Chapter 3

Quasi-inductive datatypes

3.1 Introducing Non-inductive Datatypes

Recall that an inductive datatype is defined up to isomomphism

$$\begin{tikzcd} \mathsf{T} \arrow[rr, bend left, "out"] \arrow[rr, phantom, "\cong"] & & \mathsf{F}\,\mathsf{T} \arrow[ll, bend left, "in"] \end{tikzcd}$$

and that from its definition two isomorphisms emerge — algebra in and coalgebra out — which are each other's inverses:

$$\begin{array}{rcl} in \cdot out & = & id_{\mathsf{T}} \\ out \cdot in & = & id_{(\mathsf{F}\,\mathsf{T})} \end{array}$$

Algebra in provides datatype constructors upon which one may build inductive definitions, algorithms (catamorphisms), etc. It also sets up the basis for inductive proofs about the datatype. Conversely, coalgebra out provides a "grammar" for observing the datatype in a "recursive descent" fashion.

Because in and out are each-other inverses, structuring algoritms around in or out is simply a matter of taste: the former leads to an "axiomatic" (structural, inductive) style, the latter to a more algorithmic (recursive, interpretative) style.

Not every datatype is inductive in this way. Think for instance of the *powerset* datatype $\mathcal{P}A$ containing all subsets of a finite set A:

$$\mathcal{P}A \quad = \quad \{s \mid \ s \subseteq A\}$$

We know that the empty set $\emptyset$ is a subset of A — and therefore an inhabitant of $\mathcal{P}A$ — and that, given a subset s of A and a particular $a \in A$, then $\{a\} \cup s$ is another (possibly larger) subset of A. So $\mathcal{P}A \xleftarrow{ins} 1 + A \times \mathcal{P}A$, where $ins = [\,\underline{\emptyset}, \lambda(a,s).\{a\} \cup s\,]$, is an algebra for synthesizing $\mathcal{P}A$ data values. We will write

$$ins = [\,\underline{\emptyset}, puts\,] \tag{3.1}$$

by introducing

$$puts \stackrel{\text{def}}{=} \cup \cdot (sings \times id) \tag{3.2}$$

where $sings\,x = \{x\}$ ("singleton set"). Conversely, if $s \subseteq A$ is nonempty, it can always be decomposed into pair $(a, s - \{a\})$ for some $a \in A$. So we can think of $\mathcal{P}A \xrightarrow{outs} 1 + A \times \mathcal{P}A$ where

$$outs = (! + gets) \cdot (=_\emptyset)? \tag{3.3}$$

and where, for nonempty s,

$$gets\, s \stackrel{\text{def}}{=} \begin{array}{ll} let & a \in s \\ in & (a, s - \{a\}) \end{array} \tag{3.4}$$

(The attentive reader will feel uncomfortable at this point about the undeterminacy of the choice of a in the definition of $gets$: we will come back to this later).

From the outset, $\mathcal{P}A$ is a datatype similar to $A^\star$: both share the same generative "grammar", or inductive "shape", defined by functor $\mathsf{F}\,X = 1 + A \times X$. However, a major distinction between the two types can be identified at once: in is not a right inverse of $outs$ on $\mathcal{P}A$, that is,

$$outs \cdot ins \quad = \quad id$$

does *not* hold. Because of the unordered structure of a set s one cannot reconstruct the steps along which it was built. So a powerset is not inductively generated, *i.e.* it is *non initial* in category-theoretical terms. This explains why $outs$ above could not be defined inductively (as happens with the similar coalgebra in datatype $A^\star$) and why set-theoretical equality had to become explicit. Nevertheless,

$$ins \cdot outs \quad = \quad id$$

holds (out is a right inverse of in). Altogether, this means that $1 + A \times \mathcal{P}A$ contains "more information" than $\mathcal{P}A$ alone. We convey this fact by writing

$$\begin{tikzcd} \mathcal{P}A \arrow[rr, bend left, "outs"] \arrow[rr, phantom, "\leq"] & & 1 + A \times \mathcal{P}A \arrow[ll, bend left, "ins"] \end{tikzcd} \tag{3.5}$$

