合成型上の関数

胡 振江

東京大学 計数工学科

2008 年 11 月 10 日

対型

定義

型 (T_1, T_2) は第 1 要素が T_1 型の値で第 2 要素が T_2 型の値であるような値の対で構成される型である.

$$
\begin{array}{lcl}
(17.3, 3) & :: & (Float, Int) \\
(3, 6) & :: & (Int, Int) \\
(True, (+)) & :: & (Bool, Int \to Int \to Int)
\end{array}
$$

対型上の基本関数

- 構成子 $(,)$

$$\begin{array}{lcl} (,) & :: & a \to b \to (a, b) \\ (,)\ x\ y & = & (x, y) \end{array}$$

- 分離関数 fst, snd:

$$\begin{array}{lcl} fst & :: & (a, b) \to a \\ fst\ (x, y) & = & x \end{array}$$

$$\begin{array}{lcl} snd & :: & (a, b) \to b \\ snd\ (x, y) & = & y \end{array}$$

組型

定義

型 $(T_1, \ldots, T_n)$ は第 1 要素が T_1 型の値で第 n 要素が T_n 型の値であるような値の組で構成される型である.

$$
\begin{array}{lcl}
(17.3, 3, \mathit{True}) & :: & (\mathit{Float}, \mathit{Int}, \mathit{Bool}) \\
(3, 6, 7, 8, 9) & :: & (\mathit{Int}, \mathit{Int}, \mathit{Int}, \mathit{Int}, \mathit{Int})
\end{array}
$$

組型上の基本関数

- n 組を作る構成子 $(,\ldots,)$

$$(,\ldots,) :: a_1 \to \cdots \to a_n \to (a_1,\ldots,a_n)$$
$$(,\ldots,)\ x_1\ \cdots\ x_n = (x_1,\ldots,x_n)$$

- 分離関数 sel_i^n:

$$sel_i^n :: (a_1,\ldots,a_i,\ldots,a_n) \to a_i$$
$$sel_i^n\ (x_1,\ldots,x_i,\ldots,x_n) = x_i$$

組型上の関数の定義

問題

2 次方程式 $ax^2 + bx + c = 0$ の根を求めよ。

- 2 次方程式 $ax^2 + bx + c = 0$ を係数の組 (a, b, c) で表現する.
- 二つの根を対 (r_1, r_2) で表現する.

$$
\begin{aligned}
&roots \ :: \ (Float, Float, Float) \to (Float, Float) \\
&roots \ (a, b, c) \mid d \geq 0 \ = \ (r_1, r_2) \\
&\quad \textbf{where} \\
&\qquad r_1 = (-b + r)/(2 * a) \\
&\qquad r_2 = (-b - r)/(2 * a) \\
&\qquad r = sqrt \ d \\
&\qquad d = b^2 - 4 * a * c
\end{aligned}
$$

組型上の関数の定義

問題

有理数上の四則演算を定義せよ。

- 有理数 x/y を整数の対 (x, y) で表現する.
- 有理数の正規化：$norm\ (8, 6)\ \Rightarrow\ (4, 3)$

$$
\begin{aligned}
&norm :: (Int, Int) \to (Int, Int)\\
&norm\ (x, y) \mid y \neq 0\ =\ (div\ u\ d, div\ v\ d)\\
&\qquad \textbf{where}\\
&\qquad\qquad u = (sign\ y) * x\\
&\qquad\qquad v = abs\ y\\
&\qquad\qquad d = gcd\ (abs\ u)\ v
\end{aligned}
$$

