Going functional in Relation Algebra

J.N. Oliveira

Slides for lecture of 7-1-2021

CSI 2021

University of Minho & INESC TEC

Power transpose

Given arbitrary relation $A \xrightarrow{R} B$, define the function

$$\Lambda R : A \to \mathfrak{P}\ B$$
$$\Lambda R\ a = \{\, b \mid b\ R\ a \,\}$$

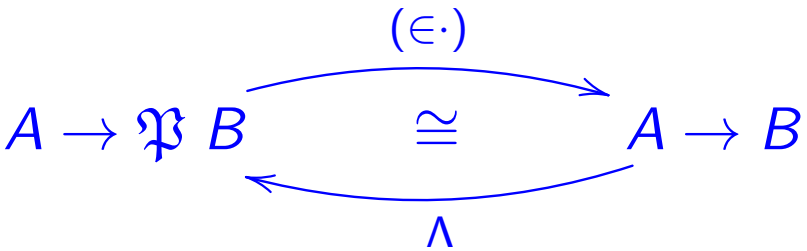
which is such that:

$$\Lambda R = f \quad \Leftrightarrow \quad \langle \forall\ b, a\ ::\ b\ R\ a \Leftrightarrow b \in f\ a \rangle \tag{1}$$

That is:

$$f = \Lambda R \ \Leftrightarrow\ \in \cdot f = R \tag{2}$$

cf.



Alloy and the power transpose

Alloy uses this transpose all the time.

Everytime you write $a.R$ in Alloy, what Alloy does is to apply the power transpose to a:

$$a.R = \Lambda R\ a = \{\, b \mid b\ R\ a \,\}$$

It actually does more, when allowing a set s in $s.R$: as the powerset forms a **monad**,

$$s \cdot R = s \gg\!= \Lambda R = \mathbf{do}\ \{\, a \leftarrow s; \Lambda R\ a \,\}$$

NB: In the powerset monad $u\ a = \{\, a \,\}$ and $\mu = \bigcup$. Thus:

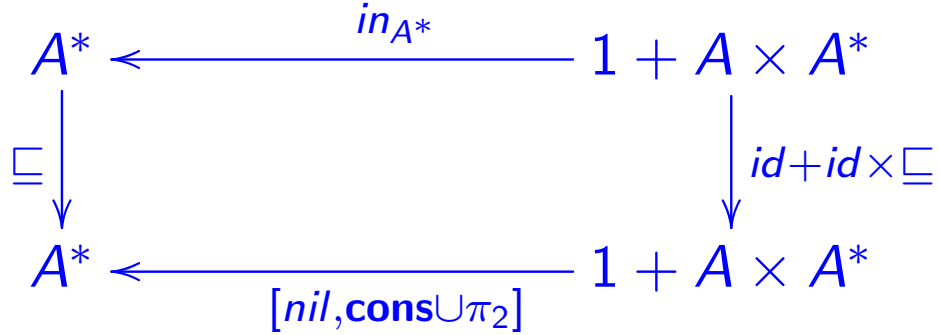
$$s \cdot R = \bigcup \{\, \Lambda R\ a \mid a \in s \,\}$$

Subsequence relation

Relational catamorphism:

$$\sqsubseteq : A^* \leftarrow A^*$$
$$\sqsubseteq = (\!|[nil, \mathbf{cons} \cup \pi_2]|\!) \tag{3}$$

Diagram:



Going pointwise

$$
\begin{array}{ll}
 & \sqsubseteq = (\!|\, [nil, \mathbf{cons} \cup \pi_2] \,|\!) \\
\Leftrightarrow & \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
 & \sqsubseteq \cdot [nil, \mathbf{cons}] = [nil, \mathbf{cons} \cup \pi_2] \cdot (id + id \times \sqsubseteq) \\
\Leftrightarrow & \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
 & \left\{ \begin{array}{l} \sqsubseteq \cdot nil = nil \\ \sqsubseteq \cdot \mathbf{cons} = (\mathbf{cons} \cup \pi_2) \cdot (id \times \sqsubseteq) \end{array} \right. \\
\Leftrightarrow & \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
 & \left\{ \begin{array}{l} y \sqsubseteq [\,] \Leftrightarrow y = [\,] \\ \\ y \sqsubseteq (h : t) \Leftrightarrow \left\langle \begin{array}{c} \exists\ h', t' : \\ y\ (\mathbf{cons} \cup \pi_2)\ (h', t') : \\ (h', t')\ (id \times \sqsubseteq)\ (h, t) \end{array} \right\rangle \end{array} \right.
\end{array}
$$

Going pointwise

$$\left\{\begin{array}{l} y \sqsubseteq [\,] \Leftrightarrow y = [\,] \\ \\ y \sqsubseteq (h : t) \Leftrightarrow \left\langle \begin{array}{c} \exists\ h', t' : \\ y\ (\mathbf{cons} \cup \pi_2)\ (h', t') : \\ (h', t')\ (id \times \sqsubseteq)\ (h, t) \end{array} \right\rangle \end{array}\right.$$

$\Leftrightarrow$ { . . . . . }

$$\left\{\begin{array}{l} y \sqsubseteq [\,] \Leftrightarrow y = [\,] \\ y \sqsubseteq (h : t) \Leftrightarrow \langle \exists\ t' \ :\ y\ (\mathbf{cons} \cup \pi_2)\ (h, t') :\ t' \sqsubseteq t \rangle \end{array}\right.$$

$\Leftrightarrow$ { . . . . . }

$$\left\{\begin{array}{l} y \sqsubseteq [\,] \Leftrightarrow y = [\,] \\ y \sqsubseteq (h : t) \Leftrightarrow \langle \exists\ t' \ :\ t' \sqsubseteq t :\ y = (h : t') \vee y = t' \rangle \end{array}\right.$$