The main consequence of the lack of in/out invertibility is that one cannot build catamorphisms over $\mathcal{P}A$: only $outs$-based hylomorphisms

$$\begin{tikzcd} \mathcal{P}A \arrow[r, "outs"] \arrow[d, "{[\![outs, g]\!]}"'] & 1 + A \times \mathcal{P}A \arrow[d, "{id + id \times [\![outs, g]\!]}"] \\ B & 1 + A \times B \arrow[l, "g"] \end{tikzcd}$$

can be defined over some suitable target $(1 + A \times \_)$-algebra g. For instance, the *card* operator

$$\begin{array}{lll} \mathit{card}\ s & \stackrel{\mathrm{def}}{=} & if\ \ s = \emptyset \\ & & then\ \ 0 \\ & & else\ \ \begin{array}{ll} let & a \in s \\ in & 1 + \mathit{card}\,(s - \{a\}) \end{array} \end{array}$$

counting the number of elements of a finite set is one such hylomorphism $[\![outs, g]\!]$, for $B = \mathbb{N}_0$ and $g = [\ \underline{0}, \mathit{succ} \cdot \pi_2\ ]$.

We will adopt notation $\{\!|g|\!\}$ as an abbreviation of $[\![outs, g]\!]$, itself an alternative notation for $(\![g]\!) \cdot [\!(outs)\!]$ over $A^\star$ finite sequences:

$$\{\!|g|\!\} \quad \stackrel{\mathrm{def}}{=} \quad (\![g]\!) \cdot [\!(outs)\!] \tag{3.6}$$

Due to the nondeterminism of $outs$ (*cf.* $gets$) the g algebra must be insensitive to the order of A values picked by $gets$ — otherwise, $\{\!|g|\!\}$ would be a relation and not a function. This can be expressed as follows, where g_2 is the second alternative of $g = [\ g_1 , g_2\ ]$:

$$g_2(a, g_2(a', b)) \quad = \quad g_2(a', g_2(a, b)) \tag{3.7}$$

Which properties of catamorphisms extend to $\{\!|g|\!\}$? The powerset fusion-law

$$f \cdot \{\!|g|\!\} = \{\!|h|\!\} \quad \Leftarrow \quad f \cdot g = h \cdot (id + id \times f) \tag{3.8}$$

can be drawn straight away from ana-cata decomposition followed by finite-sequence cata-fusion:

$$\begin{array}{cl} & f \cdot \{\!|g|\!\} = \{\!|h|\!\} \\ \Leftrightarrow & \qquad \{\text{ definition of powerset morphism}\} \\ & f \cdot (\![g]\!) \cdot [\!(outs)\!] = (\![h]\!) \cdot [\!(outs)\!] \\ \Leftarrow & \qquad \{\text{ composition by equals }\} \\ & f \cdot (\![g]\!) = (\![h]\!) \\ \Leftarrow & \qquad \{\text{ cata-fusion }\} \\ & f \cdot g = h \cdot (id + id \times f) \end{array}$$

Powerset cancellation, however, will no longer remain valid, since

$$\{\!|g|\!\} \cdot ins \quad = \quad g \cdot (id + id \times \{\!|g|\!\}) \tag{3.9}$$

does not hold in general. Think for instance of $\{\!|g|\!\} = \mathit{card}$. Then (3.9) would entail $\mathit{card} \cdot puts = \mathit{succ} \cdot \mathit{card} \cdot \pi_2$ which is false wherever the first argument of $puts$ belongs to the second one.

Some algebras g will obey (3.9) and for those we have an interesting property, a consequence of cata-fusion 2.61):

$$\{\!|g|\!\} \cdot \mathit{elems} \quad = \quad (\!|g|\!) \tag{3.10}$$

where $\mathit{elems} \stackrel{\mathrm{def}}{=} (\!|in|\!)$ is the function which "setifies" a finite list. This expresses the fact that the second alternative g_2 of $g = [\, g_1 , g_2 \,]$ is insensitive not only to order but also to repetition on its first argument (a kind of idempotency).

An example of this situation is provided by the powerset (type) functor, which is defined by

$$\mathcal{P} f \quad = \quad \{\!|ins \cdot (id + f \times id)|\!\} \tag{3.11}$$

and boils down to pointwise set-theoretical comprehension:

$$(\mathcal{P} f) s \quad = \quad \{ f\, a \mid a \in s \}$$

It can be easily checked that $\mathcal{P} f$ satisfies (3.9):

$$\begin{array}{ll}
 & \mathcal{P} f \cdot ins = (ins \cdot (id + f \times id)) \cdot (id + id \times \mathcal{P} f) \\
\Leftrightarrow & \qquad \{ + \text{ and } \times \text{ functors and identity} \} \\
 & \mathcal{P} f \cdot ins = ins \cdot (id + f \times \mathcal{P} f) \\
\Leftrightarrow & \qquad \{ \text{ polymorphism of } ins \text{ (natural transformation) } \} \\
 & \mathrm{TRUE}
\end{array}$$