有理数上の演算の定義

有理数の四則演算子の定義

$$
\begin{aligned}
&radd :: (Int, Int) \to (Int, Int) \to (Int, Int) \\
&radd\ (x, y)\ (u, v) \ = \ norm\ (x * v + u * y, y * v)
\end{aligned}
$$

$$
\begin{aligned}
&rsub :: (Int, Int) \to (Int, Int) \to (Int, Int) \\
&rsub\ (x, y)\ (u, v) \ = \ norm\ (x * v - u * y, y * v)
\end{aligned}
$$

$$
\begin{aligned}
&rmul :: (Int, Int) \to (Int, Int) \to (Int, Int) \\
&rmul\ (x, y)\ (u, v) \ = \ norm\ (x * u, y * v)
\end{aligned}
$$

$$
\begin{aligned}
&rdiv :: (Int, Int) \to (Int, Int) \to (Int, Int) \\
&rdiv\ (x, y)\ (u, v) \ = \ norm\ (x * v, y * u)
\end{aligned}
$$

有理数上の演算の定義

有理数の比較演算子の定義

$$
\begin{aligned}
&\mathit{requals} :: (\mathit{Int}, \mathit{Int}) \to (\mathit{Int}, \mathit{Int}) \to \mathit{Bool} \\
&\mathit{requals}\ (x, y)\ (u, v) \ = \ x * v == y * u
\end{aligned}
$$

$$
\begin{aligned}
&\mathit{rless} :: (\mathit{Int}, \mathit{Int}) \to (\mathit{Int}, \mathit{Int}) \to \mathit{Bool} \\
&\mathit{rless}\ (x, y)\ (u, v) \ = \ x * v < y * u
\end{aligned}
$$

$$
\begin{aligned}
&\mathit{rgreater} :: (\mathit{Int}, \mathit{Int}) \to (\mathit{Int}, \mathit{Int}) \to \mathit{Bool} \\
&\mathit{rgreater}\ (x, y)\ (u, v) \ = \ x * v > y * u
\end{aligned}
$$

有理数上の演算の定義

分数を表示する関数の定義

$$
\begin{aligned}
&\mathit{showrat} :: (\mathit{Int}, \mathit{Int}) \to \mathit{String} \\
&\mathit{showrat}\ (x, y) \ = \ \textbf{let}\ (u, v) = \mathit{norm}\ (x, y) \\
&\qquad\qquad\qquad\qquad \textbf{in if}\ v == 1\ \textbf{then}\ \mathit{show}\ u \\
&\qquad\qquad\qquad\qquad\quad \textbf{else}\ \mathit{show}\ u +\!\!+ \text{"/"} +\!\!+ \mathit{show}\ v
\end{aligned}
$$

関数型と関数上の関数

関数型 $(\rightarrow)$ はすべての関数の集まりである.

$$+,\ -,\ *,\ /,\ \mathit{square},\ (\wedge),\ \mathit{ord}, \ldots$$

関数はあらゆる型の値を引数にとりうるし，あらゆる種類の値を結果として返すことができる.

高階関数

引数として関数をとる、あるいは結果として関数を返す関数.

例：微分演算子

$$\frac{d}{dx} :: \text{関数} \rightarrow \text{導関数}$$

関数の合成

関数合成

$$(.) :: (\beta \to \gamma) \to (\alpha \to \beta) \to (\alpha \to \gamma)$$
$$(f\ .\ g)\ x = f\ (g\ x)$$

例：

$$quad :: Integer \to Integer$$
$$quad = square\ .\ square$$

性質

$$id\ .\ f = f\ .\ id = f$$
$$(f\ .\ g)\ .\ h = f\ .\ (g\ .\ h)$$

関数の外延性

関数の相等

$f == g$ iff 任意の x に対して、$f\ x == g\ x$ が成り立つ。

次の二つの関数が等しい。

$$
\begin{aligned}
&double, double' :: Integer \to Integer \\
&double\ x = x + x \\
&double'\ x = 2 * x
\end{aligned}
$$

逆関数

定義

単射関数 $f :: A \to B$ に対して，A の任意の値 x に対して，

$$g\ (f\ x) = x$$

となる g を f の逆関数といい，一般的に f^{-1} と表す．

例：関数

$$\begin{array}{lcl} f & :: & Int \to (Int, Int) \\ f\ x & = & (sign\ x, abs\ x) \end{array}$$

