ICOT Technical Memorandom: TM-1204

TM-1204

並列論理言語における観測プログラムの合成

堀内　謙二

August, 1992

ICOT

Mita Kokusai Bldg. 21F
4-28 Mita 1-Chome
Minato-ku Tokyo 108 Japan

(03)3456-3191～5
Telex ICOT J32964

Institute for New Generation Computer Technology

並行論理型言語における観測プログラムの合成

堀内 謙二

(財) 新世代コンピュータ技術開発機構
東京都港区三田 1 丁目 4-28
horiuchi@icot.or.jp

概要 並行プログラムの動作を考える場合，起動される初期ゴールやその時の環境から決まる一定の挙動や最終的な結果よりむしろ，ゴールと (ともに変化するかも知れない) 環境との間の反応的な動作 (reactive behaivior) の方が重要である．
文献 [2] では，FGHC プログラムのゴールと反応的な動作を行なうプロセスをも FGHC プログラムを用いて記述することにより，反応的な動作を操作的意味論を用いて形式化し，normal な反応的動作を定義している．
本稿では，並行論理型言語である Flat GHC プログラムのゴールを，動作を注目したい対象となるゴール (被観測ゴール) と，それらを観測，制御するゴール (観測ゴール) とに区別し，被観測ゴールと観測ゴールが normal な反応的動作をするかどうかを判定する方法を提案している．また，観測ゴールが部分的にしか与えられていないと場合，残りの観測ゴールの合成方法についても述べる．

1. はじめに

FGHC をはじめとする並行プログラムの動作を考える時，ゴールと (場合によっては実行が進むにつれて変化するかも知れない環境との間の反応的な動作 (reactive behaivior) が重要であり，Prolog などの逐次型のプログラムの場合のように，起動される初期ゴールやその時の環境から決まるような一定の挙動や最終的な結果はむしろあまり重要ではないかもしれない．

初期ゴールで起動される実行の過程で，環境は ゴールやそこから派生するサブゴールによって変化させられる．また，実際に反応的な動作を考えるなら，それらの環境の変化に応じてなにか "外部プロセス" が環境を変化させるかも知れない．例えば，そのプログラムが入出力デバイスを記述しているような場合，"外部プロセス" とは，ユーザなどの人間かもしれないし，他の (デバイスなどを記述している) プログラムかもしれない．文献 [2] では，FGHC プログラムのゴールと反応的な動作を行なう "外部プロセス" も FGHC プログラムを用いて記述することにより，そのような反応的な動作を操作的意味論を用いて形式化し，normal な反応的動作を定義している．

しかし，そこでは FGHC で記述された "外部プロセス"(observer と呼ばれているが) が既に与えられているとして議論されている．また，normal な反応的動作と発散などの制御不能な動作とを区別することは不可欠であると主張しているが，それらを区別する方法には議論は及んでいない．

本稿では，並行論理型言語である Flat GHC プログラムのゴールを，動作を注目したい対象となるゴール (被観測ゴール) と，それらを観測，制御するゴール (観測ゴール) とに区別し，被観測ゴールと観測ゴールが normal な反応的動作をするかどうかを判定する方法を提案している．また，観測ゴールが部分的にしか与えられていないと場合，残りの観測ゴールの合成方法についても述べる．

まず並行論理型言語 Flat GHC とその操作的意味論について述べた後，文献 [2] に従い被観測プログラムと観測プログラムの関係を操作的意味論を基礎の述べる．最後に，normal な反応的動作の判定，観測プログラムの合成について述べる．

2. Flat Guarded Horn Clauses

まず，FGHC について述べる前に，以下で使われる基本的な用語について簡単に定義する．

Var を変数の集合，*Func* を関数記号の集合，*Pred* を述語記号の集合，*Term* を *Var* と *Func* 上で定義されるすべての項の集合，*Atom* を *Term* と *Pred* 上で定義されるすべてのアトムの集合とする．表現は項，アトム，

