ソフトウェア基礎論　27－6
プログラミング言語　19－6
（1988. 12. 9）

失敗集合に基づく並列論理型プログラムの宣言的意味論

A Declarative Semantics of Parallel Logic Programs based on Failure Set

村上 昌己
Masaki Murakami

（財）新世代コンピュータ技術開発機構研究所
Institute for New Generation Computer Technology

Abstract

筆者は先に [Murakami 88a] で、GHC の様な Horn 論理に基づく並列プログラミング言語についての成功集合意味論について報告した。そこで提案された意味論は[Lloyd 84] における純 Horn 論理型プログラムに対する宣言的意味論の拡張であり、プログラムの表示 (denotation) は単位節にかわって入出力履歴の集合によって与えられた。本稿では、入出力履歴の概念を拡張し、プログラムの失敗集合すなわち実行中に失敗する可能性のあるゴールの集合を定義する。プログラムのセマンティクスは先に [Murakami 88a] で導入された ω 成功集合とここで定義された失敗集合の対によって与えられる。さらにここでは、失敗集合がある連続な関数の最小不動点によって特徴付けられることを示す。

A success set semantics of GHC programs was reported in [Murakami 88a]. That paper adopted the approach which is an extension of declarative semantics of pure Horn logic programs [Lloyd 84]. The semantics of a program is defined as a set of I/O histories instead of unit clauses. This paper presents a failure set semantics of programs as the set of I/O histories. The semantics of programs is defined as the tuple of the ω−success set and the failure set. It also shows that the failure set of a program is characterized as the least fixpoint of a continuous function.

1　はじめに

筆者は先に、Flat GHC の宣言的意味論について報告した [Murakami 88a]。そこで提案したセマンティクスは、純 Horn 論理型プログラムについての宣言的なセマンティクス [Lloyd 84] の拡張 / 変形であり、プログラムのセマンティクスは、そのプログラムを記述している節集合のあるモデルとして定義された。ゴール節がこのモデルで真であるという概念は、そのゴール節が失敗もデッドロックもしないで、実行を続けることが可能であること (すなわち ω成功する可能性があること) に対応していた。またこのモデルでは、通常の純 Horn 論理型プログラムの宣言的セマンティクスにおける Herbrand 基底に代わって入出力履歴 (I/O history: I/O 履歴) の領域を導入し、プログラムの表示 (denotation) すなわちプログラムの ω成功集合 (ω−success set) を I/O 履歴の集合により定義した。

しかしながら、GHC のようなコミット機構を持つような論理型言語の場合、成功集合とデッドロック又は失敗する可能性のあるゴールの集合は空でない共通部分を持つため、あるゴール節がデッドロックしないこと又は単一化が実行中に失敗しないこと等は、成功集合だけでは議論することはできない [Falaschi 88]。

そこで [Falaschi 88] では committed choice 型の言語のセマンティクスについて、新しいアプローチが提案された。そこではプログラムの失敗集合すなわち、実行中にボディ部分での単一化が失敗するような実行をする可能性のあるゴール節を真とするようなモデル理論での最小モデルを定義し、プログラムのセマンティクスは成功集合と失敗集合の対によって与えられた。

しかしながら、このアプローチでは計算が終了した時に得られるプログラムの最終結果のみを含む式によってモデルを構成している。したがって、停止しないプロセスを含むプログラムについては、この方法では対象とはならない。例えば、g_1 が停止しないでかつg_2 が失敗するとき、g_1, g_2 というゴール節はこのセマンティクスでは真とはな

らない。

純 Horn 論理型のプログラミングでは、プログラムは計算開始時にすべての入力を受け取り有限時間後に計算を終了して結果を返すものを通常考え、そのような実行を正常な計算と想定して形式的なセマンティクスを与えている[Lloyd 84]。しかし GHC の様な言語はこれとは異なり、プログラムは実行途中に外部から適切なタイミングで入力を受けとり、それに応じて変数へのバインドを返しながら実行を続ける様なプログラムを記述するものと考えられる[Ueda 89]。したがって、形式的意味論はそうのような無限な実行を正常な計算として定式化できることが重要であると考え、[Murakami 88a] では、その様なプログラムを想定して ω 成功集合による形式的なセマンティクスを与えた。

本論文では、同様な考え方に基づき Flat GHC プログラムの失敗集合すなわち実行中に失敗する可能性のあるゴールの集合を、入出力履歴の概念をもとに定義する。すなわち、ここではゴール節が真であることを、そのゴール節の実行中にボディ部分での単一化を失敗するような場合があることに対応させ、与えられたプログラムの失敗集合を、そのプログラムの最小モデルとして定義する。その結果、プログラムのセマンティクスは [Falaschi 88] と同様にここで定義した失敗集合と ω 成功集合の対によって定義される。これによって、[Falaschi 88] で提案された定式化では不可能だった無限に実行を続けるプロセスを含むプログラムについて、単一化の失敗の有無についての議論が可能となる。さらに本稿では、失敗集合がある連続関数の最小不動点によって特徴付けられることを示す。

2 ガード付ストリーム

本論文では、通常に純 Horn 論理における Herbrand 基底の要素にあたる I/O 履歴の概念を用いる。I/O 履歴は [Levi 88] における guarded atom の概念の拡張である。

I/O 履歴はプロセスの呼ばれたときの形を表わすヘッド H とプロセスのある実行における入出力を示す GU を用いて次のように表わされる。

$$H : -GU$$

ここでHは互いに異なる変数に述語記号を適用したもの，GU はある節にコミットするまでに通過するガードを解くのに必要な代入 σ とコミットした節のボディ部での単一化の実行を表す式 U_b の対 $< \sigma | U_b >$ の集合である。直観的には H という形のゴールの引数が σ によって具体化されると U_b という単一化が行われることを意味する。

本節では、I/O 履歴のボディ部分にあたるガード付ストリームの概念について述べる。[Murakami 88a] で最初に定義したガード付ストリームは、プログラムの計算のうち、失敗 / デッドロックすることなく実行が進むもののみを表現するものであった。本節ではこれを拡張し、プログラムの正常な動作及び失敗を含む動作をそれぞれ表現できるようにする。