は単射であり，次の逆関数をもつ．

$$\begin{array}{lcl} f^{-1} & :: & (Int, int) \to Int \\ f^{-1}\ (s, a) & = & s * a \end{array}$$

正格関数と非正格関数

- 正格関数
  - 定義：$f \perp = \perp$ であるような関数 f を正格関数 (strict function) という.
  - 例：$square(1/0) = \perp$
- 非正格関数
  - 定義：正格でない関数
  - 例：次の定義について考えよう.

$$\begin{array}{lcl} three & :: & Int \to Int \\ three\ x & = & 3 \end{array}$$

このときに, $three\ (1/0) = 3$ である.

無名関数：λ関数

$$\lambda x.\, x + x$$

$$\lambda x.\, x + 1$$

$$\lambda f\ x.\, f\ (f\ x)$$
$$(= \lambda f.\, (\lambda x.\, f\ (f\ x)))$$

注：Haskell では、$\lambda x.\, x + x$ を `\x->x+x` のように書く。

リスト

リスト

リスト型 $[\alpha]$ は線形に順序のついた，型が α である値の集まりである.

$$
\begin{array}{lcl}
[1,2,3] & :: & [Int] \\
['h','e','l','l','o'] & :: & [Char] \\
[[1,2],[3]] & :: & [[Int]] \\
[(+),(-)] & :: & [Int \to Int \to Int] \\
[\,] & :: & [a] \\
[1,"fine\ day"] & & \text{リストではない！}
\end{array}
$$

数学の多くの分野で集合が重要であるのと同様に，関数プログラミングにおいてはリストが重要な役割を果たすのである.

リストの構成的定義

$$\textbf{data}\ [\alpha]\ =\ [\,]\mid \alpha : [\alpha]$$

1. 空リスト [] は $[\alpha]$ の要素である. $[\,] \in [\alpha]$.
2. $x \in [\alpha]$ ならば, $a \in \alpha$ に対して, $a : x \in [\alpha]$ である.
3. これら以外に $[\alpha]$ の要素ではない.

表記：

$$[a_0, a_1, \ldots, a_{n-1}] = a_0 : (a_1 : (\cdots (a_{n-1} : [\,])))$$

リスト上の基本関数

リスト構成子

[] : 空リストを作る構成子

$$[] :: [\alpha]$$

(:) : リストの先頭に要素を付け加える構成子

$$(:) :: \alpha \to [\alpha] \to [\alpha]$$

リスト上の基本関数

リストを分離する関数

$$
\begin{array}{lcl}
\mathit{head} & :: & [\alpha] \to \alpha \\
\mathit{head}\ (a : x) & = & a
\end{array}
$$

$$
\begin{array}{lcl}
\mathit{tail} & :: & [\alpha] \to [\alpha] \\
\mathit{tail}\ (a : x) & = & x
\end{array}
$$

空リストと判断する関数

$$
\begin{array}{lcl}
\mathit{null} & :: & [\alpha] \to \mathit{Bool} \\
\mathit{null}\ [\,] & = & \mathit{True} \\
\mathit{null}\ (a : x) & = & \mathit{False}
\end{array}
$$

リスト上の関数の定義：再帰関数

再帰的なデータを処理する関数は一般的に再帰的関数になる.

リストの長さを求める関数

$$length\ [1,2,4,3,3] \Rightarrow 5$$

$$\begin{array}{lcl} length & :: & [\alpha] \rightarrow Int \\ length\ [\,] & = & 0 \\ length\ (a : x) & = & 1 + length\ x \end{array}$$

リスト上の関数の定義：再帰関数

リストの要素の和を求める関数

$$sum\ [1,2,3,4,5] \Rightarrow 15$$

$$\begin{array}{lcl} sum & :: & Num\ \alpha \Rightarrow [\alpha] \rightarrow \alpha \\ sum\ [\,] & = & 0 \\ sum\ (a : x) & = & a + sum\ x \end{array}$$

