平成18年度「プログラムの数理」期末試験

平成 18 年 2 月 5 日 8:30 – 10:00

工学部 6 号館 63 号室

以下の問いに答えよ．

1. 数列が非減少であるかどうかを判定する関数 $asc :: [Int] \to Bool$ を考える．例えば，$asc\ [1,2,2,3,4] = True$, $asc\ [1,2,3,2] = False$ である．

   (a) 関数 asc は次のように再帰的に定義することができる．

   $$\begin{array}{lcl} asc\ [\,] & = & True \\ asc\ [a] & = & True \\ asc\ (a : b : x) & = & a \leq b \wedge asc\ (b : x) \end{array}$$

   この定義に従って，式 $asc\ [1,2,2,3,4]$ を計算する簡約過程を示せ．

   (b) 関数 asc は以下のように別の方法で定義することもできる．

   $$\begin{array}{lcl} asc\ [\,] & = & True \\ asc\ [a] & = & True \\ asc\ (x +\!\!+ [a,b]) & = & E \end{array}$$

   を満たすような式 E を与えよ．

   (c) 関数 asc' を

   $$\begin{array}{l} asc'\ x = (sort\ x == x) \\ \quad \textbf{where} \\ \quad\quad sort = foldr\ insert\ [\,] \\ \quad\quad insert\ a\ [\,] = [a] \\ \quad\quad insert\ a\ (b : y) = \textbf{if}\ a \leq b\ \textbf{then}\ a : b : y\ \textbf{else}\ b : insert\ a\ y \end{array}$$

   と定義する．等式 $asc' = asc$ が成り立つことを帰納法で証明せよ．

2. 次の関数を考える．

   $$lat\ =\ \uparrow / \cdot \# * \cdot asc \triangleleft \cdot tails$$

   ただし，関数 asc は問 1 で定義したものとする．

   (a) 関数 lat の型を与えよ．

   (b) 関数 lat が何を計算するかを例を用いて説明せよ．

(c) 関数 lat は一つの準同型関数で表現できない，つまり，次の等式

$$lat\ (x +\!\!+ y) = lat\ x \oplus lat\ y$$

を満たす二項演算子 $\oplus$ が存在しない．これを証明せよ．

(d) 等式

$$lat\ =\ \uparrow / \cdot f * \cdot tails$$

を満たすような関数 f を導出せよ．

(e) 関数 lat が次のように定義することができることを証明せよ．

$$\begin{array}{l} lat\ =\ \pi_1 \cdot \odot \not\rightarrow_{(0,-\infty)} \\ \quad \textbf{where} \\ \quad\quad (e_1, e_2) \odot a = \textbf{if}\ e_2 \leq a\ \textbf{then}\ (e_1 + 1, a)\ \textbf{else}\ (1, a) \end{array}$$

3. (a) 次の分離定理を証明せよ．
[**分離定理**]: 関数 S と T を

$$\begin{array}{l} S = \oplus / \cdot f * \cdot segs \\ T = \oplus / \cdot f * \cdot tails \end{array}$$

と定義する．もし T が $T = h \cdot \odot \not\rightarrow_e$ のように表現できるならば,

$$S = \oplus / \cdot h * \cdot \odot \not\Rightarrow_e$$

が成り立つ．

(b) 分離定理と問 2 の結果を利用して，次の仕様

$$las\ =\ \uparrow / \cdot \# * \cdot asc \triangleleft \cdot segs$$

から線形時間のアルゴリズムを導出せよ．

(c) 上で導出されたアルゴリズムを Haskell で書け．

4. 構成的アルゴリズム論に基づくプログラミング手法のポイントを例を用いて説明せよ．

略解.

1. (a)

$$\begin{aligned} & asc\ [1,2,2,3,4] \\ = \ & 1 \leq 2 \wedge asc\ [2,2,3,4] \\ = \ & True \wedge asc\ [2,2,3,4] \\ = \ & asc\ [2,2,3,4] \\ = \ & \cdots \\ = \ & True \end{aligned}$$

(b)

$$asc\ (x +\!\!+ [a,b]) = asc\ (x +\!\!+ [a]) \wedge a \leq b$$

(c) 帰納法で証明する.