2.1 準備

*Var*を可算無限個の変数の集合，*Fun* を関数記号の集合とする。*Fun*の各要素に対しては arity がそれぞれ定まっているものとする。*Var*と*Fun*の元から通常のように定まる項の集合を*Terms* であらわす。項 τ が単純であるとは、τ が変数項であるか、arity が 0 の関数記号であるか，$f(X_1, \ldots, X_n)$ の形で$f \in Fun$ かつ $X_1, \ldots, X_n$ が異なる変数である場合をいう。

本論文で関心があるのは実行中に生ずる単一化の失敗についてである。単一化の失敗は*Fun*が少なくとも 2 つの異なる元を含む場合に起こる。したがって、以下では暗黙のうちに*Fun*が、ただ 1 つの元からなることはないことを仮定する。

本論文では主に、{a, b} という集合の上のリストを扱うプログラムの例を用いて説明を進める。すなわち、*Fun*= {a, b, nil, cons} であり、a, b, nil の arity を 0、consの arity を 2 とする。以下では通常の様に、nil を [ ] で、$\mathtt{cons}(\tau_1, \tau_2)$ を $[\tau_1|\tau_2]$ で表わす。

Terms の上の代入は通常通り定義される。

[定義 1]

$X \in Var$, τを単純な項とするとき，次のような式を単純な代入式 (又は、単に代入式) という。

$$X = \tau$$

特に$X = X$という式は*true*と表記する。

単純な代入式の有限集合を用いて代入を表現することができる。しかしながら一般に単純な代入式の有限集合が常に代入を定めるわけではない。

以下，代入式の集合とそれが定める代入とを同一視する。

[定義 2]

σ を単純な代入式の集合とする。σ が代入であるか又は、ある代入 θ に対して以下のように定義される$\bigcup_{k\to\infty}\theta_k$ に等しいとき、σ をω代入とよぶ。

$$\theta_0 = \theta$$
$$\theta_{k+1} = \theta_k \cup \{X = \tau \mid \text{ある}(Y = \tau') \in \theta_k\text{が存在し、}X\text{は}\tau'\text{に出現し、}(X = \tau'') \notin \theta_k\text{かつ、}\tau\text{に出現するすべての変数について}\theta_k\text{の元の左辺には出現しない.}\}$$

ω代入は無限項の上の写像を定義する。

まず、ガード付きストリームが、GHC プログラムをどのように表現するかについて例を用いて簡単に説明する。形式的な定義については次の節で述べる。

最初にプログラムが失敗しない正常な実行について、考える。例として次の様なプログラムを考える。

```
p1(X,Y) :- X = [A|X1], A = a|
           Y = [B|Y1], B = b, p1(X1,Y1).
p1(X,Y) :- X = [B|X1], B = b|
           Y = [A|Y1], A = a, p1(X1,Y1).
```

以下はプロセス p1 が、まず入力ストリーム X から a をまず読みこんで出力ストリームに b を書きこみ、次に b を読みこんで a を書き込む計算を表現している。

```
p1(X,Y) :-{< {X = [A|X1], A = a}|Y = [B|Y1] >,
           < {X = [A|X1], A = a}|B = b >,
           <{X = [A|X1], A = a, X1 = [B1|X2], B1 = b}|
            Y1 = [A1|Y2] >,
           < {X = [A|X1], A = a, X1 = [B1|X2], B1 = b}|
            A1 = a >,...}
```

次に失敗する可能性のある計算の表現例を示す。ここでは失敗する計算は U_b のかわりに $\perp$ を含むガード付ストリームによって表現される。$<\sigma|\perp>$ は直観的には、プロセスの引数が σ によって具体化された後、ある節にコミットしてボディ部分での単一化が失敗することを現わす。

例えば次の様なプログラムで、

```
p(X, Y) :- X = b | Y = a
```

このプログラムのもとで、p(X, Y) というゴールは $\{X = b, Y = b\}$ という入力代入を受けとると、上の節にコミットできその後に失敗する。このような実行は次の様なガード付きストリームによって表現される。

$$\{<\{X = b, Y = b\}|\perp>\}$$

一方、ゴールが受けとった入力代入が $\{X = b\}$ であった場合は計算は失敗しない。そのような状況は次のようなガード付きストリームによって表現される。

$$\{<\{X = b, Y = b\}|Y = a>\}$$

一つの I/O 履歴はそのプロセスの可能な実行の一つを表わす。したがって，実行中に異なる節にコミットする可能性がある場合は，異なる I/O 履歴が存在する。また，同じ節にコミットした実行でも，並列に走っているプロセスのスケジューリングによって異なる I/O 履歴が存在する可能性がある。

2.2 形式的定義

以下ではガード付ストリームの形式的な定義を述べる。

[定義 3]
$X \in Var$, τを単純な項とするとき，次のような式を単純な判定式 (または単に判定式) という。

$$X? = \tau$$

[定義 4]
代入σ，変数X，単純な項τについて，$uni(X,\tau)$を代入式 $X = \tau$ または 判定式$X? = \tau$とするとき、$<\sigma|U>$をガード付代入とよぶ。ここでUは$uni(X,\tau)$または記号$\perp$であるとする。このときσを$<\sigma|U>$のガード部，Uを能動部とよぶ。

直観的には$uni(X,\tau)$が代入式であったときは実際にXを具体化する単一化に，判定式であった場合はテスト・ユニフィケーションに対応する。また能動部に$\perp$が出現したときは、ここで失敗する単一化が呼びだされたことに対応する。

[定義 5]
$<\sigma|U>$をガード付代入とする。$|<\sigma|U>|$ は次のように定義される判定式又は代入式の集合である。
U が判定式又は代入式のとき、

$$|<\sigma|U>| = \{U\} \cup \sigma$$

$U = \perp$ のとき、

$$|<\sigma|U>| = \sigma$$

プログラムのある動作を表現するガード付代入の集合 GU は、実行順序に関して半整列集合となっている。すなわち、$<\sigma_1|U_{b1}>, <\sigma_2|U_{b2}> \in GU$について、$\sigma_1 \subset \sigma_2$ならば、$U_{b1}$は$U_{b2}$より先に実行可能となる。