Powerset-absorption

$$\{\!|g|\!\} \cdot (\mathcal{P} f) \quad = \quad \{\!|g \cdot (id + f \times id)|\!\} \tag{3.12}$$

is applicable to algebras g satisfying (3.9) (our unknown is α):

$$\begin{array}{ll}
 & \{\!|g|\!\} \cdot \mathcal{P} f = \{\!|\alpha|\!\} \\
\Leftrightarrow & \qquad \{ \text{ powerset-functor definition (3.11) } \} \\
 & \{\!|g|\!\} \cdot \{\!|ins \cdot (id + f \times id)|\!\} = \{\!|\alpha|\!\} \\
\Leftrightarrow & \qquad \{ \text{ powerset-morphism definition (3.6), twice } \} \\
 & \{\!|g|\!\} \cdot (\!|ins \cdot (id + f \times id)|\!) \cdot [\!(outs)\!] = (\!|\alpha|\!) \cdot [\!(outs)\!] \\
\Leftarrow & \qquad \{ \text{ composition of equals by equals} \} \\
 & \{\!|g|\!\} \cdot (\!|ins \cdot (id + f \times id)|\!) = (\!|\alpha|\!) \\
\Leftarrow & \qquad \{ \text{ cata-fusion (2.61) } \} \\
 & \{\!|g|\!\} \cdot ins \cdot (id + f \times id) = \alpha \cdot (id + id \times \{\!|g|\!\}) \\
\Leftrightarrow & \qquad \{ \text{ } g \text{ is assumed to obey to property (3.9) } \}
\end{array}$$

$$
\begin{array}{ll}
& g \cdot (id + id \times \{\!|g|\!\}) \cdot (id + f \times id) = \alpha \cdot (id + id \times \{\!|g|\!\}) \\
\Leftrightarrow & \quad \{ \text{ bi-functors } + \text{ and } \times \} \\
& g \cdot (id + id \cdot f \times \{\!|g|\!\} \cdot id) = \alpha \cdot (id + id \times \{\!|g|\!\}) \\
\Leftrightarrow & \quad \{ \ id \text{ is natural (1.11) } \} \\
& g \cdot (id + f \cdot id \times id \cdot \{\!|g|\!\}) = \alpha \cdot (id + id \times \{\!|g|\!\}) \\
\Leftrightarrow & \quad \{ \text{ bi-functors } + \text{ and } \times \text{ again } \} \\
& g \cdot (id + f \times id) \cdot (id + id \times \{\!|g|\!\}) = \alpha \cdot (id + id \times \{\!|g|\!\}) \\
\Leftarrow & \quad \{ \text{ composition of equals by equals} \} \\
& g \cdot (id + f \times id) = \alpha
\end{array}
$$

Powerset-absorption (3.12) is helpful in showing that the powerset-map $\mathcal{P}f$ commutes with composition:

$$
\begin{array}{ll}
\multicolumn{2}{l}{\mathcal{P}g \cdot \mathcal{P}f} \\
= & \quad \{ \text{ powerset definition} \} \\
& \{\!|ins \cdot (id + g \times id)|\!\} \cdot (\mathcal{P}f) \\
= & \quad \{ \text{ powerset-absorption (3.12)} \} \\
& \{\!|(ins \cdot (id + g \times id)) \cdot (id + f \times id)|\!\} \\
= & \quad \{ + \text{ and } \times \text{ functors, composition and identity} \} \\
& \{\!|(ins \cdot (id + (g \cdot f) \times id)|\!\} \\
= & \quad \{ \text{ powerset definition} \} \\
& \mathcal{P}(g \cdot f)
\end{array}
$$

Finally, the fact that $\mathcal{P}\ \_$ (3.11) indeed defines a functor requires property

$$\mathcal{P}id \quad = \quad id$$

to hold. From (3.11) we draw

$$
\begin{array}{lll}
\mathcal{P}id & = & \{\!|ins \cdot (id + id \times id)|\!\} \\
& = & \quad \{ \text{ composition and identity} \} \\
& & \{\!|ins|\!\}
\end{array}
$$

However, does powerset reflection

$$\{\!|ins|\!\} \quad = \quad id_{(\mathcal{P}A)} \tag{3.13}$$

hold? The truth of this fact can no longer be proved on the basis of an universal property because ins is not initial, *i.e.*, a bijection. We shall have to use induction. But this raises another question: *"which" induction?* — a pertinent question given the fact that $\mathcal{P}A$ is not inductively generated!

