Constructive Algorithmics (Part II)

Zhenjiang HU
University of Tokyo

Homomorphisms

Points in this lesson:

- Formalization of homomorphism
- Program specification in homomorphism
- Point-free calculation rules for manipulating homomorphism
- Equational reasoning without induction

A Problem

Given is a sequence x and a predicate p. Required is an efficient algorithm for computing some longest segment of x, all of whose elements satisfy p.

$$lsp\ even\ [3,1,4,1,5,9,2,6,5] = [2,6]$$

Homomorphisms

A homomorphism from a monoid $(\alpha, \oplus, id_\oplus)$ to a monoid $(\beta, \otimes, id_\otimes)$ is a function h satisfying the two equations:

$$\begin{array}{lcl} h\ id_\oplus & = & id_\otimes \\ h\ (x \oplus y) & = & h\ x \otimes h\ y \end{array}$$

Exercise: Prove that h is a homomorphism iff the following holds.

$$h \cdot \oplus / = \otimes / \cdot h*$$

Examples

- Since $f*$ is a homomorphism from $([\alpha], +\!\!+, [])$ to $([\beta], +\!\!+, [])$ whenever $f : \alpha \rightarrow \beta$, we have

$$f * \cdot +\!\!+ / = +\!\!+ / \cdot f * *$$

  which is the *map promotion* rule of the previous lecture.

- Since $\oplus/$ is a homomorphism from $([\alpha], +\!\!+, [])$ to $(R, \oplus, id_\oplus)$ whenever $(\oplus) : R \rightarrow R \rightarrow R$, we have

$$\oplus/ \cdot +\!\!+ / = \oplus/ \cdot (\oplus/)*$$

  which is the *reduce promotion* rule of the previous lecture.

Uniqueness Property

We have the fact that $([\alpha], +\!\!+, [])$ is a free monoid, that is for each monoid $(\beta, \oplus, id_\oplus)$ there is a unique homomorphism h from $([\alpha], +\!\!+, [])$ to $(\beta, \oplus, id_\oplus)$.

This homomorphism is determined by the values of h on singletons. That is, for each $f : \alpha \to \beta$, the additional equation

$$h\ [a] = f\ a$$

fixes h completely.

Characterization of Homomorphisms

Lemma 1 *Every homomorphism on lists can be expressed as the conposition of a reduction with a map, and every such combination is a homomorphisms.*

More precisely, suppose

$$
\begin{array}{lcl}
h\ [] & = & id_{\oplus} \\
h\ [a] & = & f\ a \\
h\ (x +\!\!+\, y) & = & h\ x \oplus h\ y
\end{array}
$$

then, $h = \oplus/ \cdot f*$. *Conversely, if* h *has this form, then* h *is a homomorphism.*

Proof

$\Rightarrow$:

$$
\begin{array}{ll}
 & h \\
= & \quad \{ \text{ definition of } id \ \} \\
 & h \cdot id \\
= & \quad \{ \text{ identity lemma (can you prove it?) } \} \\
 & h \cdot +\!\!+/ \cdot [\cdot]* \\
= & \quad \{ \ h \text{ is a homomorphism } \} \\
 & \oplus/ \cdot h * \cdot [\cdot]* \\
= & \quad \{ \text{ map distributivity } \} \\
 & \oplus/ \cdot (h \cdot [\cdot])* \\
= & \quad \{ \text{ definition of } h \text{ on singleton } \} \\
 & \oplus/ \cdot f*
\end{array}
$$

$\Leftarrow$: We reason that $h = \oplus/ \cdot f*$ is a homomorphism by calculating

$$
\begin{array}{ll}
 & h \cdot +\!\!+ / \\
= & \quad \{ \text{ given form for } h \text{ } \} \\
 & \oplus/ \cdot f* \cdot +\!\!+ / \\
= & \quad \{ \text{ map and reduce promotion } \} \\
 & \oplus/ \cdot (\oplus/ \cdot f*)* \\
= & \quad \{ \text{ hypothesis } \} \\
 & \oplus/ \cdot h*
\end{array}
$$

Examples of Homomorphisms

- $\#$: compute the length of a list.

$$\# = +/ \cdot K_1 *$$

- *reverse*: reverses the order of the elements in a list.

$$reverse = \tilde{+\!\!+} / \cdot [\cdot] *$$

Here, $x \tilde{\oplus} y = y \oplus x$.