[定義 6]
与えられたガード付代入の集合GUの上に次のような関係$\prec$を定義する。$<\sigma_1|u_1>, <\sigma_2|u_2> \in GU$とする。$\sigma_1 \subset \sigma_2$ かつ $\sigma_1 \neq \sigma_2$ のとき、

$$<\sigma_1|u_1> \prec <\sigma_2|u_2>$$

とする。このように定義された関係$\prec$が半整列集合になることは容易に示せる。

[定義 7]
ガード付代入の集合 GU が次の条件をみたすとき，ガード付ストリームとよぶ。

1) $<\sigma_1|U_1>, <\sigma_2|U_2> \in GU, <\sigma_1|U_1> \neq <\sigma_2|U_2>$ かつU_1とU_2が同じ左辺をもつならば、少なくとも一方は判定式であり，右辺は単一化可能. またU_1が代入式でU_2 が判定式であったとき

$$<\sigma_2|U_2> \prec <\sigma_1|U_1>$$

でない。

2) 任意の $<\theta|X = \tau> \in GU$ について、$<\sigma|U> \in GU$ ならば、$(X = \tau) \notin \sigma$.

3) 任意の $<\theta|X? = \tau>, <\sigma|U> \in GU$について、$\tau$ と τ' が単一化不能ならば、$(X = \tau') \notin \sigma$.

4) 任意の $<\sigma_1|u_1>, <\sigma_2|u_2> \in GU$について、$(X = \tau) \in \sigma$ かつ $(X = \tau') \in \sigma'$ ならば、τ と τ' は単一化可能。

上の条件 1),...,4) は、いずれも GHC における変数が論理的な変数であり、一度具体化された値が別の値になることがないことによる。

[例 1]

次のような集合GU_1, GU_2について，GU_1はZにaの列を次々と読みこんでb の列をYに出力するプロセスの，GU_2は Xに aの列を，Y に b の列を読みこんで両者をマージし，Zに出力するプロセスの動作の一例をそれぞれあらわしている。

$$GU_1 = \{gu_{1i}(1 \leq i)\},$$

$gu_{11} =< \{Z = [A|Z1], A = a\}|Y = [B|Y1] >$
$gu_{12} =< \{Z = [A|Z1], A = a\}|B = b >$
$gu_{13} =<\{Z = [A|Z1], A = a, Z1 = [A1|Z2], A1 = a\}|$
$\quad Y1 = [B1|Y2] >$
$gu_{14} =< \{Z = [A|Z1], A = a, Z1 = [A1|Z2], A1 = a\}|$
$\quad B1 = b >$
.....

$$GU_2 = \{gu_{2j}(1 \leq j)\}$$

$gu_{21} =< \{X = [A0|X1], A0 = a\}|Z = [A|Z1] >$
$gu_{22} =< \{X = [A0|X1], A0 = a\}|A = a >$
$gu_{23} =<\{X = [A0|X1], A0 = a, Y = [B|Y1], B = b\}|$
$\quad Z1 = [A1|Z2] >$
$gu_{24} =< \{X = [A0|X1], A0 = a, Y = [B|Y1], B = b\}|$
$\quad A1 = b >$
$gu_{25} =< \{X = [A0|X1], A0 = a, Y = [B|Y1], B = b,$
$\quad Y1 = [B1|Y2], B1 = b\}|Z2 = [B2|Z3] >$
$gu_{26} =< \{X = [A0|X1], A0 = a, Y = [B|Y1], B = b,$
$\quad Y1 = [B1|Y2], B1 = b\}|B2 = b >$
.....

□

次の概念は、並列に走る複数のゴールを含むようなゴール節について、各ゴールの動きを記述するガード付ストリームから、全体の動作を記述するガード付ストリームを得る操作を定義している。

以下ではUは代入式、判定式、または$\perp$であるとする。

[定義 8]

$GU_1, \ldots, GU_n$ をガード付きストリームとする。$Gu_k(1 \leq k)$ を次のように定義する。

$$Gu_1 = \{< \sigma|U > \mid \exists i, \exists < \sigma|U > \in GU_i, \forall (X = \tau) \in \sigma, \forall j, < \sigma'|X = \tau > \notin GU_j\}$$

$$\begin{aligned} Gu_{k+1} = Gu_k \cup \{ & < \sigma|U > | \exists i, \exists < \sigma'|U > \in GU_i, \forall (X = \tau) \in \sigma', \\ & ((\forall j, < \sigma''|X = \tau > \notin GU_j) \vee \\ & \quad < \sigma''|X = \tau > \in Gu_k) \wedge \\ & \sigma = (\sigma' - \{X = \tau | < \sigma''|X = \tau > \in Gu_k\}) \cup \\ & \quad \{U|U \in \sigma'' < \sigma''|X = \tau > \in Gu_k\}\} \end{aligned}$$

このとき、GU を以下のように定める。

$$GU = \bigcup_{k \to \infty} Gu_k$$

このGUがガード付ストリームになり、かつ

$$\{U| < \sigma|U > \in GU\} = \{U|\exists i < \sigma|U > \in GU_i\}$$

のとき、GUを $GU_1, \ldots, GU_n$の同期付マージとよび、

$$GU_1 \| \ldots \| GU_n.$$

であらわす。

$n = 1$ の場合同期付マージは常に定義でき結果はGU_1自身と等しくなる。

[例 2]**Brock Ackermannの例題**

次の例はの Brock Ackermann の例題 [Brock 81] を GHC で記述したもの [Takeuchi 86](付録) の動作を表わしている。すなわち、次のGU_1は付録で定義された節集合D_{BA}のもとで、 s(In, Mid, Out) というゴールに、

$$\sigma_1 = \{In = [A|In1], A = a, Mid = [B|M1], B = b\}$$

という入力代入が与えられた場合の、またGU_2は次の様なゴール
inv(Out, Mid) に、

$$\sigma_2 = \{Out = [A1|Out1], A1 = a\}$$

という入力代入が与えられた場合の実行を表現している。

$GU_1 = \{gu_{11}, gu_{12}, gu_{13}, gu_{14}\}$

$gu_{11} =<\{In = [A|In1], A = a, Mid = [B|M1], B = b\}|$
$\quad Out = [A1|Out1] >$
$gu_{12} =< \{In = [A|In1], A = a, Mid = [B|M1], B = b\}|$
$\quad A1 = a >$
$gu_{13} =< \{In = [A|In1], A = a, Mid = [B|M1], B = b\}|$
$\quad Out1 = [B1|Out2] >$
$gu_{14} =< \{In = [A|In1], A = a, Mid = [B|M1], B = b\}|$
$\quad B1 = b >$

