メソッドスキーマの型整合性の十分条件

1B－4

百々 浩樹† 石原 靖哲‡ 関 浩之‡† 嵩 忠雄†‡
†奈良先端科学技術大学院大学 情報科学研究科 ‡大阪大学 基礎工学部

1 はじめに

オブジェクト指向言語では, 同一のメソッド名に引数の型に応じて異なる定義を与え, 実行時に, 型継承に従って実引数(メッセージ)に結合すべきメソッドを決定することにより, プログラムの記述の簡潔性と柔軟性を高めている. 実引数に結合すべきメソッド(resolutionと呼ぶ)が一意に定まる場合, そのプログラムは, 型整合性をもつという. 従って, プログラムが型整合性をもつことを, コンパイル時に保証できることが望ましい.

Abiteboulらは, メソッド定義の枠組としてメソッドスキーマとその型整合性を定義し, 次の結果を得た[1].

1. 与えられたメソッドスキーマが型整合性をもつかどうかは, 一般の場合には決定不能である.
2. (a) すべてのメソッドが1引数の場合, および (b) すべてのメソッド定義が非再帰的である場合, 型整合性をもつかどうかは決定可能である.

なお, 一般の場合に対しては, [1] では発見的手法が提案されているのみである. 本稿では, 一般の場合に発見的な手法によらない型整合性の十分条件を考察し, その十分条件を判定するアルゴリズムを提案した. また, すべてのメソッドが1引数の場合, 本アルゴリズムは型整合性の必要十分条件の判定を行うことを示した. この意味で, 本研究の結果は上記の結果2(a)を含んでいる.

2 メソッドスキーマ[1]

2.1 構文

変数の集合 X と関数記号の集合 F から構成される項集合を $T_F(X)$ と書く.

定義 1: メソッドスキーマは次のような5字組 $(C, \leq, M, \Sigma_b, \Sigma_u)$ である.

1. C はクラス名の有限集合, $\leq$ は C 上の半順序. ただし, 任意の $c \in C$ について, 「$c \leq c_1$ かつ $c \leq c_2$」ならば「$c_1 \leq c_2$ または $c_2 \leq c_1$」と仮定する. $c' \leq c$ は, c' が c の下位クラスであることを表す.
2. M は, $n\ (\geq 0)$ 引数のメソッド名の互いに素な有限集合 M_n からなる族. ただし, M_n は, 基底メソッド名の集合 $M_{n,b}$ とユーザメソッド名の集合 $M_{n,u}$ に分割される.
3. Σ_b は基底メソッド宣言の集合. ここで, クラス $c_1, \ldots, c_n \in C$ におけるメソッド $m \in M_{n,b}$ の基底メソッド宣言とは, 次の式である: $(m, (c_1, \ldots, c_n \rightarrow c_0))$, ただし, $c_0 \in C$.
4. Σ_u はユーザメソッド定義の集合. ここで, クラス $c_1, \ldots, c_n \in C$ におけるメソッド $m \in M_{n,u}$ のユーザメソッド定義とは, 次の式である: $(m, (c_1, \ldots, c_n), t)$, ただし, $t \in T_M(\{x_1, \cdots, x_n\})$. □

定義 2: $(C, \leq, M, \Sigma_b, \Sigma_u)$ をメソッドスキーマとし, $m \in M_{n,b}$, $c_i \in C\ (1 \leq i \leq n)$ とする. $R = \{(m, (c'_1, \ldots, c'_n \rightarrow c'_0)) \in \Sigma_b \mid$ 各 $i\ (1 \leq i \leq n)$ について, $c_i \leq c'_i\}$ とおく. 次の条件を満たす $r = (m, (c'_1, \ldots, c'_n \rightarrow c'_0)) \in R$ が存在するとき, r をクラス $c_1, \ldots, c_n$ における m の **resolution** と呼ぶ.

＜条件＞任意の $(m, (c''_1, \ldots, c''_n \rightarrow c''_0)) \in R$ に対し, $c'_i \leq c''_i\ (1 \leq i \leq n)$ である.

