & = & (\!|\langle [\, id, max \cdot (id \times \pi_1) \,], [\, \underline{TRUE}, \alpha \,]\rangle|\!) \\ & = & \{\ \textit{exchange law } (1.47)\} \\ & & (\!|[\, \langle id, \underline{TRUE}\rangle, \langle max \cdot (id \times \pi_1), \alpha\rangle \,]|\!) \end{array}$$

Of course, $ordered = \pi_2 \cdot \langle Max, ordered\rangle$. Calling aux to the above synthesized catamorphism, we end up with the following realization of $ordered$:

$$\begin{array}{rcll} ordered\, l & = & let & (a, b) = aux\, l \\ & & in & b \end{array}$$

where

$$\begin{array}{l} aux : \ A^+ \longrightarrow A \times 2 \\ ordered\,[a] = (a, TRUE) \\ ordered\,(cons(a, l)) = \begin{array}[t]{ll} let & (m, b) = aux\, l \\ in & (max(a, m), (a > m \wedge b)) \end{array} \end{array}$$

A.3.2 "Banana-split": a corollary of the mutual-recursion law

Let $h = i \cdot \mathsf{F}\,\pi_1$ and $k = j \cdot \mathsf{F}\,\pi_2$ in (A.5). Then

$$\begin{array}{ll} & f \cdot in = (i \cdot \mathsf{F}\,\pi_1) \cdot \mathsf{F}\,\langle f, g\rangle \\ \equiv & \quad \{\ \text{composition is associative and } \mathsf{F} \text{ is a functor}\} \\ & f \cdot in = i \cdot \mathsf{F}\,(\pi_1 \cdot \langle f, g\rangle) \\ \equiv & \quad \{\ \text{by } \times\text{-cancellation } (1.20)\} \\ & f \cdot in = i \cdot \mathsf{F}\, f \\ \equiv & \quad \{\ \text{by cata-cancellation}\} \\ & f = (\!|i|\!) \end{array}$$

Similarly, from $k = j \cdot \mathsf{F}\,\pi_2$ we get

$$g = (\!|j|\!)$$

Then, from (A.5), we get

$$\langle \llparenthesis i \rrparenthesis, \llparenthesis j \rrparenthesis \rangle = \llparenthesis \langle i \cdot \mathsf{F}\,\pi_1, j \cdot \mathsf{F}\,\pi_2 \rangle \rrparenthesis$$

that is

$$\langle \llparenthesis i \rrparenthesis, \llparenthesis j \rrparenthesis \rangle = \llparenthesis (i \times j) \cdot \langle \mathsf{F}\,\pi_1, \mathsf{F}\,\pi_2 \rangle \rrparenthesis \tag{A.7}$$

by (reverse) $\times$-absorption (1.25).

This law provides us with a very useful tool for "parallel loop" inter-combination: "loops" $\llparenthesis i \rrparenthesis$ and $\llparenthesis j \rrparenthesis$ are fused together into a single "loop" $\llparenthesis (i \times j) \cdot \langle \mathsf{F}\,\pi_1, \mathsf{F}\,\pi_2 \rangle \rrparenthesis$. The need for this kind of calculation arises very often. Consider, for instance, the function which computes the average of a non-empty list of natural numbers:

$$average \stackrel{\mathrm{def}}{=} (/) \cdot \langle sum, length \rangle$$

Both sum and $length$ are $\mathbb{N}^+$ catamorphisms:

$$\begin{aligned} suma &= \llparenthesis [\, id, + \,] \rrparenthesis \\ length &= \llparenthesis [\, \underline{1}, suc \cdot \pi_2 \,] \rrparenthesis \end{aligned}$$

Function $average$ will do two independent traversals of the argument list before division $(/)$ takes place. Banana-split fuses such two traversals into a single one, thus leading to a function which: (a) runs twice as fast (b) can be converted into a *while loop* by introduction of accummulation parameters (such as seen above).

Exercise 1.3 Apply the banana-split law to the following definition of the $unzip$ function:

$$unzip \stackrel{\mathrm{def}}{=} \langle \pi_1{}^\star, \pi_2{}^\star \rangle$$

Extend $unzip$ to binary trees and repeat the exercise.
□

A.4 2001.10.25 — Paramorphisms

Consider the standard definition of the factorial function (in VDM-SL notation):

```
fac : nat -> nat
fac(n) == if n = 0 then 1 else fac(n-1) * n
```

The pattern of recursion of this function — usually known as *primitive recursion* — is somewhat more elaborate than that of a catamorphism over $\mathbb{N}$.