$GU_2 = \{gu_{21}, gu_{22}\}$

$gu_{21} =< \{Out = [A1|Out1], A1 = a\}|Mid = [B|M1] >$
$gu_{22} =< \{Out = [A1|Out1], A1 = a\}|B = b >$

この場合，$Gu_1 = \phi$であり、したがって$GU = \phi$。一方，

$$\begin{aligned} U_2 = \{ & Out = [A1|Out1], A1 = a, \\ & Out1 = [B1|Out2], B1 = b, \\ & Mid = [B|M1], B = b\} \end{aligned}$$

であり，ゆえにGU_1とGU_2の同期付マージは定義できない。しかし，GU_2は次のようなGU_1'とは同期付マージが定義できる。これは付録定義されたD'_{BA}での同じゴールについての計算を表現している。

$GU_1' = \{gu_{11}', gu_{12}', gu_{13}', gu_{14}'\}$

$gu'_{11} =< \{\mathtt{In} = [\mathtt{A}|\mathtt{In1}], \mathtt{A} = \mathtt{a}\}|\mathtt{Out} = [\mathtt{A1}|\mathtt{Out1}] >$
$gu'_{12} =< \{\mathtt{In} = [\mathtt{A}|\mathtt{In1}], \mathtt{A} = \mathtt{a}\}|\mathtt{A1} = \mathtt{a} >$
$gu'_{13} =<\{\mathtt{In} = [\mathtt{A}|\mathtt{In1}], \mathtt{A} = \mathtt{a}, \mathtt{Mid} = [\mathtt{B}|\mathtt{M1}], \mathtt{B} = \mathtt{b}\}|$
$\mathtt{Out1} = [\mathtt{B1}|\mathtt{Out2}] >$
$gu'_{14} =< \{\mathtt{In} = [\mathtt{A}|\mathtt{In1}], \mathtt{A} = \mathtt{a}, \mathtt{Mid} = [\mathtt{B}|\mathtt{M1}], \mathtt{B} = \mathtt{b}\}|$
$\mathtt{B1} = \mathtt{b} >$

結果は次のようになる。

$$\sigma_{BA} = \{\mathtt{In} = [\mathtt{A}|\mathtt{In1}], \mathtt{A} = \mathtt{a}\}$$

とおくと:

$$GU'_1 \| GU_2 = \{< \sigma_{BA}|\mathtt{Out} = [\mathtt{A1}|\mathtt{Out1}] >,$$
$$< \sigma_{BA}|\mathtt{A1} = \mathtt{a} >,$$
$$< \sigma_{BA}|\mathtt{Mid} = [\mathtt{B}|\mathtt{M1}] >,$$
$$< \sigma_{BA}|\mathtt{B} = \mathtt{b} >,$$
$$< \sigma_{BA}|\mathtt{Out1} = [\mathtt{B1}|\mathtt{Out2}] >,$$
$$< \sigma_{BA}|\mathtt{B1} = \mathtt{b} >\}$$

これは`s(In, Mid, Out)`, `inv(Out, Mid)`というゴール節に$\sigma_1 \cup \sigma_2$という入力代入を与えて実行した場合の動作を表現している。

□

[定義 9]
GUをガード付ストリーム、Vを変数の有限集合とする。ここでGUのVによる制限$GU \downarrow V$とは次のような集合である。

$$GU \downarrow V = \{< \sigma|uni(X,\tau) > |$$
$$\exists k < \sigma|uni(X,\tau) > \in GU, X \in V_k\}$$

ここで、

$V_0 = V$
$V_{i+1} = V_i \cup \{X|\exists gu \in GU, \exists uni(Y,\tau) \in |gu|,$
Xはτに含まれ$Y \in V_i$, かつ
$\forall gu' \in GU, gu' \prec < \sigma|U >$ ならば
Xはgu'に現れない. $\}$

GUがガード付ストリームのとき、$GU \downarrow V$もガード付ストリームとなる。

[定義 10]
GUをガード付ストリーム，θを単純な代入式の集合とするとき、θとGUから定まる次の集合がやはりガード付ストリームとなるとき、

$$\{< \sigma|U_b > | < \sigma'|U_b > \in GU, \sigma = \theta \cup \sigma'\}$$

これを$GU \bowtie \theta$で表わす。

3 モデル論的意味論

次に、先に述べたガード付ストリームの概念を基に、純 Horn 論理型プログラムにおける Herbrand 基底の概念に相当するものを並列論理型言語に対して導入する。

まず、GHC の計算規則にそって実行される簡単な並列プログラミング言語を示す。この言語は議論を単純にするために Flat GHC の簡単なサブセットを採用している。しかし，この制限が一般性を失うものではないことは容易に示される。本質的には [Levi 88] の strong normal form への変換アルゴリズムの拡張によって示される。

[定義 11]
述語記号の集合を$Pred$とする。$H, B_1, B_2, \ldots, B_n$を$Pred$, $Terms$から作られる原子式，かつHの引数部に出現するのは全て異なる変数，$U_{g1}, \ldots, U_{gm}, U_{b1}, \ldots, U_{bh}$を単純な代入式とする。このとき次の節：

$$H : -U_{g1}, \ldots, U_{gm}|U_{b1}, \ldots, U_{bh}, B_1, B_2, \ldots, B_n$$

をガード付節とよぶ。ガード付節の有限集合をプログラムとよぶ。

以下では，Hが$p(X_1, X_2, \ldots, X_k)$のとき

$$Var(H) = \{X_1, X_2, \ldots, X_k\}$$

とする。

[定義 12]
p を arity k の $Pred$ の元、$X_1, X_2, \ldots, X_k$ を互いに異なる変数、σ を ω 代入とするとき、$\sigma p(X_1, X_2, \ldots, X_k)$はゴールである。