Exercise 3.1 Draw the natural property of *elems*,

$$(\mathcal{P} g) \cdot \mathit{elems} \quad = \quad \mathit{elems} \cdot g^{\star} \tag{3.14}$$

from (3.10).
□

Exercise 3.2 ZF-set-abstraction

$$\{f\, a \mid \ a \in s \wedge (p\, a)\}$$

corresponds to powerset filter-and-map:

$$filter(f, p) \quad \stackrel{\mathrm{def}}{=} \quad \mathit{Union} \cdot \mathcal{P}(p \rightarrow sings \cdot f, \underline{\emptyset}) \tag{3.15}$$

where

$$\mathit{Union} \quad = \quad (\![\, \underline{\emptyset}, \cup \,]\!) \tag{3.16}$$

is finitely distributed union. Show that, for $p = \underline{\text{TRUE}}$, $\{f\, a \mid \ a \in s \wedge (p\, a)\} = \mathcal{P} f$. **Hint**: resort to powerset-absorption (3.12).
□

3.2 Structural Induction

The question above brings us into the core of the inductive proof method in general: the principle of *structural induction* based on *well-founded* relations. A relation $\prec \subseteq A \times A$ is said to be well-founded if any nonempty subset C of A has a minimum, that is, if

$$\forall \emptyset \subset C \subseteq A : (\exists m \in C : \prec_m \cap\, C = \emptyset) \tag{3.17}$$

holds, where $\prec_m$ denotes

$$\prec_m = \{a \in A \mid \ a \prec m\}$$

The principle of *structural induction* based on well-founded relation $(A; \prec)$ is stated as follows: in order to prove the validity of a predicate p on A,

$$\forall a \in A : p\, a$$

proceed as follows:

1. *Induction basis*: prove
$$p\, m$$
for all $m \in A$ such that $\prec_m = \emptyset$.

2. *Inductive step*: let $a \in A$ be such that $\prec_a \supset \emptyset$.

(a) *Induction hypothesis*:

$$\forall x \prec a : p\,x$$

holds.

(b) *Step*: prove

$$(\forall x \prec a : p\,x) \Rightarrow p\,a$$

□

For instance, well-founded relation $m \prec n \stackrel{\text{def}}{=} n > 0 \wedge m = n - 1$ supports the well-known first induction principle on the natural numbers. We want to prove fact $p\,n$ for an arbitrary predicate p over $\mathbb{N}_0$. Clearly, $\prec_0 = \emptyset$ and, for $n > 0$, $\prec_n = \{n-1\}$. Therefore,

1. *Induction basis* ($n = 0$): prove

$$p\,0$$

2. *Inductive step* ($n > 0$):

(a) *Induction hypothesis*: assume $p(n-1)$

(b) *Step*: prove

$$p(n-1) \quad \Rightarrow \quad p\,n$$

□

Exercise 3.3 Let f be the following recursive function

$$\begin{array}{rcl} f & : & \mathbb{N}_0 \longrightarrow \mathbb{N}_0 \\ f(k) & \stackrel{\text{def}}{=} & \left\{ \begin{array}{l} k = 0 \Rightarrow 0 \\ k > 0 \Rightarrow odd(k) + f(k-1) \end{array} \right. \end{array} \tag{3.18}$$

where

$$\begin{array}{rcl} odd & : & \mathbb{N} \longrightarrow \mathbb{N} \\ odd(j) & \stackrel{\text{def}}{=} & 2j - 1 \end{array}$$

Show by $\mathbb{N}_0$ induction that f computes the square of k, *i.e.* that $f(k) = k^2$. (So the sum of all first n odd numbers is n^2).

□

3.3 Well-founded coalgebras and induction

Well-foundedness is not only closely related to induction but also to termination. A given hylomorphism $[\![g, h]\!]$ will not terminate unless h is a *well-founded coalgebra*. A coalgebra is said to be well-founded if its *accessibility relation* is well-founded. The accessibility relation $\prec_h$ implicit in coalgebra $A \xrightarrow{h} \mathsf{F}\,A$ is defined as follows:

$$b \prec_h a \quad \Leftrightarrow \quad b \in_{\mathsf{F}} (h\,a)$$

where $\in_{\mathsf{F}}$ denotes structural membership. This is defined structurally over polynomial functors as follows:

$$
\begin{aligned}
k \in_C x &\stackrel{\mathrm{def}}{=} FALSE && (3.19)\\
k \in_{\lambda X.X} x &\stackrel{\mathrm{def}}{=} k = x && (3.20)\\
k \in_{\mathsf{F}\times\mathsf{G}} (x, y) &\stackrel{\mathrm{def}}{=} k \in_{\mathsf{F}} x \vee k \in_{\mathsf{G}} y && (3.21)\\
k \in_{\mathsf{F}+\mathsf{G}} x &\stackrel{\mathrm{def}}{=} \left\{ \begin{array}{l} k \in_{\mathsf{F}} y \Leftarrow x = i_1\, y \\ k \in_{\mathsf{G}} z \Leftarrow x = i_2\, z \end{array} \right. && (3.22)
\end{aligned}
$$