ベースケース.

$$\begin{aligned} & asc'\ [] \\ = \ & sort\ [] == [] \\ = \ & [] == [] \\ = \ & True \\ = \ & asc\ [] \end{aligned}$$

$$\begin{aligned} & asc'\ [a] \\ = \ & sort\ [a] == [a] \\ = \ & insert\ a\ [] == [a] \\ = \ & [a] == [a] \\ = \ & True \\ = \ & asc\ [a] \end{aligned}$$

帰納ケース. 仮定：$asc\ (b:x) = sort(b:x) == b:x$.

$$\begin{aligned} & asc'\ (a:b:x) \\ = \ & sort\ (a:b:x) == a:b:x \\ = \ & insert\ a\ (sort\ (b:x)) = a:b:x \end{aligned}$$

ケース１： $sort\ (b:x) == b:x$. 仮定により，$asc\ (b:x)$ が成立.

$$\begin{aligned} = \ & insert\ a\ (b:x) == a:b:x \\ = \ & a \leq b = \qquad\qquad a \leq b \wedge masc\ (b:x) \\ = \ & asc\ (a:b:x) \end{aligned}$$

ケース２：$sort\ (b:x) \neq b:x$. 仮定により，$asc\ (b:x) = False$ となる．従って，$asc\ (a:b:x) = False$ となる．ここで，$asc'\ (a:b:x) = False$ を証明すればよい. $sort\ (b:x) \neq b:x$ から $sort\ (a:b:x) \neq a:b:x$ を示せればよい.

2. (a)

$$lat :: [[Int]] \rightarrow Int$$

(b) 最長の上昇のテールセグメントの長さを求める関数.

(c) 仮に次の等式

$$lat\ (x +\!\!+ y) = lat\ x \oplus lat\ y$$

を満たす二項演算子 $\oplus$ が存在する.

$$\begin{array}{ll} & lat\ [1,4,2,4] \\ = & lat\ [1,4] \oplus lat\ [2,4] \\ = & 2 \oplus 2 \\ = & lat\ [1,2] \oplus lat\ [3,4] \\ = & lat\ [1,2,3,4] \end{array}$$

$lat\ [1,4,2,4] = 2$, $lat\ [1,2,3,4] = 4$ なので, 矛盾である.

(d)

$$\begin{array}{ll} & \uparrow / \cdot \# * \cdot asc \triangleleft \cdot tails \\ = & \uparrow / \cdot \# * +\!\!+ \ / \cdot \ asc' * \cdot tails \\ = & \uparrow / \cdot +\!\!+ \ / \cdot \# * * \cdot \ asc' * \cdot tails \\ = & \uparrow / \cdot \uparrow / * \cdot \# * * \cdot \ asc' * \cdot tails \\ = & \uparrow / \cdot (\uparrow / \cdot \# * \cdot asc') * \cdot tails \end{array}$$

ここでは, $asc' = asc \rightarrow [.]; K_{[]}$ と定義する.

従って, $f = \uparrow / \cdot \# * \cdot asc'$ となる.

(e) 帰納法で証明すればよい.

3. (a)

$$\begin{array}{ll} & S \\ = & \oplus / \cdot f * \cdot segs \\ = & \oplus / \cdot f * \cdot \ +\!\!+ / \cdot tails * \cdot inits \\ = & \oplus / \cdot +\!\!+ \ / \cdot f * * \cdot \ tails * \cdot inits \\ = & \oplus / \cdot \oplus / * \cdot f * * \cdot \ tails * \cdot inits \\ = & \oplus / \cdot (\oplus / \cdot f * \cdot tails) * \cdot inits \\ = & \oplus / \cdot (h \cdot \odot \not\rightarrow_e) * \cdot inits \\ = & \oplus / \cdot h * \cdot \odot \not\rightarrow_e * \cdot inits \\ = & \oplus / \cdot h * \cdot \odot \not\#_e \end{array}$$

(b) 代入すればすぐできる.

(c) reduce と scan を Haskell で書くことになる.