Note than it can be captured by the following diagram:

$$\begin{tikzcd} \mathbb{N} \arrow[d, "fac"'] & 1 + \mathbb{N} \arrow[l, "{[\, \underline{0}, suc \,]}"'] \arrow[d, "id + \langle fac, id \rangle"] \\ \mathbb{N} & 1 + (\mathbb{N} \times \mathbb{N}) \arrow[l, "{[\, \underline{1}, mul \cdot (id \times suc) \,]}"] \end{tikzcd}$$

Function fac is a particular instance of a so-called *paramorphism*. In general, a paramorphism of some f relative to functor F, is the unique morphism $\langle\!|f|\!\rangle$ which is such that

$$\begin{tikzcd} \mathsf{T} \arrow[d, "\langle\!|f|\!\rangle"'] & \mathsf{F}\,\mathsf{T} \arrow[l, "in"'] \arrow[d, "\mathsf{F}\,\langle \langle\!|f|\!\rangle, id\rangle"] \\ C & \mathsf{F}\,(C \times \mathsf{T}) \arrow[l, "f"] \end{tikzcd}$$

that is, we have the following universal property:

$$h = \langle\!|f|\!\rangle \quad \equiv \quad h \cdot in = f \cdot \mathsf{F}\,\langle h, id\rangle$$

A.4.1 Examples of paramorphisms

From above we can express the factorial function as a paramorphism:

$$fac = \langle\!|[\,\underline{1}, mul \cdot (id \times suc)\,]|\!\rangle$$

A less straightforward example is that of a function nw — *cf.* `wc -w` in LINUX— counting the number of words in text (`seq of char`):

```
nw : seq of char -> nat
nw(s) == if s = [] then 0
         else if not sep(hd s) and sepahead(tl s)
              then nw(tl s) + 1 else nw(tl s) ;

sepahead: seq of char -> bool
sepahead(s) == (s = []) or sep(hd s) ;

sep : char -> bool
sep(c) == c  = ' ' or c  = '\n' or c  = '\t' ;
```

This is list-paramorphism

$$\begin{tikzcd} \mathtt{char}^\star \arrow[d, "nw"'] & 1 + \mathtt{char} \times \mathtt{char}^\star \arrow[l, "{[\,\underline{[\,]}, cons\,]}"'] \arrow[d, "id+id\times\langle nw, id\rangle"] \\ \mathbb{N}_0 & 1 + \mathtt{char} \times (\mathbb{N}_0 \times \mathtt{char}^\star) \arrow[l, "{[\,\underline{0}, h\,]}"] \end{tikzcd}$$

where

```
h : char * (nat * seq of char) -> nat
h(c,mk_(i,s)) == if not sep(c) and sepahead(s) then i + 1 else i ;
```

A.4.2 Properties of paramorphisms

1. Clearly, every catamorphism can be expressed by a paramorphism:

$$(\!| f |\!) = \langle\!| f \cdot \mathsf{F}\,\pi_1 |\!\rangle \tag{A.8}$$

   Proof:

$$\begin{array}{cl} & h = \langle\!| f \cdot \mathsf{F}\,\pi_1 |\!\rangle \equiv h \cdot in = f \cdot \mathsf{F}\,\pi_1 \cdot \mathsf{F}\,\langle h, id\rangle \\ = & \quad \{\text{ functor versus composition (2.45)}, \times\text{-cancellation}\} \\ & h = \langle\!| f \cdot \mathsf{F}\,\pi_1 |\!\rangle \equiv h \cdot in = f \cdot \mathsf{F}\,h \\ = & \quad \{\text{ cata-universal}\} \\ & (\!| f |\!) = \langle\!| f \cdot \mathsf{F}\,\pi_1 |\!\rangle \end{array}$$

2. Conversely, every paramorphism can be expressed (indirectly) in terms of a catamorphism:

$$\langle\!| h |\!\rangle = \pi_1 \cdot (\!| \langle h, in \cdot \mathsf{F}\,\pi_2\rangle |\!) \tag{A.9}$$

   Proof: let g be id in the mutual-recursion law, leading to $f = \langle\!| h |\!\rangle$. Then the equation for $g = id$ is

$$id \cdot in = k \cdot \mathsf{F}\,\langle f, id\rangle$$

   and this is satisfied for $k = in \cdot \mathsf{F}\,\pi_2$.
   So $\langle f, g\rangle = (\!| \langle h, in \cdot \mathsf{F}\,\pi_2\rangle |\!)$ and $f = \langle\!| h |\!\rangle = \pi_1 \cdot (\!| \langle h, in \cdot \mathsf{F}\,\pi_2\rangle |\!)$.

