プログラミング言語論

第12回

(11回は欠番)

プログラムの意味論と検証（1）

ホーア論理

担当：犬塚

今日の講義

- プログラム検証の1つの枠組みであるホーア論理の概要を見る。
  - ホーア論理が扱う対象：表明付きプログラム
  - 正当性の考え方
  - ホーア論理の公理系
  - ホーア論理を用いた証明

プログラムの意味論と検証

次の2つは、密接に関連する。

- プログラムは何を表現しているのか。
- プログラムは仕様通りに動作するか。

- プログラムの正しさについて考える枠組み（＝意味論）はいくつかの考え方がある。
  - 操作的意味論
  - 公理的意味論
  - 表示的意味論
- 公理的意味論の考え方に従うホーア論理を紹介する。

ホーア論理 Hoare Logic

- 英国の計算機学者Hoareによって提案された、プログラムの正しさを議論するための論理。
- プログラムを、それが何をするものであるのかを示す表明(アサーション)つきで表わす。
- ＜A＞ P ＜B＞

  ＝「Aが成立っているとき、プログラムPを実行すると必ず停止し、そのときBが成立つ。」
- プログラムの正しさを検証するには、プログラムの仕様(specification)が必要。上のA、Bが仕様を規定する。

アサーション

- 表明付きプログラム

  ＜A＞ P ＜B＞

  「Aが成立っているとき、プログラムPを実行すると必ず停止し、そのときBが成立つ。」
- A, Bは条件、命題。プログラム検証の文脈では、表明（アサーション；assertion）という。
- Aを前条件または前件（pre-condition）、Bを後条件または後件（post-condition）という。

例

＜整数x,yの少なくとも1つはゼロでない＞

```
begin
a:=x; b:=y;
while b<>0 do
  begin
    a:=a mod b;
    c:=a; a:=b; b:=c
  end
od;
if a<0 then a:=-a fi
end
```

＜aはxとyの最大公約数である＞

アサーションとコメント

- アサーションはプログラムの働きを示したコメントとみなすことができる。
- 実際、アサーションに相当する文をコメントに入れるのは自然であり、ふさわしい。

- C言語のassert.hでは、アサーションをプログラム中に書くと、実行時これが成立っているか否か、チェックする。
- 表明付きプログラム

  ≒ きちんとコメントをつけたプログラム

停止性と正当性

〈Ａ〉Ｐ〈Ｂ〉

- Aが成立っているときPを実行するといつも停止する。
  ＝停止性(termination)
- Pが停止したときBが成立つ。
  ＝正当性(correctness)

正当性のみを表わす表明付きプログラムを

｛Ａ｝Ｐ｛Ｂ｝

とかく。

停止性と正当性

- 単に正当性といって、正当性＋停止性を意味する場合がある。
- しかし、これと正当性のみ成立つ場合を次のとおり区別する。

部分正当性（partial correctness）

- 正当性のみ成立つこと。
  即ち｛Ａ｝Ｐ｛Ｂ｝を主張すること。

全正当性（total correctness）または単に正当性

- 正当性と停止性が成立つ。
  即ち〈Ａ〉Ｐ〈Ｂ〉を主張すること。
- 全正当性＝部分正当性＋停止性

whileプログラム

- プログラムの理論研究では、プログラムの小さなクラスを研究対象とすることが多い。
- whileプログラムは代表例。
- whileプログラム：次の4形式で書けるプログラム

■ 代入文	変数 := 式;
■ 複合文（逐次）	begin 文1;･･･;文n end
■ while文	while 条件 do 文 od
■ if文	if 条件 then 文 else 文 fi

合成原理と意味

- 言語では合成原理(composition principle)は最重要原理。
  - 数式: e1, e2が数式のとき、(e1+e2), (e1*e2),...は数式。
  - 論理式: A1, A2が式のとき、(A1∧A2), (A1→A2),...は論理式。
  - whileプログラム: P1,P2がプログラムのとき、begin P1;P2 end はプログラム。

  ...