単一化はいきなりゴールしては現われないことに注意されたい。

[定義 13]
ゴール節とは、ゴール $g_i (1 \leq i \leq n)$ の並び $g_1, \ldots, g_n$ である。

[定義 14]
GUをガード付きストリームとする。I/O 履歴 t とは次のようなものである。

$$p(X_1, X_2, \ldots, X_k) : -GU$$

ここで$p \in Pred$ でアリティ k、かつ$X_1, X_2, \ldots, X_k$ は互いに異なる変数であり、

$$GU \downarrow Var(p(X_1, X_2, \ldots, X_k)) = GU.$$

ここで $p(X_1, X_2, \ldots, X_k)$ を t のヘッド部分，GUをボディ部分とよぶ。直観的にはGUにはヘッド部分から“見える”変数しか出現しない。したがって，プロセスの内部変数についての情報は乗てられている。すなわち，ボディ部のガード付ストリームが表わしているのは「このプロセスが何をするか」であり「このプロセスはどのように計算を行うか」ではない。

I/O 履歴は並列言語における単位節の概念に相当する。しかしながら I/O 履歴では本質的に同じ計算に対して複

数の記法が可能となる。 すなわちヘッドに出現する変数以外の変数については，いかなる変数が用いられようと外から観測する限りにおいては同じ計算を表現しているものとみなすことができる。そこで厳密にはI/O履歴の領域に変数名のつけかえによる適当な同値関係[1]を導入し、その同値類でHerbrand基底にあたるものを定義するのが適切である。

以下では、*Fun, Var, Pred* から定まるすべてのI/O履歴の集合の変数名のつけかえによる同値類から、適当な代表元を選びだした集合を $I/Ohist$ で表し、その元をI/O履歴とよぶ。

[定義 15]
$H : -GU$ を入出力履歴とする。GU の任意の元 $< \sigma|U >$ について U が判定式又は代入式であるとき、$H : -GU$を正常な履歴とよぶ。一方、U が $\perp$ であるような元を含むとき、失敗する履歴と呼ぶ。

$I/Ohist$は、すべての正常な履歴の集合$I/Ohist_\infty$と失敗する履歴の集合$I/Ohist_\perp$に分割される。

[定義 16]
$I/Ohist_\infty$ の任意の部分集合を∞解釈、 $I/Ohist_\perp$の任意の部分集合を$\perp$解釈と呼ぶ。

[定義 17]
t をI/O履歴、g をゴールとするとき、t がg のトレースであるとは、次の(1),...,(3) が成り立つことをいう。

(1) t は$H : -GU$という形をしており、あるω代入σが存在して、 $\sigma H = g$となる。

(2) 任意の$< \theta|U >\in GU$について、$\theta \subset \sigma$ となる。

(3) 任意の$< \theta|U >\in GU$について、Uが代入式$X = \tau$の場合 σはXを具体化せず、U が判定式$X? = \tau$の場合 σXは $\sigma\tau$に等しい。

ここで写像σが変数Xを具体化しないとは、$\sigma X = Y(\in Var)$ かつ$\sigma Z = Y$ となるZがX以外に存在しないことをいう。

ゴールgのトレースtについて、tが正常な履歴であるならば正常なトレースとよぶ。また失敗する履歴であるとき、失敗するトレースとよぶ。

[定義 18]
$I_\perp$を$\perp$解釈、gをゴールとするとき、gが $I_\perp$で真であるとはあるω代入σが存在し、σgの失敗するトレースが$I_\perp$に含まれることをいう。 また∞解釈 I_∞でgが真であるとは、gの成功するトレースがI_∞に含まれることをいう。

失敗集合を定義するための準備として [Murakami 88a] で定義したω成功集合他のいくつかの概念を用意する。

[定義 19]
I_∞ を ∞ 解釈、$g_1, \ldots, g_n$をゴール節とする。 $g_1, \ldots, g_n$ がI_∞で真であるとは、ある ω 代入 σ が存在し、各$g_i(1 \leq i \leq n)$について σg_i のトレースt_iがI_∞に含まれ、$t_1, \ldots, t_n$ のボディ部分の同期付マージが定義できることである。また空(すなわち$n = 0$ の場合) なゴール節は常に真であるとする。

[定義 20]
ガード付節の集合Dについて、解釈I_∞がDの∞モデルであるとは、 I_∞の任意の元tについてある節:

$$H : -U_{g1}, \ldots, U_{gm}|X_1 = \tau_1, \ldots, X_h = \tau_h, B_1, \ldots, B_n \in D$$

が存在して、tが次のような形をしていることである。

$$\begin{aligned} H : -\{ &< \{U_{g1}, \ldots, U_{gm},\}|uni(X_1, \tau_1) >, \ldots, \\ &< \{U_{g1}, \ldots, U_{gm},\}|uni(X_h, \tau_h) >\} \cup \\ &((GU_1 \| \ldots \| GU_n) \bowtie \{U_{g1}, \ldots, U_{gm}\}) \downarrow Var(H) \end{aligned}$$

ここでGU_iはσB_iというゴールのトレース($\in I$) のボディ部、 σは

$$\{U_{g1}, \ldots, U_{gm}\} \cup \{X_1 = \tau_1, \ldots, X_h = \tau_h\} \cup \sigma'$$

という形の$Var(H) \cup \{X_1, \ldots, X_h\}$に制限された$\omega$−代入であり、かつ

$$\forall < \theta|U > \in (GU_1 \| \ldots \| GU_n) \bowtie \{U_{g1}, \ldots, U_{gm}\}, \theta \subset \sigma.$$

となる。

与えられたガード付き節の集合Dについて最大のωモデルが一意に存在することが示される [Murakami 88a]。この最大モデルをDのω成功集合とよび M_∞^D で表わす。

次に失敗するゴールを真とするモデル理論を導入する。

以下では$\langle I_\perp, I_\infty \rangle$ を$\perp$解釈:$I_\perp$ 及び ∞解釈: I_∞の対とする。

[定義 21]
ゴール節 $g_1, \ldots, g_n$ が $\langle I_\perp, I_\infty \rangle$ で真であるとは、ある ω 代入 σ が存在し、各 $g_i(1 \leq i \leq n)$ について σg_i のトレース t_i が $I_\perp \cup I_\infty$ に含まれ、少なくともひとつの j について $t_j \in I_\perp$ であり、かつ $t_1, \ldots, t_n$ のボディ部分の同期付マージが定義できることである。このとき σ を $g_1, \ldots, g_n$ を真とする代入と呼ぶ。