For instance, let $\mathbb{N}_0 \xrightarrow{h} 1 + \mathbb{N}_0$ be coalgebra $h \stackrel{\mathrm{def}}{=} (! + pred) \cdot =_0$. One has

$$
k \in_{1+X} x \quad = \quad \left\{ \begin{array}{l} x = i_1\, y \Rightarrow \text{FALSE} \\ x = i_2\, z \Rightarrow k = z \end{array} \right.
$$

that is

$$
k \in_{1+X} x \quad \Leftrightarrow \quad x = i_2\, k
$$

So,

$$
m \prec_h n \Leftrightarrow (h\, n) = i_2\, m \Leftrightarrow n > 0 \wedge m = n - 1
$$

Therefore, this coalgebra corresponds to the first induction principle just seen above.

Let us now go back to the $\{\!|g|\!\}$ hylomorphism. It can be checked that $outs$ (3.3) is well-founded and that its accessibility relation is defined over powerset $\mathcal{P}A$ as follows:

$$
r \prec_{outs} s \quad \stackrel{\mathrm{def}}{=} \quad s \supset \emptyset \wedge \exists a \in s : r = s - \{a\}
$$

This leads to the finite powerset induction principle which follows (we want to prove the validity of $\varphi\, s$ for every $s \in \mathcal{P}A$):

1. *Induction basis* ($s = \emptyset$): prove $\varphi\, \emptyset$.
2. *Inductive step* ($s \supset \emptyset$):
   (a) *Induction hypothesis*:
   $$\forall e \in s : \varphi(s - \{e\})$$
   (b) *Step*:
   $$\forall e \in s : \varphi(s - \{e\}) \Rightarrow (\varphi\, s)$$

□

We are ready now to prove powerset-reflection (3.13) by induction, *i.e.* the validity of $\varphi\, s \stackrel{\mathrm{def}}{=} \{\!|ins|\!\}\, s = s$. By calculation we get

$$
\{\!|ins|\!\}\, s \quad \stackrel{\mathrm{def}}{=} \quad \begin{array}{lll} if & s = \emptyset & \\ then & \emptyset & \\ else & let & e \in s \\ & in & \{e\} \cup (\{\!|ins|\!\}(s - \{e\})) \end{array}
$$

The base case $s = \emptyset$ is obvious. The induction hypothesis is $(\!|ins|\!)\,(s-\{e\}) = s-\{e\}$. A basic fact about sets, $\{e\} \cup (s - \{e\}) = s$, completes the proof.

Exercise 3.4 Establish the induction principle associated with the finite sequence datatype $A^\star$.
□

3.4 Datatype invariants and proof obligations

The formal specification of real-life problems normally involves datatypes affected by properties which formalize real-life conventions, laws, rules, norms or natural constraints.

Such particularly relevant properties of specifications should hold forever along programs lifetime and are called *invariant properties*. Other such "eternal" properties may hold as consequences of the underlying invariants.

A well-known example of an invariant concerns datatype

$$Date \cong 31 \times 12 \times \mathbb{N}$$

which requires (in the Gregorian calendar) a nontrivial 'in loco' invariant:

$$\begin{array}{l} dateOk : Date \longrightarrow 2 \\ dateOk(d,m,y) \stackrel{\text{def}}{=} \\ \quad \left\{ \begin{array}{lcl} m \in \{1,3,5,7,8,10,12\} & \Rightarrow & d \leq 31 \wedge \\ & & (\neg(y = 1582 \wedge m = 10) \\ & & \vee (d < 5) \vee (14 < d)) \\ m \in \{4,6,9,11\} & \Rightarrow & d \leq 30 \\ m = 2 \wedge leapYear\, y & \Rightarrow & d \leq 29 \\ m = 2 \wedge \neg leapYear\, y & \Rightarrow & d \leq 28 \end{array} \right. \end{array} \tag{3.23}$$

where

$$\begin{array}{rcl} leapYear & : & \mathbb{N} \longrightarrow 2 \\ leapYear\, y & \stackrel{\text{def}}{=} & rem(y, \left\{ \begin{array}{lcl} 1700 \leq y \wedge rem(y,100) = 0 & \Rightarrow & 400 \\ 1700 > y \vee rem(y,100) \neq 0 & \Rightarrow & 4 \end{array} \right.) = 0 \end{array}$$

Predicate $dateOk$ (3.23) is a good illustration of the *ad hoc* character of most constraints formally imposed to specifications. It arises from a natural, cosmological fact — the number of days of the year is not a natural number — and from several historical attempts to devise a norm able to finitely approximate such a number [1].

