アルゴリズムとデータ構造入門

1.データによる抽象の構築

2.1 高階手続きによる抽象化

奥 乃 博
大学院情報学研究科知能情報学専攻
知能メディア講座 音声メディア分野
工学部情報学科計算機科学コース
http://winnie.kuis.kyoto-u.ac.jp/~okuno/Lecture/07/IntroAlgDs/
okuno@nue.org

TAの居室は10号館4階奥乃1研，2研
TAのページが運用中，質問箱もあります

1

補足

- TUT-Scheme (TUS) の時間測定
  - `(time <form>)`
  - `(time (factorial 1000))`
  - `(time (null? (factorial 10000)))`
- TUT-Scheme (TUS) の乱数 `(random)`はない．
  - http://winnie.kuis.kyoto-u.ac.jp/~okuno/Lecture/05/IntroAlgDs/random.lsp
- cygwin上のTUS: 改行が `nl` (¥n) でないといけない
  - で改行が "`cr nl`" か"`nl`"だけかをチェック．
    od -a ファイル | more
  - "`cr nl`" ならば，"`nl`" に変換
    tr '¥r' '' < ファイル > 新しいファイル名
  - scheme.lsp で"`#¥return`" を "`#¥newline`" と同じ扱いにするように指定

2

11月13日・本日のメニュー

データによる抽象化

- 1.3.4 Procedures as Returned Values
- 2 Building Abstractions with Data
- 2.1 Introduction to Data Abstraction
- 2.1.1 Example: Arithmetic Operations for Rational Numbers
- 2.1.2 Abstraction Barriers
- 2.1.3 What Is Meant by Data?
- 2.1.4 Extended Exercise: Interval Arithmetic

3

11月13日・本日のメニュー

データによる抽象化

- 1.3.4 Procedures as Returned Values
- **2 Building Abstractions with Data**
- **2.1 Introduction to Data Abstraction**
- 2.1.1 Example: Arithmetic Operations for Rational Numbers
- 2.1.2 Abstraction Barriers
- 2.1.3 What Is Meant by Data?
- 2.1.4 Extended Exercise: Interval

15

第2章 データによる抽象の構築

- 第1章は手続き抽象化
  - 基本手続き
  - 合成手続き・手続き抽象化
  - 例: Σ, Π, accumulate, filtered-accumulate
- 第2章はデータ抽象化
  - 基本データ構造(primitive data structure/object)
  - 合成データオブジェクト (compound data object)
- データ抽象化で手続きの意味(semantics)を拡張
  - 加算(+)で**どのようなデータ構造も扱える**
  - 基本手続き: 整数+整数、有理数+有理数、実数+実数
  - 合成手続き: 複素数+複素数、行列+行列

16

第2章 データ抽象化で学ぶこと

- 抽象化の壁(abstraction barrier)の構築
  - データ構造の実装を外部から隠蔽(blackbox)
- 閉包(closure)
  - 組み合わせを繰り返してもよい
- 従来型インタフェース(conventional interface)
  - Sequence を手続き間インタフェースとして使用
  - ベルトコンベア、生産ライン、UNIXのパイプ
- 記号式(symbolic expression)による表現
- 汎用演算(genetic operations)
- データ主導プログラミング(data-directed programming)

17

11月13日・本日のメニュー

データによる抽象化

- 1.3.4 Procedures as Returned Values
- 2 Building Abstractions with Data
- 2.1 Introduction to Data Abstraction
- **2.1.1 Example: Arithmetic Operations for Rational Numbers**
- 2.1.2 Abstraction Barriers
- 2.1.3 What Is Meant by Data?
- 2.1.4 Extended Exercise: Interval Arithmetic

18

2.1 データ抽象化(data abstraction)

- 抽象データの4つの基本操作

1. 構成子(constructor)

2. 選択子(selector)

3. 述語(predicate)

4. 入出力(input/ output)

19

2.1.0 Integers(整数)

- **構成子(constructor)**
  <n> ; <n> **integer**
- **選択子(selector)**
  <n> ; <n> **integer**
- **述語(predicate)**
  (integerp? <x>)
  (= <x> <y>)
- **入出力(input/output)**
  <n> ; <n> **integer**