3. PARA-REFLECTION:

$$id = \langle\!| in \cdot \mathsf{F}\,\pi_1 |\!\rangle \tag{A.10}$$

   By cata-reflection (2.58) this can be regarded as an instance of (A.8) above.

4. PARA-FUSION:

$$h \cdot \langle\!| f |\!\rangle = \langle\!| g |\!\rangle \quad \Leftarrow \quad h \cdot f = g \cdot \mathsf{F}(h \times id) \tag{A.11}$$

Example of aplication

By (A.9) the factorial function can be expressed by the projection of a catamorphism:

$$\begin{array}{cl} & fac = \pi_1 \cdot (\!| \langle [\,\underline{1}, mul \cdot (id \times suc)\,], in \cdot (id + \pi_2)\rangle |\!) \\ \equiv & \quad \{\text{ +-absorption (1.41)}\} \\ & fac = \pi_1 \cdot (\!| \langle [\,\underline{1}, mul \cdot (id \times suc)\,], [\,\underline{0}, suc \cdot \pi_2\,]\rangle |\!) \\ = & \quad \{\textit{ exchange law } (1.47)\} \\ & fac = \pi_1 \cdot (\!| [\,\langle \underline{1}, \underline{0}\rangle, \langle mul \cdot (id \times suc), suc \cdot \pi_2\rangle\,] |\!) \end{array}$$

This will lead to the following VDM-SL:

```
fac : nat -> nat
fac(n)==facaux(n).#1;

facaux: nat -> nat*nat
facaux(n) == if n=0 then mk_(1,0)
             else let p = facaux(n-1),
                      a = p.#1,
                      b = p.#2
                  in mk_(a * (b + 1), b + 1);
```

A.5 2001.11.29 — Elements of the Fixpoint Calculus

A.5.1 Basic definitions

Definition 1 (Poset) *A poset $(A, \leq_A)$ is a set A equipped with a partial ordering $\leq_A$, that is, a relation $\leq_A \subseteq A \times A$ which is reflexive, transitive and antisymmetric.*
□

Definition 2 (Pre/post-fixpoints) *Let $A \xleftarrow{f} A$ be a (endo) function on poset $(A, \leq_A)$. Then*

- *every $a \in A$ such that*

$$a \quad \leq_A f\, a \tag{A.12}$$

  is said to be a post-fixpoint *of f.*

- *every $a \in A$ such that*

$$a \quad \geq_A f\, a \tag{A.13}$$

  is said to be a pre-fixpoint *of f.*

- *every $a \in A$ which is both a pre-fixpoint and a post-fixpoint of f is said to be a* fixpoint *of f and is such that*

$$a \quad = \quad f\, a \tag{A.14}$$

  holds.

□

Examples:

- Given endofunction

$$\begin{array}{rcl} f : [0, 10] & \to & [0, 10] \\ x & \rightsquigarrow & 10 - x \end{array}$$

  one very easily checks that 5 is a fixpoint of f, since $f\,5 = 10 - 5 = 5$.

- Let $R \subseteq P \times P$ be a relation on nonempty P in

$$x = R \cup R \circ x \tag{A.15}$$

Define

$$f\,x = R \cup R \circ x \tag{A.16}$$

on poset $(\mathcal{P}(P \times P), \subseteq)$. Then

  – $P \times P$ is an example of a pre-fixpoint of f ($P \times P$ is the largest relation in the poset).

  – $\emptyset$ and R are examples of post-fixpoints of f. In fact, $\emptyset \subseteq R$ and $R \subseteq R \cup R^2$.