表現の並びまたは表現の集合や multi 集合であるとする．表現 Exp に現われる全ての変数の集合を $var(Exp)$ で表わす．

2.1 FGHC の構文

FGHC プログラムは flat guarded clause の集合で，flat guarded clause (または単に clause) は，次のような形をしている．

$$p(t_1,\ldots,t_k) :- G_1,\ldots,G_m \mid B_1,\ldots,B_n.$$

ここで，$p \in Pred$ で p は k 引数の述語記号，$t_1,\ldots,t_m \in Term$ で $G_1,\ldots,G_m,B_1,\ldots,B_n \in Atom$ である．また，$G_1,\ldots,G_m$ は guard ゴール，$B_1,\ldots,B_n$ は body ゴールと呼ばれる．アトム $p(t_1,\ldots,t_k)$ は head と呼ばれ，head と guard ゴールとを合わせた部分を guard と，body ゴールの部分は body と呼ばれる．また，2 つの項を unify するための 2 引数の述語記号 "=" のみが built-in として準備されており，"=" によるゴールを unification ゴールと呼ぶ．各 guard ゴール G_i は unification ゴールである．

2.2 Flat GHC の操作的意味論

文献 [2] に従い，FGHC の操作的意味論を transition system を用いて定義している．

FGHC プログラム $Prog$ の transition system を *configuration* と *transition relation* を用いて定義する．configuration はゴールの multi 集合 B と具体化環境 *binding environment*(または，単に環境 (environment) と呼ばれる) Env との対 $\langle B,\ Env\rangle$ である．環境 Env は $var(B) \cup E \subseteq V$ を満たすような変数集合 V 付きの unification アトムの multi 集合 E で，$E{:}V$ で表される．ここで，E はゴール B に関する具体化情報を V はその具体化情報に関わった変数集合を表現している．また以下では，$\models F$ は，F が Clark の equality theory から求まる論理的帰結 (logical consequence) であることを表しており，$\forall(var(F)).F$ や $\exists(var(F)).F$ を $\forall.F$ や $\exists.F$ で表すことにする．任意の unification アトムの集合 E_1, E_2 と変数の集合 V に対して，

$$\models \forall.(E_1 \supset \exists(var(E_2) \setminus V).E_2)$$

が成り立つ時，V の下で E_2 は E_1 より一般的であるといい，$E_2 \succeq_V E_1$ で表す．また，$E_2 \succeq_V E_1$，かつ，$E_1 \succeq_V E_2$ である時，V の下で E_1 は E_2 と等価であるといい，$E_1 \sim_V E_2$ で表す．

例 2.1　変数の集合 $\{X\}$ と unification アトム $X = f(a,B)$ と $X = f(Y,Z)$ に対して，

$$\models \forall XB.(X = f(a,B)) \supset \exists YZ.(X = f(Y,Z))$$

が成り立つので，$X = f(Y,Z) \succeq_{\{X\}} X = f(a,B)$ である．

プログラム $Prog$ における transition relation は以下の configuration 上の 2 項関係 ($\cdot \rightarrow \cdot$ で表すが) で定義される．

$$\frac{\langle B_1,\ E_1{:}V_1\rangle \rightarrow \langle B_1',\ E_1'{:}V_1'\rangle}{\langle B_1 \cup B_2,\ E_1{:}V_1\rangle \rightarrow \langle B_1' \cup B_2,\ E_1'{:}V_1'\rangle} \tag{1}$$

$$\frac{\langle \{A{=}H\} \cup G,\ E{:}V \cup var((H,G))\rangle \xrightarrow{*} \langle \emptyset,\ E \cup E_g{:}V'\rangle}{\langle \{A\},\ E{:}V\rangle \rightarrow \langle B,\ E \cup E_g{:}V' \cup var(B)\rangle}$$

$$\text{もし } \exists c \in Prog\ ((H :- G \mid B) \equiv c\eta \wedge V \cap var(c\eta) = \emptyset) \quad \eta \text{は renaming でかつ } E_g \succeq_{var(A)} E \tag{2}$$

$$\frac{}{\langle s{=}t,\ E{:}V\rangle \rightarrow \langle \emptyset,\ E \cup \{s{=}t\}{:}V\rangle} \tag{3}$$