また空(すなわち $n = 0$ の場合) なゴール節は、すべての ∞ 解釈で真であり、すべての $\langle I_\perp, I_\infty \rangle$ で偽であるとする。

ここで与えられた定義では、手続き的にはたとえ無事に走り続けるプロセスがあったとしても、どれかひとつのプロセスで失敗すれば、全体としては異常動作とみなす立場をとっている。

[定義 22]
ガード付節:

$$H : -U_{g1}, \ldots, U_{gm}|U_{b1}, \ldots, U_{bh}, B_1, B_2, \ldots, B_n$$

が$\langle I_\perp, I_\infty \rangle$で真であるとは、次の三つの条件が真であることを言う。

[1][Murakami 88a] でこのような同値関係の形式的な定義とその定義の妥当性について議論している。

(1) $U_{b1},\ldots,U_{bh}$ が H を通して外部から見える変数への具体化を含むならば、

$$H:-<\{U_{g1},\ldots,U_{gm}\}|\ \bot>\in I_\bot$$

である。

(2) 次の様な二つの代入、$\{U_{g1},\ldots,U_{gm}\},\{U_{b1},\ldots,U_{bh}\}$ によってそれぞれ具体化される変数の集合が空でない共通部分を持ち、かつ両者の具体化の結果同士が単一化できないならば、

$$H:-<\{U_{g1},\ldots,U_{gm}\}|\ \bot>\in I_\bot$$

である。

(3) Hから見えない変数を具体化せずかつ ゴール節 $B_1,B_2,\ldots,B_n$ を$\langle I_\bot,I_\infty\rangle$ で真とするω 代入 σ が存在するならば、次のような入出力履歴が $I_\bot$に含まれる。

$$H:-\{<\{U_{g1},\ldots,U_{gm},\}|U_{b1}>,\ldots,<\{U_{g1},\ldots,U_{gm},\}|U_{bh}>\}\cup((GU_1\|\ldots\|GU_n)\bowtie\{U_{g1},\ldots,U_{gm}\})\downarrow Var(H)$$

ここでGU_iはσB_i というゴールのトレース($\in I_\bot\cup I_\infty$)のボディ部分である。

[定義 23]

Dを GHC プログラム、M_∞^D を D の ω成功集合とする。⊥解釈:$I_\bot$ が D の ⊥モデルであるとは、D に含まれる全ての節が、$\langle I_\bot,M_\infty^D\rangle$ で真であることをいう。

定義より、D の⊥モデルは複数存在する。次の命題は⊥モデルの定義より明らかになりたつ。

[命題 1]

添え字の集合Indについて、節集合Dの⊥モデルの族 M_i $(i\in Ind)$を考える。 このとき

$$\bigcap_{i\in Ind}M_i$$

はやはりDの⊥モデルである。

上の命題よりDの全ての⊥モデルの共通部分をとったものはやはり Dの⊥モデルであり、これはDの最小の⊥モデルとなる。 これを、Dの失敗集合と呼び、$M_\bot^D$であらわす。節集合DのセマンティクスとはDの失敗集合とω成功集合の対$\langle M_\bot^D,M_\infty^D\rangle$によって定義される。

与えられた flat GHC プログラム D について、上で定義されたセマンティクスで真となるゴール節は、D で実行した際に失敗する可能性のあるゴール節となっている。

4 不動点的セマンティクス

[Murakami 88a] では、与えられた flat GHC プログラム の ω 成功集合が、そのプログラムから得られるある連続関数の最大不動点で特徴付けられることを示した。

本節では先に定義した失敗集合が、ω成功集合と D から定義される関数の最小不動点によって特徴づけられることをしめす。

$I/O-hist$が与えられたとき、全ての解釈の集合IPは集合の包含関係のもとで完備束となり、最大元は$I/O-hist$、最小限は空集合ϕである。

[定義 24]

ガード付き節の有限集合 Dについて、集合 $Base_D$ を次の様に定める。

$$\begin{aligned}Base_D=\{&H:-<\sigma|\ \bot>|\\&H:-U_{g1},\ldots,U_{gm}|\\&\quad U_{b1},\ldots,U_{bh},B_1,\ldots,B_n\in D\\&\text{について}U_{b1},\ldots,U_{bh}\text{ が }H\text{ を通して}\\&\text{外部から見える変数への具体化を含むか、}\\&\text{又は二つの代入}\{U_{g1},\ldots,U_{gm}\},\{U_{b1},\ldots,U_{bh}\}\\&\text{によって具体化される変数の集合が}\\&\text{空でない共通部分を持ち、かつ両者の}\\&\text{具体化の結果同士が単一化できない。}\}\end{aligned}$$

このとき D から定まる関数 $\Psi_D:IP\to IP$ を次の様に定義する。

$$\begin{aligned}\Psi_D(S)=&Base_D\cup\\&\{H:-\{<\{U_{g1},\ldots,U_{gm},\}|U_{b1}>,\ldots,\\&\quad<\{U_{g1},\ldots,U_{gm},\}|U_{bh}>\}\cup\\&((GU_1\|\ldots\|GU_n)\bowtie\{U_{g1},\ldots,U_{gm}\})\downarrow Var(H)|\\&H:-U_{g1},\ldots,U_{gm}|U_{b1},\ldots,U_{bh},B_1,\ldots,B_n\in D\\&\text{について }H\text{ から見えない変数を具体化せず、}\\&B_1,\ldots,B_n\text{を}\langle S,M_\infty^D\rangle\text{で真とするようなω代入}\sigma\\&\text{が存在し、}GU_i\text{は}\sigma B_i\text{ というゴールの}\\&\text{トレース}(\in S\cup M_\infty^D)\text{ のボディ部分.}\}\end{aligned}$$

