第 2 章

2017 年 7 月 25 日

第二章では論文で使う概念や用語の定義やその説明などを行う．

1 可逆プログラミング言語 janus

可逆プログラミング言語 janus とは，C 言語に似た構文に加えて可逆性を保証するための構文要素をもつ．今回用いる janus は単射関数しか表すことができないという意味で可逆であることが証明されている．
例を用いて可逆プログラミング言語 janus について説明する．以下のプログラムは二項係数を計算するプログラムを janus で記述したものである．

```
1:   procedure binom(int n, int k, int r) // binomial coefficient
2:       if k >= 0 then
3:          r += 1
4:             local int i = 0
5:                from i = 0 loop i += 1
6:                   local int tmp = r*(n+1-i)/i
7:                      r <=> tmp
8:                   delocal int tmp = r*i/(n+1-i)
9:                until i = k
10:            delocal int i = k
11:     fi k >= 0
12:
13:  procedure main()
14:    int n = 6
15:    int k = 3
16:    int result
17:
18:    call binom(n,k,result)
```

janus の構文と意味論の概要を例を用いて示す．変数の基本型は，整数型，整数型の配列とスタックである．janus では非単射なものを扱うことができないため，単純な代入 $x := 1$ のように記述することができないが，かわりに 3,5 行目のように複合代入演算子 +=,-=,^= を用いて記述することができる．ただし，複合代入演算子式 $x+ = e$ の左オペランド x であらわれるオブジェクトが右オペランドの式 e にあらわれてはならず，左オペランドが添字演算子式の場合 $(x[a]+ = e)$ はその添字 a にもあらわれてはならない．この制約は $x+ = x$ のような非単射な計算を行う記述をできないようにするために必要である．

2 ランク計算

2.1 二分木

二分木とは，一つの節に対して多くても 2 つ以下の子を持つような木を二分木とよぶ．

2.2 自然順序

2.3 二分木の表現

2.4 二分木のランク

3 可逆シミュレーション

3.1 単射化

単射化を説明するうえでまずは単射や写像について説明する．写像とは二つの集合 A と B があり，A の各要素 a に対して B の要素 b が 1 つ対応しているとき，この対応を A から B への写像といい，対応する要素を $b = f(a)$ のように表す．また，f が集合 A から B への写像であることを $f : A \to B$ と表す．集合 B が実数や複素数の集合のときは，写像のことを関数と呼ぶ．単射とは，集合から他の集合の中への 1 対 1 の写像のことである．すなわち，集合 A から集合 B への写像 f が単射であるとは，$a, b \in A$ ，$a \neq b$ なら，$f(a) \neq f(b)$ であること，あるいは対偶をとって，$f(a) = f(b)$ なら $a = b$ であることを意味する．単射であるということは，1 対 1 の写像なので可逆であるといえる．単射なものと単射ではないものの例を挙げると，まず集合 A と集合 B の要素をそれぞれ $a1, a2 \in A$，$b1, b2 \in B$ として，写像 $f1, f2$ を次の規則で用意したとき，$f1(a1) = b1 \cap f1(a2) = b2$，$f2(a1) = b1 \cap f2(a2) = b1$ 写像 $f1$ は単射で写像 $f2$ は単射ではない．単射化とは単射もしくは単射ではない関数を拡張して単射な関数にすることである．記号を用いて単射化を定義する．任意の関数 f：$A \to B$ に対して，ある C が存在して任意の x $\in A$ に対して $fst(f'(x)) = f(x)$ となる単射関数 $f' : A \to B \times C$ が存在する．fst は $fst(x, y) = x$ となる関数である．必ずしも単射にならない関数 f から上記の条件を満たす単射関数 f' を作ることを単射化という．このときの C の要素は，元の関数 f の出力にはでないことからゴミ出力という．また，C がスタックの場合は C に属するスタックをゴミスタックと呼ぶ.

3.2 可逆化

可逆とは，ある状態から違う状態に遷移したとき遷移した状態から元の状態に戻ることができることである．可逆化とは，非可逆なものを可逆なものに変えることである．可逆化を行う方法として bennett による一般解法がある．この一般解法とは，非可逆なアルゴリズムについてもゴミ情報となる入力を特定するための情報を出力に追加することで可逆にすることができるという方法である.

3.3 埋め込み

埋め込みとは，非可逆なプログラムで代入などによって失われるデータや制御情報をすべて記憶するようにプログラムを変換することで可逆なプログラムに作り替える方法である．埋め込みによる可逆シミュレーションの実行には，時間計算量，空間計算量，消去情報量，および実時間性などの観点でトレードオフをもつ手法が提案されている.

3.4 シミュレーション