20

2.1.1 Rational Numbers(有理数)

- 構成子(constructor)
  `(make-rat <n> <d>)`
  `<n>` numenator(分子),
  `<d>` denominator (分母)
- 選択子(selector)
  `(numer <x>)`
  `(denom <x>)`
  `<x>` rational number
- 述語(predicate)
  `(rational? <x>)`
  `(equal-rat? <x> <y>)`
- 入出力(input/output)
  `<n>/<d>`

21

2.1.1 Rational Numbers(有理数)

- 加算 (addition)
- 減算 (subtraction)
- 乗算 (multiplication)
- 除算 (division)
- 述語

$$\frac{n_1}{d_1}+\frac{n_2}{d_2}=\frac{n_1d_2+n_2d_1}{d_1d_2}$$

$$\frac{n_1}{d_1}-\frac{n_2}{d_2}=\frac{n_1d_2-n_2d_1}{d_1d_2}$$

$$\frac{n_1}{d_1}\times\frac{n_2}{d_2}=\frac{n_1n_2}{d_1d_2}$$

$$\frac{n_1}{d_1}\div\frac{n_2}{d_2}=\frac{n_1d_2}{d_1n_2}$$

$n_1d_2=n_2d_1 \Rightarrow \frac{n_1}{d_1}=\frac{n_2}{d_2}$

22

Rational Number Operations

$$\frac{n_1}{d_1}+\frac{n_2}{d_2}=\frac{n_1d_2+n_2d_1}{d_1d_2}$$

$$\frac{n_1}{d_1}-\frac{n_2}{d_2}=\frac{n_1d_2-n_2d_1}{d_1d_2}$$

```
(define (add-rat x y)
  (make-rat
```

```
(define (sub-rat x y)
  (make-rat
```

23

Rational Number Operations

$\frac{n_1}{d_1} \times \frac{n_2}{d_2} = \frac{n_1 n_2}{d_1 d_2}$ $\frac{n_1}{d_1} \div \frac{n_2}{d_2} = \frac{n_1 d_2}{d_1 n_2}$ $n_1 d_2 = n_2 d_1$ ⇨ $\frac{n_1}{d_1} = \frac{n_2}{d_2}$

```
(define (mul-rat x y)
  (make-rat

(define (div-rat x y)
  (make-rat

(define (equal-rat? x y)
```

25

Rational Number Representation

```
(define (make-rat n d) (cons n d))
```

n	d

ペア(pair)で表現

```
(define (numer x) (car x))

(define (denom x) (cdr x))

(define (print-rat x)
   (newline)
   (display (numer x))
   (display "/")
   (display (denom x))
   x )
```

27

Rational Number Reduction(既約化)

```
(define (make-rat n d) (cons n d))
```

この表現は曖昧: e.g., 2/3, 4/6, 6/9

```
(define (make-rat n d)
   (let ((g (gcd n d)))
      (cons (/ n g) (/ d g)) ))
```

既約化: *reducing rational numbers to the lowest terms*

28

いつの時点で簡略化するべきか？

```
(define (make-rat n d)
  (let ((g (gcd n d)))
    (cons (/ n g) (/ d g)) ))
```

両者の長所・短所は？

```
(define (make-rat n d) (cond n d))
(define (numer x)
  (let ((g (gcd (car x) (cdr x))))
    (/ (car x) g) ))
(define (denom x)
  (let ((g (gcd (car x) (cdr x))))
    (/ (cdr x) g) ))
```

この違いは他のプログラムに影響を与えるか？

11月13日・本日のメニュー

データによる抽象化

- 1.3.4 Procedures as Returned Values
- 2 Building Abstractions with Data
- 2.1 Introduction to Data Abstraction
- 2.1.1 Example: Arithmetic Operations for Rational Numbers
- **2.1.2 Abstraction Barriers**
- 2.1.3 What Is Meant by Data?
- 2.1.4 Extended Exercise: Interval Arithmetic

30

既約化を抽象化の壁から見ると

有理数を使ったプログラム

プログラム領域での有理数

`add-rat sub-rat mul-等`

分子と分母から構成される有理数

`make-rat numer denom`

ペアとして構成される有理数

`cons car cdr`

ペアの実装法

11月13日・本日のメニュー

データによる抽象化

- 2 Building Abstractions with Data
- 2.1 Introduction to Data Abstraction
- 2.1.1 Example: Arithmetic Operations for Rational Numbers
- 2.1.2 Abstraction Barriers
- **Intermission**
- 2.1.3 What Is Meant by Data?
- 2.1.4 Extended Exercise: Interval Arithmetic

32

What is this instrument?