$m \in M_{n,u}$ の場合も同様に定義する. □

2.2 意味

定義 3: メソッドスキーマ $(C, \leq, M, \Sigma_b, \Sigma_u)$ の**解釈**とは, 2字組 $I = (\nu, \mu)$ である. ここで,

- ν は, 各 $c \in C$ に互いに素なオブジェクトの集合 $\nu(c)$ を割り当てる. $O = \bigcup_{c \in C} \nu(c)$ とおく.
- 各 $m \in M_{n,b}$ に対し, $\mu(m)$ は O^n から O への部分関数である. ただし, m の $c_1, \ldots, c_n$ における resolution が $(m, (c'_1, \ldots, c'_n \rightarrow c))$ のとき, $\mu(m)|_{\nu(c_1) \times \ldots \times \nu(c_n)}$ は, $\bigcup_{c' \leq c} \nu(c')$ への全域関数である. □

項 t において, 変数 $x_i\ (1 \leq i \leq n)$ を項 t_i で置き換えた項を $t[t_1/x_1, \cdots, t_n/x_n]$ と書く. また, 項 t の出現[2] v における部分項を項 t' に置き換えた項を $t[v \leftarrow t']$ と表す. $T_M(O)$ に属する項をインスタンス項という.

I をメソッドスキーマ S の一つの解釈とする. 最左最内戦略[2]による書換え関係 $\underset{I}{\rightarrow}$ を以下のように定義する.

定義 4: t をインスタンス項, v を t の最左最内リデックスとし, v における t の部分項を $m(o_1, \cdots, o_n)$ とする. ここで, $o_i \in \nu(c_i)\ (1 \leq i \leq n)$ と仮定する.

1. m が $c_1, \ldots, c_n$ において resolution をもたないなら, $t \underset{I}{\rightarrow} \perp$
2. m が $c_1, \ldots, c_n$ において resolution をもつとき,
   2.1 $m \in M_{n,b}$ ならば, $t \underset{I}{\rightarrow} t[v \leftarrow \mu(m)(o_1, \cdots, o_n)]$
   2.2 $m \in M_{n,u}$ ならば,
   resolution が $(m, (c'_1, \ldots, c'_n), s)$ であれば,
   $t \underset{I}{\rightarrow} t[v \leftarrow s[o_1/x_1, \cdots, o_n/x_n]]$ □

2.3 型整合性

定義 5: $S = (C, \leq, M, \Sigma_b, \Sigma_u)$ をメソッドスキーマ, $I = (\nu, \mu)$ を S の任意の解釈とする. $t = m(o_1, \cdots, o_n)$ $(m \in M_n, o_j \in \nu(c_j))$ の形の任意の項を考える. m がクラス

A Sufficient Condition for Consistency of a Method Schema
Hiroki Dodo†, Yasunori Ishihara‡, Hiroyuki Seki‡†, and Tadao Kasami†‡
† Nara Institute of Science and Technology
‡ Osaka University

```
procedure TEST (C, ≤, M, Σ_b, Σ_u)
  { 初期化 }
  for each (m_u, (c_1, ..., c_n), t) in Σ_u
    for each (c'_1, ..., c'_n) in (m_u, (c_1, ..., c_n), t) を
    resolution とする下位クラス
      output_class(m_u, (c'_1, ..., c'_n)) ← ∅
  for each (m_b, (c_1, ..., c_n → c)) in Σ_b
    for each c' in c の下位クラス
      output_class(m_b, (c_1, ..., c_n)) に c' を追加
  { 本体 }
  repeat
    for each (m_u, (c_1, ..., c_n), t) in Σ_u
      for each (c'_1, ..., c'_n) in
      (m_u, (c_1, ..., c_n), t) を resolution とする下位クラス
        t' ← t[c'_1/x_1, ..., c'_n/x_n];
        UPDATE (m_u, (c'_1, ..., c'_n), t')
  until output_class が一つも更新されない
  "型整合性をもつ" と表示
procedure UPDATE (m, (c_1, ..., c_n), t)
  if t に最左最内リデックスが存在 then
    最左最内リデックスのラベルを m' とする;
    m' の引数を c'_1, ..., c'_n とする;
    if m' のクラス c'_1, ..., c'_n における resolution が存在
    then
      resolution の引数クラスを d'_1, ..., d'_n とする;
    else
      "型整合性をもつかどうかわからない" と表示して
      停止
    if m' が基底メソッド名 then
      o_c ← output_class(m', (d'_1, ..., d'_n))
    else { m' がユーザメソッド名 }
      o_c ← output_class(m', (c'_1, ..., c'_n))
    if o_c ≠ ∅ then
      for each c'' in o_c
        t' ← t[m' ← c''];
        UPDATE (m, (c_1, ..., c_n), t')
  else { 出力クラス c が求まっている }
    output_class(m, (c_1, ..., c_n)) に c を追加
```