Clearly, every fixpoint $a = f\,a$ can be regarded as a "solution" to equation

$$x = f\,x \tag{A.17}$$

But one can also regard this equation as a "recursive" definition of its fixpoints. For instance, recall equation (2.3)

$$x = 1 + \frac{x}{2}$$

The fact that 2 is a fixpoint of this equation can be rephrased to: "$x = 1 + \frac{x}{2}$" *is a recursive definition of number* 2.

However, the following equation

$$x = \frac{x^2 + 3}{4}$$

admits two solutions (fixpoints) 1 e 3. What are we "recursively defining" here? The 1 or the 3? Furthermore, equation

$$x = x$$

defines any object! By contrast, some equations don't have any solution at all. Think *e.g.* of

$$x = x + 1$$

in $\mathbb{N}$. So, in this case, our recursive equation defines. . . nothing!

A.5.2 Computing fixpoints

Definition 3 (Monotone functions) *A function* $B \xleftarrow{f} A$ *from poset* $(A, \leq_A)$ *to poset* $(B, \leq_B)$ *is said to be* monotone *iff*

$$\forall a, a' \in A : a \leq_A a' \Rightarrow (f\,a) \leq_B (f\,a')$$

holds.

□

Definition 4 (Ordering on functions) *Given two functions* $B \xleftarrow{f} A$ *and* $B \xleftarrow{g} A$ *from poset* $(A, \leq_A)$ *to poset* $(B, \leq_B)$ *define*

$$f \leq g \quad \stackrel{\mathrm{def}}{=} \quad \forall a \in A : (f\, a) \leq_B (g\, a) \tag{A.18}$$

□

Theorem 1 (Lattice Fixpoints) *[Tarski 1955]*
Let

- $A \xleftarrow{f} A$ *be a monotone function on a* complete lattice $(A; \leq)$*;*
- P *be the set of all fixpoints of* f, i.e.

$$P = \{a \in A \mid \; a = f\, a\}$$

Then

- P *is non-empty and* $(P; \leq)$ *is a complete (sub)lattice.*
- *In particular, the least of all fixpoints* $(\bigwedge P)$ *and the greatest one* $(\bigvee P)$ *are as follows:*

$$\bigwedge P \quad = \quad \bigwedge \{x \mid \; x \geq f\, x\} \tag{A.19}$$

$$\bigvee P \quad = \quad \bigvee \{x \mid \; x \leq f\, x\} \tag{A.20}$$

We define:

$$\mu f \quad \stackrel{\mathrm{def}}{=} \quad \bigwedge P \tag{A.21}$$

$$\nu f \quad \stackrel{\mathrm{def}}{=} \quad \bigvee P \tag{A.22}$$

□

In the sequel we shall be focussing on *least* fixpoints.

A.6 2001.12.06 — Laws of the Fixpoint Calculus

Computation rule:

$$\mu f \quad = \quad f\, \mu f \tag{A.23}$$

Rolling rule:

$$\mu(g \cdot f) \quad = \quad g(\mu(f \cdot g)) \tag{A.24}$$

Square rule:

$$\mu f = \mu(f^2) \tag{A.25}$$

Monotonicity:

$$\mu f \leq \mu g \quad \Leftarrow \quad f \leq g \tag{A.26}$$

Induction rule:

$$\mu f \leq x \quad \Leftarrow \quad f\,x \leq x \tag{A.27}$$

A.6.1 Illustration

Let $f\,x = 1 + \frac{x}{2}$. Successive application of the computation rule (A.23) leads to:

$$\begin{array}{rcl} \mu f & = & 1 + \dfrac{\mu f}{2} \\ & = & 1 + \dfrac{(1 + \frac{\mu f}{2})}{2} = 1 + \dfrac{1}{2} + \dfrac{\mu f}{4} \\ & \vdots & \\ & = & \displaystyle\sum_{i=1}^{n} \frac{1}{2^n} + \frac{\mu f}{2^{n+1}} \end{array}$$

In the limit ($n \to \infty$), we get $\frac{\mu f}{2^{n+1}} = 0$ and therefore $\mu f = \sum_{i=1}^{\infty} \frac{1}{2^n} = 2$.

The rolling rule (A.24) can be applied decomposing $f = g \cdot h$ for $h\,x = \frac{x}{2}$ and $g\,x = 1 + x$. Then

$$\begin{array}{rcl} \mu f & = & \mu(g \cdot h) = g(\mu(h \cdot g)) \\ & = & 1 + \mu x.\dfrac{1+x}{2} \end{array}$$

whereby $x = \frac{1+x}{2}$ has solution 1.