- *sort*: reorders the elements of a list into ascending order.

$$sort = \curlyvee\!\!\curlyvee / \cdot [\cdot]*$$

Here, $\curlyvee\!\!\curlyvee$ (pronounced *merge*) is defined by the equations:

$$\begin{array}{lcll} x \curlyvee\!\!\curlyvee [\,] & = & x & \\ [\,] \curlyvee\!\!\curlyvee y & = & y & \\ ([a] +\!\!+ x) \curlyvee\!\!\curlyvee ([b] +\!\!+ y) & = & [a] +\!\!+ (x \curlyvee\!\!\curlyvee ([b] +\!\!+ y)), & \text{if } a \leq b \\ & = & [b] +\!\!+ (([a] +\!\!+ x) \curlyvee\!\!\curlyvee y), & \text{otherwise} \end{array}$$

- *all* p: returns True if every element of the input list satisfies the predicate p.

$$all\ p = \wedge/ \cdot p*$$

- *some* p: returns True if at least one element of the input list satisfies the predicate p.

$$some\ p = \vee/ \cdot p*$$

- *split*: splits a non-empty list into its last element and the remainder.

$$
\begin{aligned}
\mathit{split}\ [a] &= ([\,], a) \\
\mathit{split}\ (x \mathbin{+\!\!+} y) &= \mathit{split}\ x \oplus \mathit{split}\ y \\
&\ \text{where } (x, a) \oplus (y, b) = (x \mathbin{+\!\!+} [a] \mathbin{+\!\!+} y, b)
\end{aligned}
$$

Exercise: Let $\mathit{init} = \pi_1 \cdot \mathit{split}$ and $\mathit{last} = \pi_2 \cdot \mathit{split}$ where $\pi_1\ (a, b) = a$ and $\pi_2(a, b) = b$. Show that *init* is not a homomorphism, but *last* is.

- *tails*: returns all the tail (final) segments of a list.

$$tails = \oplus/ \cdot f*$$

where

$$\begin{array}{lcl} f\ a & = & [[], [a]] \\ xs \oplus ys & = & init\ xs ++ (last\ xs ++) * ys. \end{array}$$

All applied to

The operator ^{o} (pronounced *all applied to*) takes a sequence of functions and a value and returns the result of applying each function to the value.

$$[f_1, f_2, \ldots, f_n]^o a = [f_1\ a, f_2\ a, \ldots, f_n\ a]$$

Formally, $(^o\ a)$ is a homomorphism:

$$\begin{array}{lcl} [\,]^o\ a & = & [\,] \\ [f]^o\ a & = & [f\ a] \\ (fs +\!\!+\, gs)\ ^o\ a & = & (fs\ ^o\ a) +\!\!+\, (gs\ ^o\ a) \end{array}$$

Exercise: Show that $[\cdot] = [id]^o$.
Exercise: Show that we can redefine *tails* to be $tails = \oplus/ \cdot [[]^o, [id]^o]^o *$.

Conditional Expressions

The conditional notation

$$\begin{aligned} h\ x &= f\ x, \quad \text{if } p\ x \\ &= g\ x, \quad \text{otherwise} \end{aligned}$$

will be written by the McCarthy conditional form:

$$h = (p \rightarrow f, g)$$

Laws on Conditional Forms

$$\begin{aligned} h \cdot (p \rightarrow f, g) &= (p \rightarrow h \cdot f, h \cdot g) \\ (p \rightarrow f, g) \cdot h &= (p \cdot h \rightarrow f \cdot h, g \cdot h) \\ (p \rightarrow f, f) &= f \end{aligned}$$

