

プログラミング意味論

国立情報学研究所

佐藤一郎

E-mail: ichiro@nii.ac.jp

Ichiro Satoh

授業概要

- 計算モデルとは
- 操作的意味論
- プロセス代数(Process Calculi)
- 様相論理
- モデルチェック
- 関数型計算モデル
- 型理論

Ichiro Satoh

プログラミング言語

構文(syntax)

- プログラミング言語の文法、表現

意味論(semantics)

- プログラムがどのような動作・結果を持つかを定式化

語用論(pragmatics)

- プログラムの書き方

Ichiro Satoh

プログラミング言語の意味

プログラミング言語の意味の定義方法

- 計算機のハードウェア動作として理解する
- 日本語や英語などの自然言語により意味を記述する
- 既知のプログラミング言語によって類似の文・式の意味を記述する

厳密で矛盾なく意味を定義するには

- 形式的意味論(formal semantics)
- 計算機とは独立して厳密に規定できること
- 定義に曖昧さが無いこと
- プログラムの推論・解析に理論的基礎を提供すること

Ichiro Satoh

意味論

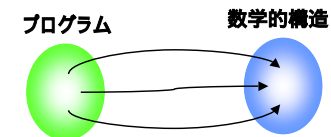
プログラミング言語意味論

- 操作意味論(Operational Semantics)
- 表示意味論(Denotational Semantics)
- 公理意味論(Axiomatic Semantics)

Ichiro Satoh

表示の意味論

集合や関数などの数学的概念を用いて構文上の対象の意味を記述



Ichiro Satoh

表示意味論

構文構成要素ごとに意味を定義

```
S ::= if E then S else S
    | while E do S od
    | S;S
    | E:=E
```

$\{\} c: M \quad (M \ M)$ ここでcは環境(変数の中身を定義)
 $(\text{if } E \text{ then } S1 \text{ else } S2) c = (S1) c \text{ (} (E) c = \text{true} \text{ のとき)}$
 $= (S2) c \text{ (} (E) c = \text{false} \text{ のとき)}$
 $(\text{while } E \text{ do } S \text{ od}) c = (\text{while } E \text{ do } S \text{ od}) c$
 $= (\{S\} c) \text{ (} (E) c = \text{true} \text{ のとき)}$
 $= \{\}$
 $\{S1;S2\} c = \{S2\} c \text{ (} \{S1\} c)$
 $\{E1:=E2\} c = c[(E2) c / (E1) c]$

Ichiro Satoh

公理的意味論

プログラミング言語の意味を公理と推論規則によって定めようとするもの

- Hoareモデル
- 時相論理による意味定義他

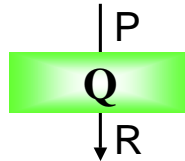
Ichiro Satoh

Hoare論理

C.A.R. Hoare
プログラムの事前条件と事後条件を論理式により定式化

$P\{Q\}R$

プログラムQを実行直前に論理式Pが成立して、
もしQの実行が終了すれば、その直後に論理式Rが成立する



Ichiro Satoh

Hoare論理

Hoare論理によるプログラミング意味論

$e ::= \text{null} \mid v = e \mid e_1 ; e_2 \mid \text{if}(e) e_1 \text{ else } e_2 \mid \text{while}(e) e$

$$\text{null} \frac{P \supset Q}{P \{ \text{null} \} Q}$$

$$\text{assignment} \frac{P \supset Q [e/x]}{P \{ x = e \} Q}$$

$$\text{sequence} \frac{P \{ e_1 \} R \quad R \{ e_2 \} Q}{P \{ e_1 ; e_2 \} Q}$$

$$\text{if-statement} \frac{P \wedge e \{ e_1 \} Q \quad P \wedge \neg e \{ e_2 \} Q}{P \{ \text{if}(e) e_1 \text{ else } e_2 \} Q}$$

$$\text{while-statement} \frac{P \supset L \quad L \wedge e \{ e \} L \quad L \wedge \neg e \supset Q}{P \{ \text{while}(e) e \} Q}$$

Ichiro Satoh

操作的意味論

抽象的な計算機を定義し、プログラミング言語の意味を抽象的計算機の動作(状態遷移)として記述する

抽象機械の例

- 有限状態機械(オートマトン)
- チューリングマシン
- ランダムアクセス機械
- ラムダ計算

プログラミング言語の意味定義には不適當

- 構造操作的意味論 (Plotkin)

Ichiro Satoh

操作的意味論

数式の操作的意味

$e ::= m \mid v \mid (e + e') \mid (e - e') \mid (e \times e')$

$$\frac{\langle e_0, \sigma \rangle \rightarrow \langle e'_0, \sigma \rangle}{\langle e_0 + e_1, \sigma \rangle \rightarrow \langle e'_0 + e_1, \sigma \rangle}$$

$$\frac{\langle e_1, \sigma \rangle \rightarrow \langle e'_1, \sigma \rangle}{\langle m + e_1, \sigma \rangle \rightarrow \langle m + e'_1, \sigma \rangle}$$

$$\langle m + m', \sigma \rangle \rightarrow \langle n, \sigma \rangle \quad (\text{where } n = m + m')$$

Ichiro Satoh

▶ 操作的意味論

簡易言語の意味

$$\langle \text{nil}, \sigma \rangle \rightarrow \sigma$$

$$\frac{\langle c_0, \sigma \rangle \rightarrow \langle c'_0, \sigma' \rangle}{\langle c_0; c_1, \sigma \rangle \rightarrow \langle c'_0; c_1, \sigma' \rangle} \quad \frac{\langle e, \sigma \rangle \rightarrow^* \langle m, \sigma \rangle}{\langle v := e, \sigma \rangle \rightarrow \sigma[m/v]}$$

$$\frac{\langle b, \sigma \rangle \rightarrow^* \langle \text{tt}, \sigma \rangle}{\langle \text{if } b \text{ then } c \text{ else } c', \sigma \rangle \rightarrow \langle c, \sigma \rangle}$$

$$\frac{\langle b, \sigma \rangle \rightarrow^* \langle \text{tt}, \sigma \rangle}{\langle \text{while } b \text{ do } c, \sigma \rangle \rightarrow \langle c; \text{ while } b \text{ do } c, \sigma \rangle}$$

$$\frac{\langle b, \sigma \rangle \rightarrow^* \langle \text{ff}, \sigma \rangle}{\langle \text{while } b \text{ do } c, \sigma \rangle \rightarrow \sigma}$$

Ichiro Satoh