The aother rules enable us to reason inequationally. For instante, fact $1+\frac{x}{2} \leq 2+\frac{x}{2}$, for all x, and monotonicity (A.26) enables us to say that $\mu x.(1 + \frac{x}{2}) = 2$ is smaller than $\mu x.(2 + \frac{x}{2}) = 4$.

Similar intuition can be gathered from (A.16), providing evidence that $\mu f = R^+$ (transitive closure of R).

For instance (rolling rule), we can decompose f into $g \cdot h$ where $h\,x = R \cdot x$ and $g\,x = R \cup x$. Then

$$\begin{array}{rcl} \mu f & = & \mu(g \cdot h) \\ & = & \qquad \{ \;\cdots\cdots\cdots\cdots \} \\ & & g(\mu(h \cdot g)) \end{array}$$

$$
\begin{aligned}
= & \quad \{ \ \cdots\cdots\cdots\cdots \} \\
& R \cup (\mu x.(R \cdot (R \cup x)) \\
= & \quad \{ \ \cdots\cdots\cdots\cdots \} \\
& R \cup \mu x.(R^2 \cup R \cdot x)
\end{aligned}
$$

Further application of this rule will “factor out” R^2, R^3, *etc.*, leaving a “smaller and smaller” fixpoint to be calculated. In the limit, one gets $\mu f = \bigcup_{j=1}^{\infty} R^j = R^+$.

A.6.2 Inductive datatypes “are” fixpoints

Recall

$$
X \quad = \quad \underbrace{1 + A \times X}_{\mathsf{F}\,X} \tag{A.28}
$$

- The “=” symbol in equation (A.28) should be understood as “$\cong$”
- F should be understood as a *functor*
- So any solution X_0 to the equation should carry along an algebra in and its inverse out thus providing evidence of the required *isomorphism*:

$$
\begin{tikzcd}
X_0 \arrow[rr, bend left, "out"] \arrow[rr, phantom, "\cong"] & & 1 + A \times X_0 \arrow[ll, bend left, "in"]
\end{tikzcd}
$$

  For instance,

$$
\begin{tikzcd}
A^\star \arrow[rr, bend left, "out"] \arrow[rr, phantom, "\cong"] & & 1 + A \times A^\star \arrow[ll, bend left, "{in=[\ \underline{[\,]}, cons\ ]}"]
\end{tikzcd}
$$

  where out is the obvious inverse of in.

- The $\leq$-ordering corresponds to right-invertibility:

$$
\begin{tikzcd}
A \arrow[r, bend left, "r"] \arrow[r, phantom, "\leq"] & B \arrow[l, bend left, "f"]
\end{tikzcd}
\quad \text{that is} \quad \left\{ \ f \cdot r = id_A \right. \tag{A.29}
$$

In general: For F a polynomial functor, equation $X \cong \mathsf{F}\,X$

- admits a standard solution — its **least fixpoint** solution

$$\begin{tikzcd} \mu\mathsf{F} \arrow[r, bend left, "out"] \arrow[r, phantom, "\cong"] & \mathsf{F}\,\mu\mathsf{F} \arrow[l, bend left, "in"] \end{tikzcd} \tag{A.30}$$

  Example:

$$\mu X.1 + A \times X \quad = \quad A^\star$$

  — where $\mu X.\mathsf{F}\,X$ abbreviates $\mu(\lambda X.\mathsf{F}\,X)$.

- $\mu\mathsf{F}$ is *initial* among all other F-structures — that is to say, for a given $(A,\ A \xleftarrow{\alpha} \mathsf{F}\,A\ )$, arrow k in

$$\begin{tikzcd} \mu\mathsf{F} \arrow[d, "k"'] & \mathsf{F}\,\mu\mathsf{F} \arrow[l, "in"'] \arrow[d, "\mathsf{F}\,k"] \\ A & \mathsf{F}\,A \arrow[l, "\alpha"] \end{tikzcd}$$

  is unique, recall **universal property**:

$$k = (\!|\alpha|\!) \quad \Leftrightarrow \quad k \cdot in = \alpha \cdot \mathsf{F}\,k$$

A.6.3 Application of the Fixpoint Calculus to datatypes

Computation rule:

$$\begin{tikzcd} \mu\mathsf{F} \arrow[r, bend left, "out"] \arrow[r, phantom, "\cong"] & \mathsf{F}\,(\mu\mathsf{F}) \arrow[l, bend left, "in"] \end{tikzcd} \tag{A.31}$$

cf. (A.30).