Filter

The operator $\triangleleft$ (pronounced *filter*) takes a predicate p and a list x and returns the sublist of x consisting, in order, of all those elements of x that satisfy p.

$$p\triangleleft = +\!\!+ / \cdot (p \rightarrow [id]^o, [\,]^o)*$$

Exercise: Prove that the filter satisfies the *filter promotion* property:

$$(p\triangleleft) \cdot +\!\!+ / = +\!\!+ / \cdot (p\triangleleft)*$$

Exercise: Prove that the filter satisfies the *map-filter swap* property:

$$(p\triangleleft) \cdot f* = f * \cdot (p \cdot f)\triangleleft$$

Cross-product

$X_{\oplus}$ is a binary operator that takes two lists x and y and returns a list of values of the form $a \oplus b$ for all a in x and b in y.

$$[a, b] X_{\oplus} [c, d, e] = [a \oplus c, b \oplus c, a \oplus d, b \oplus d, a \oplus e, b \oplus e]$$

Formally, we define $X_{\oplus}$ by three equations:

$$\begin{array}{lcl} x X_{\oplus} [\,] & = & [\,] \\ x X_{\oplus} [a] & = & (\oplus a) * x \\ x X_{\oplus} (y \mathbin{+\!\!+} z) & = & (x X_{\oplus} y) \mathbin{+\!\!+} (x X_{\oplus} z) \end{array}$$

Thus $(x X_{\oplus})$ is a homomorphism.

Properties

$[\,]$ is the *zero element* of $X_\oplus$:

$$[\,]X_\oplus x \;=\; xX_\oplus[\,] \;=\; [\,]$$

We have *cross promotion* rules:

$$\begin{aligned} f**\cdot X_{+\!+}/ &= X_{+\!+}/\cdot f*** \\ \oplus/*\cdot X_{+\!+}/ &= X_\oplus/\cdot(X_\oplus/)* \end{aligned}$$

And, if $\otimes$ distributes through $\oplus$, then we have the following general promotion rule:

$$\oplus/\cdot X_\otimes/ = \otimes/\cdot(\oplus/)*$$

Example Uses of Cross-product

- *cp*: takes a list of lists and returns a list of lists of elements, one from each component.

$$cp : [[\alpha]] \rightarrow [[\alpha]]$$
$$cp\ [[a, b], [c], [d, e]] = [[a, c, d], [b, c, d], [a, c, e], [b, c, e]]$$

$$cp = X_{++} / \cdot ([id]^{o} *) *$$

- *subs*: computes all subsequences of a list.

$$subs : [\alpha] \to [[\alpha]]$$
$$subs\ [a, b, c] = [[], [a], [b], [a, b], [c], [a, c], [b, c], [a, b, c]]$$

$$subs = X_{++} / \cdot [[\,]^o, [id]^o]^o *$$

- $(all\ p)\triangleleft$:

$$(all\ p)\triangleleft = +\!\!+ / \cdot (all\ p \rightarrow [id]^o, []^o)*$$

Note that *all* can be eliminated with the following property.

$$all\ p \rightarrow [id]^o, []^o = X_{+\!\!+} / \cdot (p \rightarrow [[id]^o]^o, [\,]^o)*$$

Exercise: Compute the value of the expression $(all\ even) \triangleleft [[1,3],[2]]$.

Selection Operators

Suppose f is a numeric valued function. We want to define the operator $\uparrow_f$ by

$$\begin{array}{rcll} x \uparrow_f y & = & x, & f\ x \geq f\ y \\ & = & y, & \text{otherwise} \end{array}$$

Properties:

1. $\uparrow_f$ is *associative and idempotent*;
2. $\uparrow_f$ is *selective* in that

$$x \uparrow_f y = x \quad \text{or} \quad x \uparrow_f y = y$$

3. $\uparrow_f$ is *maximizing* in that

$$f(x \uparrow_f y) = f\ x \uparrow f\ y$$

An Example: $\uparrow_{\#}$

Distributivity of $\uparrow_{\#}$:

$$
\begin{aligned}
x +\!\!+ (y \uparrow_{\#} z) &= (x +\!\!+ y) \uparrow_{\#} (x +\!\!+ z) \\
(y \uparrow_{\#} z) +\!\!+ x &= (y +\!\!+ x) \uparrow_{\#} (y +\!\!+ z)
\end{aligned}
$$