[1]The situation is actually more complex, because this invariant is "geography" dependent. In fact, the 1582 discontinuity (10 days omitted in October of that year) of the Gregorian calendar (3.23) was adopted much later in several countries — *e.g.* Poland (1586), Hungary (1587), Germany (1700), England (1752) *etc.* [Gof84]. (A world-wide "date" datatype taking into account current and past dating systems, *e.g.* Chinese, Indian, *etc.*, would be overwhelming in complexity.)

In practice, both the need to bring *legislation* into many computer applications and the overall interest in devising adequate models for real-life problems justify our study of datatype invariants and their impact in the formal specification life-cycle.

Data-type invariants are a burden for the software developer because formal arguments have to be produced ensuring that they are preserved by all the operations which create the corresponding data. Such arguments are called *proof obligations*.

Given $\alpha : A \longrightarrow 2$, let A_α denote the set

$$\{a \in A \mid \; \alpha\, a\}$$

that is, the result of imposing invariant α upon A, for example,

$$Date_{dateOk}$$

Suppose that one wishes to define a function $f : B \longrightarrow A_\alpha$. Due to the occurrence of invariant α on the codomain of f, its definition cannot be regarded as complete until the following proof obligation (called *invariant proof*) is discharged:

$$\forall b \in B : \alpha(f\, b) \tag{3.24}$$

Invariants may decorate function input domains as well as output domains. While in the latter case they are a burden and force a formal proof, in the former they are welcome because they provide an antecedent for the proof (*i.e.* they act as *preconditions*). For instance, the correctness of the definition of function $f : B_\beta \longrightarrow A_\alpha$ is established by

$$\forall b \in B : \beta\, b \Rightarrow \alpha(f\, b) \tag{3.25}$$

Wherever B is an inductive or quasi-inductive datatypes, proof-obligations will be carried out by means induction proofs as studied in the previous section. In many situations B is the same as A (ibib. β and α) or, more generally, A_α occurs in the domain of f. In this case one says that f is an A_α "transformer" and that invariant α is *maintained* by α.

Datatype invariants restrict pre-existing datatypes wherever on reaches the limits of formal modelling by discrete mathematics techniques and logic has to be added to the design. Sometimes one finds such restricted datatypes so useful that particular notation is invented are properties are investigated for them. The finite mapping datatype discussed in the following section is one such piece of formal notation.

Exercise 3.5 Let the following function specify a particular list insertion discipline:

$$\begin{array}{lcl}
ins & : & \mathbb{N}^* \times \mathbb{N} \longrightarrow \mathbb{N}^* \\
ins(l,a) & \stackrel{\mathrm{def}}{=} & \left\{ \begin{array}{lcl}
l = [\,] & \Rightarrow & [a] \\
l \neq [\,] & \Rightarrow & \begin{array}{ll} let & h = \mathsf{hd}\, l \\ & t = \mathsf{tl}\, l \\ in & \left\{ \begin{array}{lcl} a = h & \Rightarrow & l \\ a < h & \Rightarrow & cons(a,l) \\ a > h & \Rightarrow & cons(h, ins(t,a)) \end{array} \right. \end{array}
\end{array} \right.
\end{array}$$

Does ins maintain the following datatype invariant on finte lists?

$$\phi(l) \stackrel{\mathrm{def}}{=} length(l) = card(\mathsf{elems}\, l)$$

Produce an (indutive) proof or a counter-example.

□

3.5 Binary relations and finite mappings

There are two standard specializations of the powerset datatype which are particularly useful for the formal specification practitioner: finite binary relations (n-ary relations in general) and finite mappings.