単に構文的に合成できるだけでなく、部分式の意味と、それをつなぐ結合子の意味で、全体の意味が再帰的に決まる。

表明付きプログラムの証明

- 部分正当性に注目して、証明について考える。
- 停止性はここでは考えない。

- プログラム検証とは、

  $$\{ A \}\ P\ \{ B \}$$

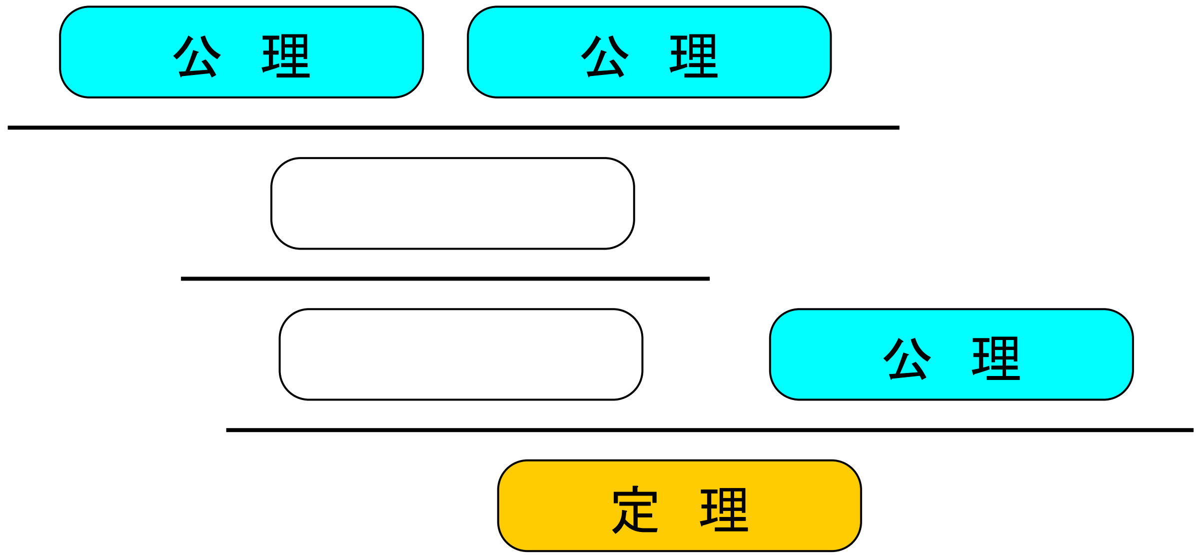
  が成立つとき、そのときに限って｛ A ｝ P ｛ B ｝を形式的に導く証明を与えることである。

論理のときと同様に次の形式を用いる。

- $\models \{ A \}\ P\ \{ B \}$; この表明付きプログラムが正しい
- $\vdash \{ A \}\ P\ \{ B \}$; この表明付きプログラムを証明できる

形式体系の証明論

- 公理系＝公理＋推論規則



- 公理に推論規則を繰り返し適用して導かれる式を定理といい、⊢ 定　理 と書く。

while プログラムの公理系

公理

A_{as} {A[t/x]} x:=t {A} ; Aは任意の論理式

A_{sk} {A} skip {A} ; skipは何もしないプログラム

推論規則

R_{if}

$$\frac{\{C \wedge A\}\ P\ \{B\} \qquad \{\neg C \wedge A\}\ Q\ \{B\}}{\{A\}\ \text{if } C \text{ then } P \text{ else } Q \text{ fi}\ \{B\}}$$

while プログラムの公理系(つづき)

推論規則

R_{wh}

$$\frac{\{C \wedge A\}\ P\ \{A\}}{\{A\}\ \mathrm{while}\ C\ \mathrm{do}\ P\ \mathrm{od}\ \{\neg C \wedge A\}}$$

R_{cp}

$$\frac{\{A\}P_1\{S_1\}\quad \{S_1\}P_2\{S_2\}\cdots\{S_{n-1}\}P_n\{B\}}{\{A\}\ \mathrm{begin}\ P_1;\ P_2;\ \cdots\ ;\ P_n\ \mathrm{end}\ \{B\}}$$

R_{cs}

$$\frac{\{B\}\ P\ \{C\}}{\{A\}\ P\ \{D\}}$$

ただし、

$S \vdash A \rightarrow B$ かつ $S \vdash C \rightarrow D$

Sはプログラム中の演算などについての非論理的公理。

whileの条件Aは、ループ不変条件(loop invariant)