リスト上の関数の定義：再帰関数

二つリストを連接する関数

$$[1,2,3] +\!\!+ [4,5,6,7] \Rightarrow [1,2,3,4,5,6,7]$$

$$\begin{array}{lcl} (+\!\!+) & :: & [\alpha] \to [\alpha] \to [\alpha] \\ [] +\!\!+ y & = & y \\ (a : x) +\!\!+ y & = & a : (x +\!\!+ y) \end{array}$$

リスト上の関数の定義：再帰関数

ネストリストを平坦化する関数

$$concat\ [[1,2,3],[4,5],[],[4,5,6,7]] \Rightarrow [1,2,3,4,5,4,5,6,7]$$

$$\begin{array}{lcl} concat & :: & [[\alpha]] \rightarrow [\alpha] \\ concat\ [\,] & = & [\,] \\ concat\ (x : xs) & = & x +\!\!+ concat\ xs \end{array}$$

リスト上の関数の定義：再帰関数

リストを逆順にする関数

$$reverse\ [1,2,3,4,5] \Rightarrow [5,4,3,2,1]$$

$$\begin{array}{lcl} reverse & :: & [\alpha] \to [\alpha] \\ reverse\ [\,] & = & [\,] \\ reverse\ (a : x) & = & reverse\ x +\!\!+ [a] \end{array}$$

リスト上の関数の定義：再帰関数

last 関数

$$last\ [1,2,3,4,5] \Rightarrow 5$$

解法１：

$$\begin{array}{lcl} last & :: & [\alpha] \to \alpha \\ last & = & head\ .\ reverse \end{array}$$

解法２：

$$\begin{array}{lcl} last\ [a] & = & a \\ last\ (a:x) & = & last\ x \end{array}$$

左の二つのパータンに overlap がある。次の関数 $last'$ は正しく動かない。

$$\begin{array}{lcl} last'\ (a:x) & = & last'\ x \\ last'\ [a] & = & a \end{array}$$

解法３：

$$\begin{array}{lcl} last\ [a] & = & a \\ last\ (a:b:x) & = & last\ (b:x) \end{array}$$

リスト上の関数の定義：再帰関数

リストの綴じ合わせ関数

$$zip\ [1,2,3,4]\ [11,12,13,14] \Rightarrow [(1,11),(2,12),(3,13),(4,14)]$$

$$\begin{array}{lcl} zip & :: & [\alpha] \to [\beta] \to [(\alpha,\beta)] \\ zip\ [\,]\ [\,] & = & [\,] \\ zip\ (a:x)\ (b:y) & = & (a,b) : zip\ x\ y \end{array}$$

リスト上の関数の定義：再帰関数

先頭部分の取り出し

$$take\ 3\ [1,2,3,4] \Rightarrow [1,2,3]$$
$$take\ 3\ [1,2] \Rightarrow [1,2]$$
$$take\ 3\ "functional" \Rightarrow "fun"$$

$$\begin{array}{lcl} take & :: & Int \rightarrow [\alpha] \rightarrow [\alpha] \\ take\ 0\ x & = & [\,] \\ take\ (n+1)\ [\,] & = & [\,] \\ take\ (n+1)\ (a:x) & = & a : take\ n\ x \end{array}$$

リスト上の関数の定義：再帰関数

先頭部分の取り出し

$$drop\ 3\ [1,2,3,4] \Rightarrow [4]$$
$$drop\ 5\ [1,2,3,4] \Rightarrow [\,]$$

$$\begin{array}{lcl} drop & :: & Int \to [\alpha] \to [\alpha] \\ drop\ 0\ x & = & x \\ drop\ (n+1)\ [\,] & = & [\,] \\ drop\ (n+1)\ (a:x) & = & drop\ n\ x \end{array}$$