Rolling rule:

$$\begin{tikzcd} \mu(\mathsf{G}\cdot\mathsf{F}) \arrow[r, bend left, "(\!|\mathsf{G}\,in_{\mu(\mathsf{F}\cdot\mathsf{G})}|\!)"] \arrow[r, phantom, "\cong"] & \mathsf{G}(\mu(\mathsf{F}\cdot\mathsf{G})) \arrow[l, bend left, "in_{\mathsf{G}\cdot\mathsf{F}}\cdot\mathsf{G}(\!|\mathsf{F}\,in_{\mathsf{G}\cdot\mathsf{F}}|\!)"] \end{tikzcd} \tag{A.32}$$

cf.

$$\begin{tikzcd}
\mu(\mathsf{G}\cdot\mathsf{F}) \arrow[d, "(|g|)"'] & (\mathsf{G}\cdot\mathsf{F})(\mu(\mathsf{G}\cdot\mathsf{F})) \arrow[l, "in_{\mu(\mathsf{G}\cdot\mathsf{F})}"'] \arrow[d, "(\mathsf{G}\cdot\mathsf{F})(|g|)"] \\
\mathsf{G}(\mu(\mathsf{F}\cdot\mathsf{G})) & (\mathsf{G}\cdot\mathsf{F})(\mathsf{G}(\mu(\mathsf{F}\cdot\mathsf{G}))) \arrow[l, "g=\mathsf{G}\,in_{\mu(\mathsf{F}\cdot\mathsf{G})}"']
\end{tikzcd}$$

Example: Let

$$\begin{array}{rcl} \mathsf{F}\,X & = & 1+X \\ \mathsf{G}\,X & = & A\times X \end{array}$$

Then

$$\begin{array}{rcl} (\mathsf{F}\cdot\mathsf{G})X & = & 1+A\times X \\ (\mathsf{G}\cdot\mathsf{F})X & = & A\times(1+X)\cong A+A\times X \end{array}$$

and so

$$\begin{array}{rcl} \mu(\mathsf{F}\cdot\mathsf{G}) & = & A^\star \\ \mu(\mathsf{G}\cdot\mathsf{F}) & = & A^+ \end{array}$$

The rolling rule will state the obvious fact that

$$A^+ \quad\cong\quad A\times A^\star$$

holds, that is

$$\begin{tikzcd}
A^+ \arrow[d, "f"'] & A\times(1+A^+) \arrow[d, "id\times(id+f)"] & A+A\times A^+ \arrow[l, "k"] \arrow[ll, bend right, "in_{A^+}"'] \arrow[d, "id+id\times f"] \\
A\times A^\star & A\times(1+A\times A^\star) \arrow[l, "id\times in_{A^\star}"'] & A+A\times(A\times A^\star) \arrow[l, "k"']
\end{tikzcd} \tag{A.33}$$

for $k=\langle [\,id,\pi_1\,],(!+\pi_2)\rangle$

Exercise 1.4 Concerning (A.33), show that $f=\langle f_1,f_2\rangle$ where f_1 is the "head" function and f_2 is the "tail" function on A^+.
□

Monotonicity:

$$\begin{tikzcd}
\mu\mathsf{F} \arrow[r, bend left, "(|in_{\mathsf{G}}\cdot r|)_{\mathsf{F}}"] \arrow[r, phantom, "\leq"] & \mu\mathsf{G} \arrow[l, bend left, "(|in_{\mathsf{F}}\cdot f|)_{\mathsf{G}}"]
\end{tikzcd}
\quad\Leftarrow\quad
\begin{tikzcd}
\mathsf{F}\,X \arrow[r, bend left, "r"] \arrow[r, phantom, "\leq"] & \mathsf{G}\,X \arrow[l, bend left, "f"]
\end{tikzcd}$$