初期ゴール B のプログラム $Prog$ の下での実行は，$\langle B,\ \emptyset{:}\,var(B)\rangle$ を c_1 とした時，次のような (無限に続くかも知れない)transition の連鎖として表現できる．

$$c_1 \rightarrow c_2 \rightarrow \ldots \rightarrow c_i \rightarrow \ldots$$

また，上記 (1)-(3) のどの規則を用いて得られた transition かを特に区別したい時にはそれぞれに対応して $\xrightarrow{\cup}$，$\xrightarrow{c}$ や $\xrightarrow{=}$ などの記法を用いる．また，$\rightarrow$ の reflexive で transitive な閉包 (closure) を $\xrightarrow{*}$ で表し，$\xrightarrow{c}$(または $\xrightarrow{=}$)

を一回だけ適用し，それ以外は (0 回かも知れないが) $\xrightarrow{u}$ を適用することによって得られた閉包 $\xrightarrow{*}$ を $\stackrel{S}{\Rightarrow}$ (または $\stackrel{\equiv}{\Rightarrow}$) と記述する.

ゴール b と clause c に対して，c の適当な renaming を H :- $G \mid B$ とする. その時，$\{A=H\} \cup G$ を clause c に対するゴール b の最汎環境 (most general environment) と呼び，ゴール b は，その最汎環境で B にリダクションされるという.

引数が相異なる変数である述語 p によるゴール b があり，述語 p は節 $c_1, \ldots, c_n$ によってのみ定義されているとしよう. 各 $c_i (n \geq i \geq 1)$ に対するゴール b の mge を E_i とすると，$i, j (n \geq i \neq j \geq 1)$ に対して，$E_i \not\succeq_{var(b)} E_j$ でかつ $E_j \not\succeq_{var(b)} E_i$ である時，c_i と c_j は排他的であると呼び，p を定義するすべての節がお互いに排他的である時，述語 p は排他的に定義されていると呼ぶ. プログラム $Prog$ に現れるすべての述語が排他的に定義されている時，プログラム $Prog$ は排他的に定義されていると呼ぶ.

上記の定義において，環境の unification アトムの集合に付随している変数の集合は，新しく unification アトムが導入される時に，以前使われた変数と衝突しないようにするために必要である. 以下の議論では，新しく unification アトムが導入される時は変数が衝突しないように注意深く導入されるものとして，変数の集合を省略し，configuration をゴールと unification アトムの対で表すものとする.

3. 観測プログラム

ここでは，FGHC プログラムのゴールと反応的な動作を行なうプロセスを FGHC プログラムを用いて記述することにより，文献 [2] に従い，反応的な動作を 3 節の操作的意味論を用いて形式化し，そのような normal な反応的動作を定義する.

初期 configuration を $\langle P_0 \cup O_0, \emptyset \rangle$ とした時，P_0 を今注目しているゴールとし，O_0 は P_0 を観測しているゴールとする. ここで，$var(P_0) \cap var(O_0) = \overline{X} \neq \emptyset$ とする. この時，P_0 を被観測ゴール，O_0 を観測ゴールと呼ぶ. 何回かの transition の後，現在，$\langle P \cup O,\ E_P \cup E_O \rangle$ に至っているとする. ここで，E_P, E_O はそれぞれ，P_0 または O_0 によって生成された環境である. その時，以下のような transition を考える.

(1) 環境 E_P に関わらず (空かも知れない) $\langle P \cup O,\ E_P \cup E_O \rangle \xrightarrow{*} \langle P \cup O',\ E_P \cup E_O \cup \alpha \rangle$ が起こる. ここで，α は E_O と矛盾しない.