リスト上の関数の定義：再帰関数

リストを分割する関数

$$splitAt\ 3\ [1,2,3,4] \Rightarrow ([1,2,3],[4])$$

解法１：

$$\begin{array}{lcl} splitAt & :: & Int \to [\alpha] \to ([\alpha],[\alpha]) \\ splitAt\ n\ x & = & (take\ n\ x, drop\ n\ x) \end{array}$$

解法２：

$$\begin{array}{lcl} splitAt\ 0\ x & = & ([\,],x) \\ splitAt\ (n+1)\ [\,] & = & ([\,],[\,]) \\ splitAt\ (n+1)\ (a:x) & = & (a:y,z) \\ & & \textbf{where}\ (y,z) = splitAt\ n\ x \end{array}$$

リスト上の関数の定義：再帰関数

リストのインデクス

$$[1,2,3,4] \;!!\; 2 \Rightarrow 3$$
$$[1,2,3,4] \;!!\; 0 \Rightarrow 1$$

$$\begin{array}{lcl} (!!) & :: & [\alpha] \to \mathit{Int} \to \alpha \\ (a:x) \;!!\; 0 & = & a \\ (a:x) \;!!\; (n+1) & = & x \;!!\; n \end{array}$$

リスト上の関数の定義：map と filter

リストの各要素に関数を適用する関数

$$map\ square\ [1,2,3,4] \Rightarrow [1,4,9,16]$$
$$map\ (<3)\ [1,2,3] \Rightarrow [True, True, False]$$
$$map\ nextLetter\ "HAL" \Rightarrow "IBM"$$

$$\begin{array}{lcl} map & :: & (\alpha \to \beta) \to [\alpha] \to [\beta] \\ map\ f\ [\,] & = & [\,] \\ map\ f\ (a : x) & = & f\ a : map\ f\ x \end{array}$$

リスト上の関数の定義：map と filter

リストの各要素に述語を適用し述語を満たさない要素を取り除く関数

$$filter\ even\ [1,2,3,4] \Rightarrow [2,4]$$
$$(sum\ .\ map\ square\ .\ filter\ even)[1..10] \Rightarrow 220$$

$$\begin{array}{lcl} filter & :: & (\alpha \to Bool) \to [\alpha] \to [\alpha] \\ filter\ p\ [\,] & = & [\,] \\ filter\ p\ (a : x) & = & \textbf{if}\ p\ a\ \textbf{then}\ a : filter\ p\ x\ \textbf{else}\ filter\ p\ x \end{array}$$

リスト上の関数の定義：map と filter

リスト内包表記 (list comprehension)

$$
\begin{aligned}
&[x * x \mid x \leftarrow [1..5],\ \mathit{odd}\ x] \\
&\quad \Rightarrow [1, 9, 25] \\
&[(a, b) \mid a \leftarrow [1..3], b \leftarrow [1..2]] \\
&\quad \Rightarrow [(1,1),(1,2),(2,1),(2,2),(3,1),(3,2)] \\
&[(i, j) \mid i \leftarrow [1..4], j \leftarrow [i+1..4]] \\
&\quad \Rightarrow [(1,2),(1,3),(1,4),(2,3),(2,4),(3,4)] \\
&[(i, j) \mid i \leftarrow [1..4], \mathit{even}\ i, j \leftarrow [i+1..4], \mathit{odd}\ j] \\
&\quad \Rightarrow [(2,3)]
\end{aligned}
$$

リスト上の関数の定義：map と filter

リスト内包表記から map, filter での表記への翻訳

$$
\begin{array}{lcl}
[a \mid a \leftarrow x] & = & x \\
[f\ a \mid a \leftarrow x] & = & map\ f\ x \\
[e \mid a \leftarrow x, p\ a, ...] & = & [e \mid a \leftarrow filter\ p\ x, ...] \\
[e \mid a \leftarrow x, b \leftarrow y, ...] & = & concat\ [[e \mid b \leftarrow y] \mid a \leftarrow x, ...]
\end{array}
$$