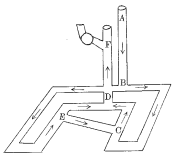
- 計算尺（slide rule, slipstick）
- 対数による積の計算
- 乗算→対数→加算
- 累乗→対数→乗算
- 2^{30} はいくら
- 2^{10} →対数→10log2 →3.01
- $2^{10} \fallingdotseq 10^3$ →1K
- $2^{30} \fallingdotseq 10^9$ →1G
- 音楽のピッチ
- 音の知覚

33

Slide Rule

鉱山用通気計算尺
G.R.R.I.Mine Ventilation Slide Rule

一資源試型計算尺使用法説明書



ヘンミ計算尺株式会社
東京都千代田区神田駿河台４－４
２６３１（代）
（６９０６Ｎ）

- アポロ13
- ミクロの決死隊
- タイタニック

34

Scale Label	Value Relative to C/D
A	X^2
B	X^2
C	X
CF	X x pi
CF/M	X x $\log_e 10$
CI	1 / X
CIF	1 / (pi x X)
D	X
DF	X x pi
DF/M	X x $\log_e 10$
DFM	X x $\log_e 10$
DI	1 / X
DIF	1 / (pi x X)
E	e^x
H	-
H1, H2	Square root of $(1+X^2)$
HC	-

K	X^3
KZ	X x 360
L	log X
Lg	log X
Ln	ln X
LL0	$e^{0.001x}$
LL1 or ZZ1	$e^{0.01x}$
LL2 or ZZ2	$e^{0.1x}$
LL3 or ZZ3	e^x
LL00 or LL/0	e^{-x}
LL01 or LL/1	$e^{-0.1x}$
LL02 or LL/2	$e^{-0.01x}$
LL03 or LL/3	$e^{-0.001x}$
M	log X
P	Square root of $(1-X^2)$
p%	-
P1, P2	Square root of $(1-X^2)$

R	1 / X
R1, R2	Square root of X
S	sin X
Sh1, Sh2	sinh X
Sq1, Sq2	Square root of X
SRT	sin X, tan X
ST	sin, tan X
T	tan X, cotan X
T1, T3	tan X, cotan X
T2	tan X, cotan X
Th	tanh X
V	Volts
W1, W2	Square root of X
Z	X
ZZ1, ZZ2, ZZ3	e^x

大きな数・小さな数

deca	da	$\times 10^1$
hecto	h	$\times 10^2$
kilo	K	$\times 10^3$
mega	M	$\times 10^6$
giga	G	$\times 10^9$
tera	T	$\times 10^{12}$
peta	P	$\times 10^{15}$
exa	E	$\times 10^{18}$
zetta	Z	$\times 10^{21}$
yotta	Y	$\times 10^{24}$

deci	d	$\times 10^{-1}$
centi	c	$\times 10^{-2}$
milli	m	$\times 10^{-3}$
micro	μ	$\times 10^{-6}$
nano	n	$\times 10^{-9}$
pico	p	$\times 10^{-12}$
femto	f	$\times 10^{-15}$
atto	a	$\times 10^{-18}$
zepto	z	$\times 10^{-21}$
yocto	y	$\times 10^{-24}$

36

10^1	ten *or* decad
10^2	hundred *or* hecatontad
10^3	thousand *or* chiliad
10^4	myriad
10^5	lac *or* lakh
10^6	million
10^7	crore
10^8	myriamyriad
10^9	milliard *or* billion
10^{12}	trillion
10^{15}	quadrillion
10^{18}	quintillion

10^{21}	sextillion
10^{24}	septillion
10^{33}	decillion
10^{63}	vigintillion
10^{303}	centillion
10^{100}	googol
10^{googol}	googolplex
10^N	*N* plex
10^{-N}	*N* minex

38

88plex	無量大数	20plex	垓	4minex	絲 (糸)
80plex	不可思議	16plex	京	5minex	忽
72plex	那由他	12plex	兆	6minex	微
64plex	阿僧祇	8plex	億	7minex	纖 (繊)
56plex	恒河砂	4plex	萬(万)	8minex	沙
48plex	極	3plex	千	9minex	塵
44plex	載	2plex	百	10minex	埃
40plex	正	1plex	十	11minex	渺
36plex	澗	0plex	一	12minex	漠
32plex	溝	1minex	分	13minex	模糊
28plex	穣	2minex	厘	14minex	逡巡
24plex	杼 (禾偏)	3minex	毫(毛)	15minex	須臾