Going functional

Implementing function $\Lambda\sqsubseteq$ in Haskell, using the **list monad** as an implementation of the powerset monad:

$$(\Lambda\sqsubseteq)\ [\,] = \mathbf{return}\ [\,]$$
$$(\Lambda\sqsubseteq)\ (h : t) = \mathbf{do}\ \{t' \leftarrow (\Lambda\sqsubseteq)\ t; [t', h : t']\}$$

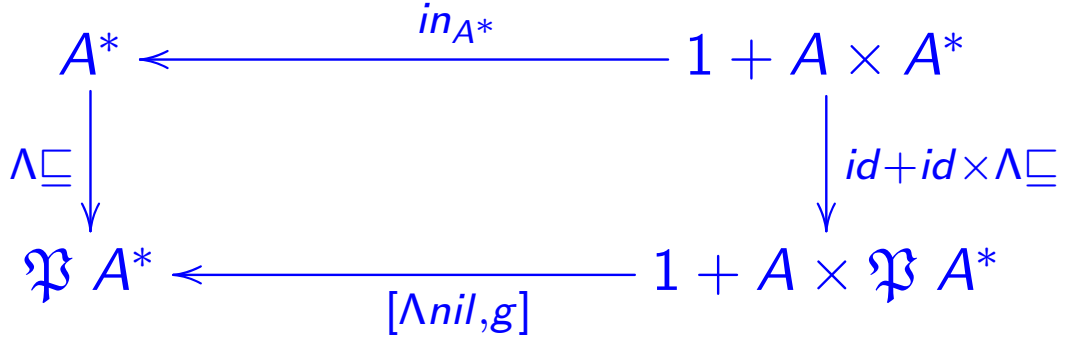
Example, where `subseqf` is the concrete syntax for $\Lambda\sqsubseteq$:

```
*CSI2021> subseqf "Ana"
["","A","n","An","a","Aa","na","Ana"]
```

and so on.

Future readings

Explanation for monadic code? Compare with the following diagram:



where

$$g\ (h, s) = \mathbf{do}\ \{ t \leftarrow s; \{ h : t, t \} \}.$$

NB: More details in the Annex.

Annex

Eilenberg-Wright Lemma:

$$X = (\!|R|\!) \quad \Leftrightarrow \quad \Lambda X = (\!|\Lambda(R \cdot \mathfrak{F} \in)|\!) \tag{4}$$

Thus

$$\begin{array}{ll}
 & \sqsubseteq = (\!|[nil, \mathbf{cons} \cup \pi_2]|\!) \\
\Leftrightarrow & \quad \{ \text{ Eilenberg-Wright Lemma } \} \\
 & \Lambda\sqsubseteq = (\!|\Lambda[nil, (\mathbf{cons} \cup \pi_2) \cdot (id \times \in)]|\!) \\
\Leftrightarrow & \quad \{ \ \Lambda[R, S] = [\Lambda R, \Lambda S] \ \} \\
 & \Lambda\sqsubseteq = (\!|[\Lambda nil, \Lambda((\mathbf{cons} \cup \pi_2) \cdot (id \times \in))]|\!) \\
\Leftrightarrow & \quad \{ \ \Lambda(R \cdot f) = \Lambda R \cdot f;\ \Lambda id = \eta \ \} \\
 & \Lambda\sqsubseteq = (\!|[\eta \cdot nil, \Lambda((\mathbf{cons} \cup \pi_2) \cdot (id \times \in))]|\!) \quad \square
\end{array}$$

Annex

$$
\left\{
\begin{array}{l}
\Lambda\sqsubseteq \cdot \mathit{nil} = \eta \cdot \mathit{nil} \\
\Lambda\sqsubseteq \cdot \mathbf{cons} = \Lambda((\mathbf{cons} \cup \pi_2) \cdot (\mathit{id} \times \in)) \cdot (\mathit{id} \times \Lambda\sqsubseteq)
\end{array}
\right.
$$

$\Leftrightarrow$ $\{$ variables $\}$

$$
\left\{
\begin{array}{l}
\Lambda\sqsubseteq [\,] = \mathbf{return}\ [\,] \\
\Lambda\sqsubseteq (h : t) = \Lambda((\mathbf{cons} \cup \pi_2) \cdot (\mathit{id} \times \in))\ (h, \Lambda\sqsubseteq\ t)
\end{array}
\right.
$$

$\Leftrightarrow$ $\{\ \Lambda(R \cdot S) = \Lambda R \bullet \Lambda S\ \}$

$$
\left\{
\begin{array}{l}
\Lambda\sqsubseteq [\,] = \mathbf{return}\ [\,] \\
\Lambda\sqsubseteq (h : t) = (\Lambda(\mathbf{cons} \cup \pi_2) \bullet \Lambda(\mathit{id} \times \in))\ (h, \Lambda\sqsubseteq\ t)
\end{array}
\right.
$$

Annex

Then, from

$$\Lambda(f \cup g)\ a = \{f\ a, g\ a\}$$
$$(f \bullet g)\ x = \mathbf{do}\ \{a \leftarrow g\ x; f\ a\}$$

we get:

$$\Lambda\sqsubseteq (h : t) = \mathbf{do}\ \{(a, b) \leftarrow \Lambda(id \times \in)\ (h, \Lambda\sqsubseteq\ t); \{a : b, b\}\}$$

And finally, from

$$\Lambda(id \times \in)\ (h, t) = \{(x, y) \mid x = h \wedge y \in t\} = \{(h, y) \mid y \in t\}$$

one gets:

$$\Lambda\sqsubseteq (h : t) = \mathbf{do}\ \{t' \leftarrow \Lambda\sqsubseteq\ t; \{h : t', t'\}\}$$