翻訳例

$$
\begin{array}{lcl}
 & & [x * x \mid x \leftarrow xs, even\ x] \\
 & = & [x * x \mid x \leftarrow filter\ even\ xs] \\
 & = & [square\ x | x \leftarrow filter\ even\ xs]\ \textbf{where}\ square\ x = x * x \\
 & = & map\ square\ (filter\ even\ xs)
\end{array}
$$

高階関数：fold 関数

右側畳み込み関数

$$
\begin{array}{lcl}
\mathit{foldr} & :: & (a \to b \to b) \to b \to [a] \to b \\
\mathit{foldr}\ (\oplus)\ e\ [\,] & = & e \\
\mathit{foldr}\ (\oplus)\ e\ (a : x) & = & a \oplus \mathit{foldr}\ (\oplus)\ e\ x
\end{array}
$$

例：

$$\mathit{foldr}\ (\oplus)\ e\ (a_1 : (a_1 : (a_3 : [\,]))) \Rightarrow a_1 \oplus (a_2 \oplus (a_3 \oplus e))$$

foldr はリスト上の一番自然で強力あな計算パータンを表現している。

高階関数：fold 関数

foldr の使用例：

$$
\begin{array}{lcl}
sum & :: & [Int] \to Int \\
sum & = & foldr\ (+)\ 0
\end{array}
$$

$$
\begin{array}{lcl}
produt & :: & [Int] \to Int \\
product & = & foldr\ (*)\ 1
\end{array}
$$

$$
\begin{array}{lcl}
and & :: & [Bool] \to Bool \\
and & = & foldr\ (\wedge)\ True
\end{array}
$$

$$
\begin{array}{lcl}
or & :: & [Bool] \to Bool \\
or & = & foldr\ (\vee)\ False
\end{array}
$$

リスト上の関数の定義：高階関数

foldr の使用例：

$$
\begin{array}{lcl}
\mathit{concat} & :: & [[\alpha]] \to [\alpha] \\
\mathit{concat} & = & \mathit{foldr}\ (+\!\!+)\ [\,] \\
 & & \\
\mathit{reverse} & :: & [\alpha] \to [\alpha] \\
\mathit{reverse} & = & \mathit{foldr}\ \mathit{postfix}\ [\,]\ \textbf{where}\ \mathit{postfix}\ a\ r = r +\!\!+ [a] \\
 & & \\
\mathit{map} & :: & (\alpha \to \beta) \to [\alpha] \to [\beta] \\
\mathit{map}\ f & = & \mathit{foldr}\ ((:)\ .\ f)\ [\,] \\
 & & \\
\mathit{filter} & :: & (\alpha \to \mathit{Bool}) \to [\alpha] \to [\alpha] \\
\mathit{filter}\ p & = & \mathit{foldr}\ (\lambda a\ r.\,\textbf{if}\ p\ a\ \textbf{then}\ a : r\ \textbf{else}\ r)\ [\,]
\end{array}
$$

リスト上の関数の定義：高階関数

左側畳み込み関数

$$
\begin{array}{lcl}
\mathit{foldl} & :: & (b \to a \to b) \to b \to [a] \to b \\
\mathit{foldl}\ (\oplus)\ e\ [\,] & = & e \\
\mathit{foldl}\ (\oplus)\ e\ (x +\!\!+ [a]) & = & (\mathit{foldl}\ (\oplus)\ e\ x) \oplus a
\end{array}
$$

例：

$$\mathit{foldl}\ (\oplus)\ e\ [a_1, a_1, a_3] \Rightarrow ((e \oplus a_1) \oplus a_2) \oplus a_3$$

練習問題

次の pack 関数を foldl で定義せよ.

$$\mathit{pack}\ [x_n, x_{n-1}, \ldots, x_0] = x_n * 10^n + x_{n-1} * 10^{n-1} + \cdots + x_0$$

練習問題

- 教科書の 2.4, 2.5, 2.6, 5.1, 5.2, 5.3 を復習すること.