cf. diagram

$$\begin{tikzcd}
\mu\mathsf{G} \arrow[d, "(\!|in_{\mathsf{F}}\cdot f|\!)_{\mathsf{G}}"'] & \mathsf{F}\,(\mu\mathsf{G}) \arrow[d, "\mathsf{F}\,(\!|in_{\mathsf{F}}\cdot f|\!)_{\mathsf{G}}"] & \mathsf{G}\,(\mu\mathsf{G}) \arrow[l, "f"'] \arrow[ll, bend right, "in_{\mathsf{G}}"'] \arrow[d, "\mathsf{G}\,(\!|in_{\mathsf{F}}\cdot f|\!)_{\mathsf{G}}"] \\
\mu\mathsf{F} & \mathsf{F}\,(\mu\mathsf{F}) \arrow[l, "in_{\mathsf{F}}"] & \mathsf{G}\,(\mu\mathsf{F}) \arrow[l, "f"]
\end{tikzcd}$$

Let us see an example of application, whereby possibly empty sequences are represented by non-empty ones — $\mu\mathsf{F} = A^\star$ and $\mu\mathsf{G} = A^+$, for $1 \leq A$:

$$\begin{tikzcd}
A^\star \arrow[r, phantom, "\cong"] & 1 + A \times A^\star \arrow[l, bend left, "{[\,\underline{[\,]},cons\,]}"]
\end{tikzcd}$$

$$\begin{tikzcd}
A^+ \arrow[r, phantom, "\cong"] & A + A \times A^+ \arrow[l, bend left, "{[\,sing,cons\,]}"]
\end{tikzcd}$$

where $sing\,a = [a]$.
Of course

$$\begin{tikzcd}
1 + A \times X \arrow[r, bend left, "\underline{a_0}+id"] \arrow[r, phantom, "\leq"] & A + A \times X \arrow[l, bend left, "!+id"]
\end{tikzcd}$$

holds for some $a_0 \in A$ since

$$\begin{tikzcd}
1 \arrow[r, bend left, "\underline{a_0}"] \arrow[r, phantom, "\leq"] & A \arrow[l, bend left, "!"]
\end{tikzcd}$$

by hypothesis. According to (2.78) we infer not only that non-empty lists implement empty lists

$$\begin{tikzcd}
A^\star \arrow[r, bend left, "embed"] \arrow[r, phantom, "\leq"] & A^+ \arrow[l, bend left, "blast"]
\end{tikzcd}$$

(obvious!) but how they do it:

$$\begin{aligned}
blast &= (\!|\,[\,\underline{[\,]}, cons\,] \cdot (! + id)\,|\!) \\
embed &= (\!|\,[\,sing, cons\,] \cdot (\underline{a_0} + id)\,|\!)
\end{aligned}$$

(less obvious). Let us calculate further:

$$
\begin{array}{cl}
 & blast \cdot [\, sing, cons \,] \\
= & \quad \{ \text{ cata-cancellation} \} \\
 & [\, \underline{[\,]}, cons \,] \cdot (! + id) \cdot (id + id \times blast) \\
= & \quad \{ \text{ $+$-bifunctor and trivia} \} \\
 & [\, \underline{[\,]}, cons \,] \cdot (! + id \times blast) \\
= & \quad \{ \text{ $+$-absorption and trivia} \} \\
 & [\, \underline{[\,]}, cons \cdot (id \times blast) \,]
\end{array}
$$

that is

$$
\begin{array}{rcl}
blast\,[a] & = & [\,] \\
blast\,(cons(a,l)) & = & cons(a, blast\,l)
\end{array}
$$

($blast$="all <u>b</u>ut <u>last</u> ")

Similarly:

$$
\begin{array}{rcl}
embed\,[\,] & = & [a_0] \\
embed\,(cons(a,l)) & = & cons(a, embed\,l)
\end{array}
$$

Constructive proof

We want to find a right-inverse for $(\!| in_{\mu\mathsf{G}} \cdot g |\!)$ in (2.78) which, of course, will be expressible as a catamorphism $(\!| \alpha |\!)$, *cf.* diagram

$$
\begin{tikzcd}
\mu\mathsf{F} \arrow[d, "(\!|\alpha|\!)_{\mathsf{F}}"'] & \mathsf{F}\,\mu\mathsf{F} \arrow[l, "in_{\mu\mathsf{G}}"'] \arrow[d, "\mathsf{F}\,(\!|\alpha|\!)_{\mathsf{F}}"] \\
\mu\mathsf{G} \arrow[d, "(\!|in_{\mu\mathsf{G}} \cdot g|\!)_{\mathsf{G}}"'] & \mathsf{F}\,\mu\mathsf{G} \arrow[l, "\alpha"] \arrow[d, "\mathsf{F}\,(\!|in_{\mu\mathsf{G}} \cdot g|\!)_{\mathsf{G}}"] \\
\mu\mathsf{F} & \mathsf{F}\,\mu\mathsf{F} \arrow[l, "in_{\mu\mathsf{G}} \cdot g"]
\end{tikzcd}
$$