Ψ_D の性質を示す前に、いくつかの概念と命題を用意する。これは [Park 69] 等による。

鎖$S_i:S_1\subset S_2\subset\ldots$について、$S_i$の最小上界を $\bigcup S_i$ で表わすことにする。

Lを完備束とする。 関数$f:L\to L$が, 任意の鎖$S_i:S_1\subset S_2\subset\ldots$について

$$\bigcup\{f(S_i)|0\le i\}=f(\bigcup\{S_i|0\le i\}).$$

となるとき、fは上向きにω−連続であるという。

よく知られているように, 関数fがω−連続ならばfは単調である。すなわち$S_1\subset S_2$ ならば $f(S_1)\subset f(S_2)$ である。また次の 2 つの命題はよく知られたものである。

[命題 2]

完備束Lの最小元を$\bot_L$とするとき、fが上向きに連続であるならばfの最小不動点lfp$f$は次のような式で特徴づけられる。

$$\mathsf{lfp}f=\bigcup\{f^n(\bot_L)|n\ge 0\}$$

ここで、$f^0(X)=X,f^{n+1}(X)=f(f^n(X))$とする。

[命題 3]

fが単調な関数のとき、次の式が成立する。

$$\mathrm{lfp} f = \bigcap\{X | f(X) \subset X\}$$

ここで、$\bigcap S$ は集合 S の下限である。

以下に$\Psi_D(S)$ の性質を幾つか示す。

[命題 4]

$\Psi_D(S)$ は上向きにω−連続である。

証明:

任意の鎖 $S_i : S_1 \subset S_2 \subset \ldots$ について、

$$\Psi_D(\bigcup S_j) = \bigcup\{\Psi_D(S_j)\}$$

であることを示せばよい。

(1) $\Psi_D(\bigcup S_j) \subset \bigcup\{\Psi_D(S_j)\}$:

任意の $t \in \Psi_D(\bigcup S_j)$ について、$t \in Base_D$ の場合は定義より S_j にかかわりなく、明らかに $t \in \Psi_D(S_j)$ であり、

$$t \in \bigcup\{\Psi_D(S_j)\}.$$

一方、$t \notin Base_D$ の場合は定義より、

$t \in \{H: -\{< \{U_{g1}, \ldots, U_{gm},\} | U_{b1} >, \ldots,$
$< \{U_{g1}, \ldots, U_{gm},\} | U_{bh} >\} \cup$
$((GU_1 \| \ldots \| GU_n) \bowtie \{U_{g1}, \ldots, U_{gm}\}) \downarrow Var(H) |$
$H : -U_{g1}, \ldots, U_{gm} | U_{b1}, \ldots, U_{bh}, B_1, \ldots, B_n \in D$
について H から見えない変数を具体化せず、
$B_1, \ldots, B_n$を$\langle \bigcup S_j, M_\infty^D \rangle$で真とするような
ω代入σ が存在し、GU_iはσB_i というゴールの
トレース$(\in \bigcup S_j \cup M_\infty^D)$ のボディ部分.$\}$

各 i について σB_i のトレースが $\bigcup S_j$ に含まれることから、$S_1, S_2, \ldots$ は鎖であることより、ある k が存在し任意の iについてσB_iのトレースがS_kに含まれる。

したがって、Ψ_D の定義より

$$t \in \Psi_D(S_k).$$

ゆえに、

$$t \in \bigcup\{\Psi_D(S_j)\}$$

となる。

(2) $\Psi_D(\bigcup S_j) \supset \bigcup\{\Psi_D(S_j)\}$:

任意の $t \in \bigcup\{\Psi_D(S_j)\}$ について、あるkが存在して、

$$t \in \Psi_D(S_k)$$

となる。このとき、$t \in Base_D$ ならば (1) の場合と同様に明らかに、

$$t \in \Psi_D(\bigcup S_j)$$

である。一方、

$t \in \{H: -\{< \{U_{g1}, \ldots, U_{gm},\} | U_{b1} >, \ldots,$
$< \{U_{g1}, \ldots, U_{gm},\} | U_{bh} >\} \cup$
$((GU_1 \| \ldots \| GU_n) \bowtie \{U_{g1}, \ldots, U_{gm}\}) \downarrow Var(H) |$
$H : -U_{g1}, \ldots, U_{gm} | U_{b1}, \ldots, U_{bh}, B_1, \ldots, B_n \in D$
について H から見えない変数を具体化せず、
$B_1, \ldots, B_n$を$\langle S_k, M_\infty^D \rangle$で真とするような
ω代入σ が存在し、GU_iはσB_i というゴールの
トレース$(\in S_k \cup M_\infty^D)$ のボディ部分.$\}$

の場合、各 σB_i のトレース $(\in S_k)$ は $S_1, S_2, \ldots$ は鎖であることより、$\bigcup S_j$ に含まれる。したがって、Ψ_Dの定義より、

$$t \in \Psi_D(\bigcup S_j)$$

となる。

□

[命題 5]

$\Psi_D(I) \subset I$であるときかつそのときのみ、IはDの⊥モデル。

証明:

if part:

D に含まれる任意の節が $\langle I, M_\infty^D \rangle$ で真であることを示せばよい。ある節が条件 (1), (2) の前提にあてはまる場合ついては、仮定よりIは$Base_D$を含むので明らかに成立する。条件 (3) については、 Hから見えない変数を具体化せずかつ $B_1, B_2, \ldots, B_n$ を真とする ω 代入 σ が存在するとき、次の入出力履歴tがIに含まれることを示せばよい。

$H : -\{< \{U_{g1}, \ldots, U_{gm},\} | U_{b1} >, \ldots,$
$< \{U_{g1}, \ldots, U_{gm},\} | U_{bh} >\} \cup$
$((GU_1 \| \ldots \| GU_n) \bowtie \{U_{g1}, \ldots, U_{gm}\}) \downarrow Var(H)$

Ψ_Dの定義より、tは$\Psi_D(I)$に含まれ、さらに仮定より $t \in I$ となる。

only if part:

任意の$t \in \Psi_D(I)$ について、$t \in Base_D$ の場合はIがモデルであることより、モデルの定義の条件 (1) および (2) から明らかに$t \in I$となる。