39

1minex	分	13minex	模糊
2minex	厘	14minex	逡巡 シュンジュン
3minex	毫 (毛) モウ	15minex	須臾シュユ
4minex	絲 (糸) シ	16minex	瞬息シュンソク
5minex	忽 コツ	17minex	弾指ダンシ
6minex	微 ビ	18minex	殺那
7minex	纖 (繊)セン	19minex	六徳 リットク
8minex	沙シャ	20minex	虚
9minex	塵ジン	21minex	空
10minex	埃アイ	22minex	清
11minex	渺ビョウ	23minex	浄
12minex	漠バク		

40

11月13日・本日のメニュー

データによる抽象化

- 1.3.4 Procedures as Returned Values
- 2 Building Abstractions with Data
- 2.1 Introduction to Data Abstraction
- 2.1.1 Example: Arithmetic Operations for Rational Numbers
- 2.1.2 Abstraction Barriers
- **2.1.3 What Is Meant by Data?**
- 2.1.4 Extended Exercise: Interval Arithmetic

41

2.1.3 データって何？ ペア(対、pair)再考

(make-rat n d) の満足すべき条件は

$$\frac{\text{(numer x)}}{\text{(denom x)}} = \frac{n}{d}$$

1. cons, car, cdr を通常のセルで構築
2. 次の手続きで構築

```
(define (cons x y)
   (define (dispatch m)
      (cond ((= m 0) x)
            ((= m 1) y)
            (else (error "Argument not 0 or 1
                         -- CONS" m ))))
   dispatch )
(define (car z) (z 0))   (z 0) ⇒ car
(define (cdr z) (z 1))   (z 0) ⇒ cdr
```

11月13日・本日のメニュー

データによる抽象化

- 1.3.4 Procedures as Returned Values
- 2 Building Abstractions with Data
- 2.1 Introduction to Data Abstraction
- 2.1.1 Example: Arithmetic Operations for Rational Numbers
- 2.1.2 Abstraction Barriers
- 2.1.3 What Is Meant by Data?
- **2.1.4 Extended Exercise: Interval Arithmetic**

56

2.1.4 Interval Arithmetic (宿題)

- **Constructor**
  `(define (make-interval a b) (cons a b))`
- **Selectors 等**
  `(define (upper-bound x)`
  `(define (lower-bound x)`
  `(define (equal-interval? x y)`
  `(define (sub-interval x y)`
- **Interval arithmetic は単位系変換で重要。**
  - *1in*≒2.54*cm*, *1ft*≒30.48*cm*, *1yd*≒0.914*m*, *1mile*≒1.609*km*, *1nautical mile*≒1.852*km*, *1acre*≒4047m^2, *1 UKgal*≒4.54*l*, *1USgal*≒3.79*l*, *1bbl*≒159*l*, ***1πsec≒1 nano-century*** **(10^{-7} *year*)** , *1 light year*≒9.461×10^{12}km (**9.461T*km***)
  - *1oz*≒28.3*g*, *1lb*≒0.454*Kg*, *1ct*≒0.2*g*

57

2.1.4 Interval Arithmetic

```
(define (add-interval x y)
   (make-interval
     (+ (lower-bound x) (lower-bound y))
     (+ (upper-bound x) (upper-bound y)) ))
(define (mul-interval x y)
  (let ((p1 (* (lower-bound x) (lower-bound y)))
        (p2 (* (lower-bound x) (upper-bound y)))
        (p3 (* (upper-bound x) (lower-bound y)))
        (p4 (* (upper-bound x) (upper-bound y))))
    (make-interval (min p1 p2 p3 p4)
                   (max p1 p2 p3 p4))))
(define (div-interval x y)
   (mul-interval x
      (make-interval (/ 1.0 (upper-bound y))
                     (/ 1.0 (lower-bound y)) )))
```

宿題:11月20日正午締切

- 不動点の考え方を習得すること
- 宿題は、次の2題:
- Ex.2.7, 2.8, 2.9.

このDon't Panic の図も2006年度受講生の作品