So α is the unknown. Calculation of α:

$$
\begin{array}{cl}
 & (\!| in_{\mu\mathsf{F}} \cdot g |\!)_{\mathsf{G}} \cdot (\!| \alpha |\!) = id \\
\Leftrightarrow & \quad \{ \text{ by F-cata-reflexion} \} \\
 & (\!| in_{\mu\mathsf{F}} \cdot g |\!)_{\mathsf{G}} \cdot (\!| \alpha |\!)_{\mathsf{F}} = (\!| in_{\mu\mathsf{F}} |\!)_{\mathsf{F}} \\
\Leftarrow & \quad \{ \text{ by F-cata-fusion} \}
\end{array}
$$

$$
\begin{array}{ll}
& (\!|in_{\mu\mathsf{F}} \cdot g|\!)_{\mathsf{G}} \cdot \alpha = in_{\mu\mathsf{F}} \cdot \mathsf{F}\,(\!|in_{\mu\mathsf{F}} \cdot g|\!)_{\mathsf{G}} \\
\Leftrightarrow & \qquad \{\text{ decompose } \alpha = in_{\mu\mathsf{G}} \cdot \alpha'\} \\
& (\!|in_{\mu\mathsf{F}} \cdot g|\!)_{\mathsf{G}} \cdot in_{\mu\mathsf{G}} \cdot \alpha' = in_{\mu\mathsf{F}} \cdot \mathsf{F}\,(\!|in_{\mu\mathsf{F}} \cdot g|\!)_{\mathsf{G}} \\
\Leftrightarrow & \qquad \{\text{ by } \mathsf{G}\text{-cata-cancellation}\} \\
& in_{\mu\mathsf{F}} \cdot g \cdot \mathsf{G}\,(\!|in_{\mu\mathsf{F}} \cdot g|\!)_{\mathsf{G}} \cdot \alpha' = in_{\mu\mathsf{F}} \cdot \mathsf{F}\,(\!|in_{\mu\mathsf{F}} \cdot g|\!)_{\mathsf{G}} \\
\Leftrightarrow & \qquad \{\ g \text{ is natural (A.34)}\} \\
& in_{\mu\mathsf{F}} \cdot \mathsf{F}\,(\!|in_{\mu\mathsf{F}} \cdot g|\!)_{\mathsf{G}} \cdot g \cdot \alpha' = in_{\mu\mathsf{F}} \cdot \mathsf{F}\,(\!|in_{\mu\mathsf{F}} \cdot g|\!)_{\mathsf{G}} \\
\Leftarrow & \qquad \{\ g \cdot s = id\} \\
& \alpha' = s
\end{array}
$$

So

$$\alpha \quad = \quad in_{\mathsf{G}} \cdot s$$

NB: both g and s are natural, *e.g.*

$$(\mathsf{F}\,f) \cdot g = g \cdot (\mathsf{G}\,f) \qquad \begin{tikzcd} A \arrow[d, "f"'] \\ B \end{tikzcd} \qquad \begin{tikzcd} \mathsf{G}\,A \arrow[r, "g_A"] \arrow[d, "\mathsf{G}\,f"'] & \mathsf{F}\,A \arrow[d, "\mathsf{F}\,f"] \\ \mathsf{G}\,B \arrow[r, "g_B"'] & \mathsf{F}\,B \end{tikzcd} \tag{A.34}$$

holds.

Square rule:

$$\begin{tikzcd} \mu\mathsf{F} \arrow[r, bend left, "g"] \arrow[r, phantom, "\cong"] & \mu(\mathsf{F}^2) \arrow[l, bend left, "(\!|in \cdot (\mathsf{F} in)|\!)"] \end{tikzcd} \tag{A.35}$$