一方、

$t \in \{H : -\{< \{U_{g1}, \ldots, U_{gm},\} | U_{b1} >, \ldots,$
$< \{U_{g1}, \ldots, U_{gm},\} | U_{bh} >\} \cup$
$((GU_1 \| \ldots \| GU_n) \bowtie \{U_{g1}, \ldots, U_{gm}\}) \downarrow Var(H) |$
$H : -U_{g1}, \ldots, U_{gm} | U_{b1}, \ldots, U_{bh}, B_1, \ldots, B_n \in D$
について H から見えない変数を具体化せず、
$B_1, \ldots, B_n$を$\langle I, M_\infty^D \rangle$で真とするような$\omega$代入$\sigma$ が存在し、
GU_iはσB_i というゴールの
トレース$(\in I \cup M_\infty^D)$ のボディ部分.$\}$

の場合は、モデルの定義の条件 (3) より$t \in I$ となる。

□

以上により、次の定理が得られる。

[定理] 節集合Dの失敗集合$M_\perp^D$はΨ_D の最小不動点であり、次の式で表わされる。

$$M_\perp^D = \bigcup\{\Psi_D^n(\phi) | n \geq 0\}$$

5 おわりに

本稿では、Flat GHC プログラムの失敗する動作を、宣言的に特徴付けるセマンティクスすなわち、プログラムの失敗集合を計算履歴集合の概念に基づいて定義し、その最小不動点による特徴付けについて述べた。ここで述べた失敗集合は、プログラムが計算途中に好ましくない失敗をすることがないことを議論するためのものであった。しかしながら、本稿ではゴール及びゴール節についての真の概念を、外部からのある代入によって失敗する可能性があることに対応付けた。このことからわかるように、何らかの意味のある出力をする殆どすべてのゴールは失敗集合意味論で真となる。このことからわかるように、ゴールを真とするω代入が、そのプログラム自身によって値が具体化されるいかなる変数をも具体化することがないか否かの議論するための手法は、失敗集合意味論の有用性をより明確にするものとして重要である。これについては今後の課題として残されている。また、本稿ではプログラムの異常な動作として単一化の失敗のみを扱ったが、並列プログラムについてはデッドロックの検出は重要な問題である。本稿で述べた入出力履歴は、デッドロックの宣言的 / 不動点的特徴付けにも有用であると考えられる。

6 謝辞

本研究をすすめるにあたり、有益な議論をして下さったICOT 古川次長、第1研究室の皆様、およびPisa大学 Levi教授に感謝します。

References

[Apt 82] K. Apt and M. H. Van Emden, Contributions to the theory of logic programming, J. Assoc. Comput. Mach. 29, (1982)

[Brock 81] J. D. Brock, W. B. Ackermann, Scenarios: A Model of Non-determinate Computation, Lecture Notes in Computer Science, No. 107, Springer, 1981

[Falaschi 88] M. Falaschi and G. Levi, Operational and fixpoint semantics of a class of committed-choice logic languages, Dipartimento di Informatica, Università di Pisa, Italy, Techn. Report, January 1988

[Levi 87] G. Levi and C. Palamidessi, An Approach to the Declarative Semantics of Synchronization in Logic Languages, Proc. of International Conf. on Logic Programming 87, 1987

[Levi 88] G. Levi, A new declarative semantics of Flat Guarded Horn Clauses, ICOT Technical Report, 1988

[Lloyd 84] J. W. Lloyd, Foundations of logic programming, Springer-Verlag, 1984

[Maher 87] M. J. Maher, Logic Semantics for a Class of Committed-Choice Programs, Proc. of International Conf. on Logic Programming 87, 1987

[Murakami 88a] M. Murakami, A New Declarative Semantics of Parallel Logic Programs with Perpetual Processes, to appear in Int. Conf.on Fifth Generation Computer System 1988, 1988

[Murakami 88b] M. Murakami, A Declarative Semantics of Parallel Logic Programs based on Failure/Deadlock Set, Proc. of 5th Conf. of JSSST, (in Japanese) 1988

[Park 69] D. Park, Fixpoint induction and proofs of program properties, Machine Intelligence 5, Edinburgh University Press, Edinburgh, 1969

[Saraswat 85] V. A. Saraswat, Partial Correctness Semantics for CP[↓, |, &], Lecture Notes in Comp. Sci., No. 206, 1985

[Saraswat 87] V. A. Saraswat, The Concurrent logic programming CP: definition and operational semantics, Proc. of ACM Symp. on Principles of Programming Languages, 1987

[Shibayama 87] E. Shibayama, A Compositional Semantics of GHC, Proc. of 4th Cof. JSSST, 1987

[Takeuchi 86] A. Takeuchi, Towards a Semantic Model of GHC, Tech. Rep. of IECE, COMP86-59, 1986

[Ueda 88] K. Ueda, Transformation Rules for GHC Programs, to appear in Int. Conf.on Fifth Generation Computer System 1988, 1988

[Ueda 89] K. Ueda, Guarded Horn Clauses, MIT Press, to appear

付録: Brock Ackermannの例題

次の例はの Brock Ackermann の例題 [Brock 81] を GHC で記述したもの [Takeuchi 86] である。ただし以下のプログラムは normal form にはなっていない。

D_{BA} :

```
s(In, Mid, Out)  :- true |
                    dup(In, X), dup(Mid, Y),
                    merge(X, Y, Z), pop(Z, Out).
```

```
dup([A|_], U) :- true | U = [A, A1], A1 = A.
pop([A1, A2|_], U) :- true | U = [A1, B1].

merge([A|Ix], Iy, U) :- true | U = [A|U1],
                                merge(Ix, Iy, U1).
merge(Ix, [A|Iy], U) :- true | U = [A|U1],
                                merge(Ix, Iy, U1).
merge(Ix, [], U) :- true | Ix = U.
merge([], Iy, U) :- true | Iy = U.

inv([A|In], U) :- A = a | U = [B], B = b.
inv([B|In], U) :- B = b | U = [A], A = a.
```

D'_{BA}:

D_{BA} より、pop の定義を次の2行で置き替えたもの。

```
pop([A1|X1], U) :-true | U = [A1|U1],
                         pop1(X1, U1).

pop1([A2|_], U1) :- true | U1 = [A2].
```

□