図 1: 十分条件判定アルゴリズム

$c_1, \cdots, c_n$において resolution をもつならば決して $t \xrightarrow[I]{*} \bot$ とならないとき, Sは**型整合性**をもつと言う. □

3 型整合性をもつための十分条件

3.1 アルゴリズム

図1に十分条件判定アルゴリズムを示す. 本アルゴリズムの入力はメソッドスキーマ $S = (C, \leq, M, \Sigma_b, \Sigma_u)$ であり, 出力は "型整合性をもつ" あるいは "型整合性をもつかどうかわからない" のいずれかである.

$output\_class(m, (c_1, \cdots, c_n))$ は, クラスの集合を値としてとる変数である. $c \in output\_class(m, (c_1, \cdots, c_n))$ となるのは以下の条件を満たすときである.

- $m \in M_{n,b}$ならば, $(m, (c_1, \cdots, c_n \to c'))$ を resolution としてもつ実引数に対して m を実行したときに, 値のクラスが c になりうる.
- $m \in M_{n,u}$ならば, $o_i \in \nu(c_i)$ $(1 \leq i \leq n)$ を実引数として mを実行したときに, 値のクラスが cになりうる.

本アルゴリズムでは, 各メソッド定義に対して, $output\_class$ を構成しながら, 型整合性をもつかどうかを判定する (手続き $TEST$). 特に, ユーザメソッドの場合, 項中のオブジェクトをそのオブジェクトの属するクラスで置き換え, メソッド名は, 引数クラスに対応する $output\_class$ の元で置き換えて書換えを行う (手続き $UPDATE$).

- $C = \{c, c'\}$, $c' \leq c$
- $\Sigma_b = \{(m_{b_1}, (c \to c)), (m_{b_1}, (c' \to c')), (m_{b_2}, (c' \to c'))\}$
- $\Sigma_u = \{(m_{u_1}, (c), m_{u_1}(m_{b_1}(x_1))), (m_{u_1}, (c'), m_{b_1}(x_1)), (m_{u_2}, (c), m_{b_2}(m_{u_1}(x_1)))\}$

図 2: メソッドスキーマ S

表 1: 基底メソッドに対する $output\_class$

ループ数	$m_{b_1}(c)$	$m_{b_1}(c')$	$m_{b_2}(c')$
初期設定	c, c'	c'	c'

表 2: ユーザメソッドに対する $output\_class$

ループ数	$m_{u_1}(c)$	$m_{u_1}(c')$	$m_{u_2}(c)$	$m_{u_2}(c')$
初期設定	∅	∅	∅	∅
1	∅	c'	∅	c'
2	c'	c'	c'	c'
3	c'	c'	c'	c'

3.2 アルゴリズムの正当性

一般の場合のメソッドスキーマに関しては, 以下の定理が成り立つ.

定理 1: 一般の場合のメソッドスキーマ Sが型整合性をもつときのみ, Sに対して本アルゴリズムは "型整合性をもつ" と出力する. □

また, メソッドがすべて1引数であるモナディックなメソッドスキーマに対しては, 以下の定理が成り立つ.

定理 2: モナディックなメソッドスキーマ Sが型整合性をもつときかつそのときのみ, Sに対して本アルゴリズムは "型整合性をもつ" と出力する. □

3.3 例

図2のメソッドスキーマ Sの型整合性を本アルゴリズムで解析する. まず, 基底メソッドの $output\_class$ は表1のように構成され, 次にユーザメソッドの $output\_class$ は表2のように構成される. 3回目の `repeat` ループで $output\_class$ が一つも更新されないので, ループを抜け, "型整合性をもつ" と出力する. よって, 定理1より Sは無矛盾性の十分条件を満たすので, 無矛盾である.

参考文献

[1] Abiteboul, S. , Kanellakis, P. and Waller, E. : *"Method Schemas"* , Proc. of the Ninth ACM SIGACT-SIGMOD SIGART Symp. on Principles of Database Systems, pp.16–27 (1990).

[2] 二木, 外山 : "項書き換え型計算モデルとその応用", 情報処理, 24, 2, pp.133–146(昭 58-02).