A finite binary relation is an inhabitant of datatype $\mathcal{P}(A \times B)$, for finite A and B. A finite mapping is a finite binary relation subject to a datatype invariant *fdp* establishing an "$A \to B$" functional dependency:

$$fdp\, r \stackrel{\text{def}}{=} \forall a \in A : \mathsf{card}\,((collect\, r)\, a) \leq 1$$

over datatype $\mathcal{P}(B \times A)$, for finite A and B, where $collect$ is an isomorphism:

$$\begin{tikzcd} \mathcal{P}(A \times B) \arrow[rr, bend left, "collect"] \arrow[rr, phantom, "\cong"] & & (\mathcal{P}B)^A \arrow[ll, bend left, "discollect"] \end{tikzcd} \tag{3.26}$$

which stems from a more basic one

$$\begin{tikzcd} \mathcal{P}A \arrow[rr, bend left, "cf"] \arrow[rr, phantom, "\cong"] & & 2^A \arrow[ll, bend left, "zf"] \end{tikzcd} \tag{3.27}$$

where

$$\begin{aligned} zf\,\phi &\stackrel{\text{def}}{=} \{a \in A \mid \phi\, a\} \\ cf\, s &\stackrel{\text{def}}{=} \lambda a. a \in s \end{aligned}$$

The finite mapping data model (restricting binary relations) is so useful in practice that special notation is introduced

$$A \rightharpoonup B = \mathcal{P}(A \times B)_{fdp} \tag{3.28}$$

as well as dedicated operators [2]:

$$\begin{aligned}
\mathit{dom} &\stackrel{\text{def}}{=} (\!|ins \cdot (id + \pi_1 \times id)|\!) \quad \textit{/*finite mapping domain */} && (3.29)\\
\mathit{rng} &\stackrel{\text{def}}{=} (\!|ins \cdot (id + \pi_2 \times id)|\!) \quad \textit{/*finite mapping range */} && (3.30)\\
A \rightharpoonup f &= (\!|ins \cdot (id + (id \times f) \times id)|\!) \quad \textit{/*finite mapping type functor */} && (3.31)\\
(|s) &\stackrel{\text{def}}{=} filter(id, (\in s) \cdot \pi_1) \quad \textit{/*domain restriction */} && (3.32)\\
(\backslash s) &\stackrel{\text{def}}{=} filter(id, \neg \cdot (\in s) \cdot \pi_1) \quad \textit{/*domain subtraction */} && (3.33)\\
\mathit{pap} &\stackrel{\text{def}}{=} the \cdot \mathit{rng} \cdot | \cdot (id \times sings) \quad \textit{/*application (the is partial) */} && (3.34)\\
\sigma \dagger \tau &\stackrel{\text{def}}{=} (\sigma \setminus \mathsf{dom}\,\tau) \cup \tau \quad \textit{/*mapping over-writing */} && (3.35)
\end{aligned}$$

[2] We adopt the following *curryied notation* conventions concerning a binary operator $C \xleftarrow{\theta} A \times B$: $(a\theta)$ denotes $\overline{\theta}\, a$; (θb) denotes $\overline{flip\,\theta}\, b$ (where $flip\,\theta = \theta \cdot swap$ swaps the order of arguments of a binary function); θ_x denotes either $(x\theta)$ or (θx) depending upon the context.

Partial map application $pap(\sigma, a)$ is abbreviated by $\sigma\, a$, by analogy with (total) functions. A pair (a, b) in $A \rightharpoonup B$ is usually denoted by $a \mapsto b$. Converted to pointwise notation using these conventions, a finite mapping morphism $f = \{\!|[\, \underline{k}, h \,]|\!\}$ takes the following shape:

$$f\, \sigma \;\stackrel{\mathrm{def}}{=}\; \begin{array}{lll} if \;\; \sigma = \{\} & & \\ then \;\; k & & \\ else & let & a \in \mathit{dom}\, \sigma \\ & & \sigma' = \sigma \setminus \{a\} \\ & in & h((a, \sigma\, a), f\, \sigma') \end{array} \tag{3.36}$$

In pointwise notation finite mappings are enumerated by writing *e.g.*

$$\{1 \mapsto a, 2 \mapsto b, 3 \mapsto c\}.$$

This notation extends to mapping-comprehension, as in *e.g.*

$$\{i \mapsto i + 1 \mid\; i \in 3\}$$

which denotes the same mapping as $\{1 \mapsto 2, 2 \mapsto 3, 3 \mapsto 4\}$ or in the following pointwise definition of the very useful "mapping join" operator:

$$\sigma \bowtie \tau \;\stackrel{\mathrm{def}}{=}\; \{a \mapsto (\sigma\, a, \tau\, a) \mid a \in \mathit{dom}\, \sigma \cap \mathit{dom}\, \tau\} \tag{3.37}$$

3.6 An overview of the powerset and finite mapping algebras

A permutative law is one which can be regarded as stating a homomorphism between two F-algebras of the identity functor (unary case) or of the product diagonal functor $\mathsf{F}\, X = X \times X$ (binary case). Using diagrams, one will have