That is,

$$
\begin{aligned}
(x +\!\!+\ ) \cdot \uparrow_{\#} / &= \uparrow_{\#} / \cdot (x +\!\!+\ )* \\
(+\!\!+\ x) \cdot \uparrow_{\#} / &= \uparrow_{\#} / \cdot (+\!\!+\ x)*
\end{aligned}
$$

We assume $\omega = \uparrow_{\#} / [\,]$.

A short calculation

$$
\begin{array}{ll}
& \uparrow_{\#} / \cdot (all\ p)\triangleleft \\
= & \quad \{ \text{ definition before } \} \\
& \uparrow_{\#} / \cdot +\!\!+ / \cdot (X_{+\!\!+} / \cdot (p \rightarrow [[id]^o]^o, [\,]^o)*)* \\
= & \quad \{ \text{ reduce promotion } \} \\
& \uparrow_{\#} / \cdot (\uparrow_{\#} / \cdot X_{+\!\!+} / \cdot (p \rightarrow [[id]^o]^o, [\,]^o)*)* \\
= & \quad \{ \text{ cross distributivity } \} \\
& \uparrow_{\#} / \cdot (+\!\!+ / \cdot \uparrow_{\#} / * \cdot (p \rightarrow [[id]^o]^o, [\,]^o)*)* \\
= & \quad \{ \text{ map distributivity } \} \\
& \uparrow_{\#} / \cdot (+\!\!+ / \cdot (\uparrow_{\#} / \cdot (p \rightarrow [[id]^o]^o, [\,]^o))*)* \\
= & \quad \{ \text{ conditionals } \} \\
& \uparrow_{\#} / \cdot (+\!\!+ / \cdot (p \rightarrow \uparrow_{\#} / \cdot [[id]^o]^o, \uparrow_{\#} / \cdot [\,]^o)*)* \\
= & \quad \{ \text{ empty and one-point rules } \} \\
& \uparrow_{\#} / \cdot (+\!\!+ / \cdot (p \rightarrow [id]^o, K_{\omega})*)*
\end{array}
$$

Solution to the Problem

Recall the problem of computing the longest segment of a list, all of whose elements satisfied some given property p.

$$
\begin{aligned}
& \uparrow_{\#} / \cdot (\mathit{all}\ p) \triangleleft \cdot \mathit{segs} \\
= \quad & \{\ \text{segment decomposition}\ \} \\
& \uparrow_{\#} / \cdot (\uparrow_{\#} / \cdot (\mathit{all}\ p) \triangleleft \cdot \mathit{tails}) * \cdot \mathit{inits} \\
= \quad & \{\ \text{result before}\ \} \\
& \uparrow_{\#} / \cdot (\uparrow_{\#} / \cdot (+\!\!+ / \cdot (p \rightarrow [\mathit{id}]^{o}, K_{\omega})*) * \cdot \mathit{tails}) * \cdot \mathit{inits} \\
= \quad & \{\ \text{Horner's rule with } x \odot a = (x +\!\!+ (p\ a \rightarrow [a], \omega) \uparrow_{\#} [\,]\ \} \\
& \uparrow_{\#} \cdot \odot \not\rightarrow_{[\,]} * \cdot \mathit{inits} \\
= \quad & \{\ \text{accumulation lemma}\ \} \\
& \uparrow_{\#} \cdot \odot /\!\!\!/\!\!\!\rightarrow_{[\,]}
\end{aligned}
$$

Exercise: Show that the definition of $\odot$ can be simplified to

$$x \odot a = p\ a \to x +\!\!+ [a], [].$$

Exercise: Show the final program is linear in the number of calculation of p.

Exercise: Code the final algorithm in Haskell.