例

- A_{as}より
  $\{a=10 \wedge a+a=20\}$ $b:=a+a$ $\{a=10 \wedge b=20\}$ (*)

- 同様にA_{as}より
  $\{b=20\}$ $c:=b$ $\{c=20\}$

- S ├─ $a=10 \wedge b=20 \rightarrow b=20$なので(*)にRcsを適用すると

$$\frac{\{a=10 \wedge a+a=20\}\ b:=a+a\ \{a=10 \wedge b=20\}}{\{a=10 \wedge a+a=20\}\ b:=a+a\ \{b=20\}}\ R_{cs}$$

- よってRcpを用いると、

$$\frac{\{a=10 \wedge a+a=20\}b:=a+a\{b=20\} \quad \{b=20\}c:=b\{c=20\}}{\{a=10 \wedge a+a=20\}\ \text{begin } b:=a+a;c:=b \text{ end}\{c=20\}}\ R_{cp}$$

例

次の表明付きプログラムを証明する

{} if a>b then skip else a:=b fi {a=max(a,b)}

$$\dfrac{\dfrac{\dfrac{}{\{a>b\}\ \mathrm{skip}\ \{a>b\}}A_{sk}}{\{a>b\}\ \mathrm{skip}\ \{a=\max(a,b)\}}R_{cs} \qquad \dfrac{\dfrac{}{\{\neg b>b\}\ a:=b\ \{\neg b>a\}}A_{as}}{\{\neg a>b\}\ a:=b\ \{a=\max(a,b)\}}R_{cs}}{\{\}\ \mathrm{if}\ a>b\ \mathrm{then}\ \mathrm{skip}\ \mathrm{else}\ a:=b\ \mathrm{fi}\ \{a=\max(a,b)\}}R_{if}$$

よって

$\vdash$ {} if a>b then skip else a:=b fi {a=max(a,b)}

健全性と完全性

- 次の2つを満たす公理系を与えることが重要である。
  - $\vdash$ {A}P{B} ならば $\models$ {A}P{B} ； 健全性
  - $\models$ {A}P{B} ならば $\vdash$ {A}P{B} ； 完全性
- 命題論理／述語論理の体系と同様、健全性は公理、推論規則の性質から容易に示される。
- 完全性の証明はいくらかのテクニックが必要。
- 前に示した、whileプログラムの公理系は完全性、健全性を満たす。

全正当性

- 全正当性の証明には、そのための公理系を用いる。
  たとえば、whileではおおよそ次の形式。

$$\mathrm{TR}_{wh}\quad \frac{\langle \mathrm{bound}=n \wedge C \wedge A\rangle\, P\, \langle \mathrm{bound}<n \wedge A\rangle}{\langle A\rangle\ \mathbf{while}\ C\ \mathbf{do}\ P\ \mathbf{od}\ \langle \neg C \wedge A\rangle}$$

ある変数　boundが、Pを一回行う毎に、確実に小さくなる
ことをいえるなら、停止性を含めて正当性を示せる。

ホーア論理の能力

- ホーア論理について凡そ次のことが分かっている。
  - whileプログラムについては、完全性を持つホーア論理の公理系を与えることができる。
  - whileプログラム以外の多くのプログラムクラスについても、完全性・健全性をもつ公理系を与えることができる。

しかし

  - 手続きプログラムには、完全性をもつホーア論理が存在しないことが分かっている。

手続きプログラム：手続き定義＋手続きを引数で渡す＋再帰呼出し＋静的スコープ＋大域変数

まとめ

- ホーア論理は、プログラムの意味を表明つきプログラムの形式で与える。
- その正しさを証明するための形式的体系＝公理系である。
- 表明付きプログラムの正しさは、 部分正当性と、全正当性があり、記述法で区別する。
- それぞれの正当性を示すための公理系がある。
- whileプログラムやその他の広い範囲のプログラムのクラスに、完全・健全な体系を与えられる。
- しかし、実際のプログラムの検証に使えるほど実際的な利用には耐えない。
- 理論的考察、新しいプログラミングに関する概念を生む、背景としての影響力が大きい。