Unary

$$\begin{tikzcd} A \arrow[d, "f"'] & A \arrow[l, "\alpha"'] \arrow[d, "f"] \\ B & B \arrow[l, "\beta"] \end{tikzcd}$$

$$f \cdot \alpha \;=\; \beta \cdot f$$

Binary

$$\begin{tikzcd} A \arrow[d, "f"'] & A \times A \arrow[l, "\alpha"'] \arrow[d, "f \times f"] \\ B & B \times B \arrow[l, "\beta"] \end{tikzcd}$$

$$f \cdot \alpha \;=\; \beta \cdot (f \times f)$$

In pointwise notation, *e.g.* for the binary case, one has:

$$f(\alpha(x, y)) \;=\; \beta(f\, x, f\, y)$$

In the following collection of permutative laws, $|_S$ abbreviates curried map restriction (that is, $|_S = | \cdot \langle id, \underline{S} \rangle$).

$$
\begin{aligned}
elems \cdot + &= \cup \cdot (elems \times elems) & (3.38)\\
length \cdot + &= + \cdot (length \times length) & (3.39)\\
dom \cdot \cup &= \cup \cdot (dom \times dom) & (3.40)\\
dom \cdot \dagger &= \cup \cdot (dom \times dom) & (3.41)\\
rng \cdot \cup &= \cup \cdot (rng \times rng) & (3.42)\\
\backslash_S \cdot \cup &= \cup \cdot (\backslash_S \times \backslash_S) & (3.43)\\
|_S \cdot \dagger &= \dagger \cdot (|_S \times |_S) & (3.44)\\
\backslash_S \cdot \dagger &= \dagger \cdot (\backslash_S \times \backslash_S) & (3.45)\\
|_S \cdot \bowtie &= \bowtie \cdot (|_S \times |_S) & (3.46)
\end{aligned}
$$

3.7 Exercises

Exercise 3.6 Define the natural properties of the main mapping operators, *e.g.* *rng*, $\bowtie$ *etc.*
□

Exercise 3.7 Define set union $s \cup t$ in terms of powerset morphisms. **Hint**: describe $(\cup t)$ as a powerset morphism.
□

Exercise 3.8 Suppose that $A \rightharpoonup B$ describes the statement (B) of all accounts (A) stored in the information system of bank *X*. Let $s \subseteq A$ be the set of accounts which have been selected by the bank's manager to be awarded the "TOP X 2000" prize, whereby all such accounts will be credited some amount dependent on the accounts' statement. Let $B \xrightarrow{f} B$ be the formula which updates each account balance in order to credit the award.

One of the analysts of the "TOP X 2000" package wrote

$$\dagger \cdot \langle id, (A \rightharpoonup f) \cdot |_s \rangle$$

as a formal specification of the operation which will implement the award procedure.

Inspect this solution by converting it to pointwise notation and answer informally to the following questions: **(a)** what will happen if the bank manager includes in s accounts which no longer exist? **(b)** should the bank manager decide to award all current accounts, find an expression simpler than the above to describe the award procedure. **(c)** and what if the award amounts to nothing to be credited?
□

Exercise 3.9 Function f in exercise 3.7 is an instance of the co-called "selective update" functional,

$$
\begin{array}{ccc}
B & & A \rightharpoonup B \\
{\scriptstyle f}\downarrow & & \downarrow{\scriptstyle \mathsf{U}_S\, f} \\
B & & A \rightharpoonup B
\end{array}
$$

defined thus:

$$\mathsf{U}_S\, f \stackrel{\mathrm{def}}{=} \dagger \cdot \langle id, (A \rightharpoonup f) \cdot |_S \rangle \qquad (3.47)$$

1. Write $\mathsf{U}_S\ f$ in pointwise notation.
2. Prove of contradict the following properties of U_S:

$$\mathsf{U}_S\ id = id \tag{3.48}$$

$$\mathsf{U}_\emptyset\ f = id \tag{3.49}$$

$$\mathsf{U}_S\ (f \cdot g) = (\mathsf{U}_S\ f) \cdot (\mathsf{U}_S\ g) \tag{3.50}$$

$$(A \rightharpoonup \underline{c}) \cdot \mathsf{U}_S\ f = A \rightharpoonup \underline{c} \tag{3.51}$$

$$(\mathsf{U}_S\ f) \cdot (A \rightharpoonup g) = (A \rightharpoonup g) \cdot (\mathsf{U}_S\ h) \quad \text{if} \quad f \cdot g = g \cdot h \tag{3.52}$$

□