(2) (空かも知れない) $\langle P \cup O',\ E_P \cup E_O \cup \alpha \rangle \xrightarrow{*} \langle P' \cup O',\ E'_P \cup E_O \cup \alpha \rangle$ が起こる.

(3) 空ではない transition

$\langle P' \cup O',\ E'_P \cup E_O \cup \alpha \rangle \rightarrow$

$\langle P' \cup O'',\ E'_P \cup E_O \cup \alpha \cup \beta_1 \rangle$,

...

$\langle P' \cup O''',\ E'_P \cup E_O \cup \alpha \cup \beta_1 \cup \ldots \cup \beta_n \rangle$

が起こる. ここで，上記の一連の transition の内 $\langle G, E \rangle \stackrel{\equiv}{\rightarrow} \langle G',\ E \cup \beta_i \rangle$ の形の transition に対して，$\models \forall.(E \supset \delta\overline{X}.(E \cup \beta_i))$ である. すなわち，観測ゴールによって生成される環境は $\overline{X}$ を通して被観測ゴールに影響を与えることはない. また，$\not\models \forall.(E_O \cup \alpha \supset \delta\overline{X}.(E_O \cup \alpha \cup \beta_1 \cup \ldots \cup \beta_n))$ である. すなわち，観測ゴールによって生成された (つまり，観測された) 環境は $\overline{X}$ に関して，なにかしら新しい情報である.

この時，観測ゴール O_0 は，被観測ゴール P_0 との normal transaction (α, β) を観測したと言う.

4. FGHC ゴールの拡張リダクションと観測プログラムの合成

ここでは，3 節で導入した 観測ゴール，被観測ゴールを含む拡張されたゴールに対する拡張されたリダクション規則について述べる.

P を被観測ゴール，O を観測ゴールとし，E_P, E_O をそれぞれ unification アトムの集合とし，$ob(\overline{V})$ は $ob \notin Pred$ なる述語 ob からなり，引数が $\overline{V} = var(P) \cup var(O)$ なるゴールである. この時，$\langle P,\ E_P \rangle \blacktriangleright \langle O, ob(\overline{V}),\ E_O \rangle$ を拡張ゴールと呼ぶ.

例 4.1　以下のプログラムを $Prog$ とする.

```
pro([a|L1], M) :- | M=[b|M1], pro(L1,M1).
ob1(L) :- | L=[A|L1], ob1(L1).
```

```
ob2([A|M1]) :- | ob2(M1).
```

この時，

$$\langle \texttt{pro(X,Y)}, \emptyset\rangle \blacktriangleright \langle \texttt{ob1(X1),X=[A|X1],ob2(Y)}, ob\texttt{(X,Y,A,X1)}, \emptyset\rangle$$

は，拡張ゴールである．これを，G_0 とする．

$\langle P, E_P\rangle \blacktriangleright \langle O, ob(\overline{V}), E_O\rangle$ なる拡張ゴールと節の集合 Δ に対して，以下の拡張リダクション規則のいづれかが適用でき，規則によっては Δ は変更される．

(1) **(unification アトム展開)** もし，unification アトム U が P(または，O) に現れたとき，$\langle P, E_P\rangle \blacktriangleright \langle O, ob(\overline{V}), E_O\rangle$ は $\langle P\backslash\{U\}, E_P\cup\{U\}\rangle \blacktriangleright \langle O, ob'(\overline{V}\cup var(U)), E_O\rangle$ (または，$\langle P, E_P\rangle \blacktriangleright \langle O\backslash\{U\}, ob'(\overline{V}\cup var(U)), E_O\cup\{U\}\rangle$) にリダクションされる．$\Delta := \Delta\cup\{ob(\overline{V})\ \texttt{:-}\ |ob'(\overline{V}\cup var(U))\}$ である．

例 4.2 再び 例 4.1 のプログラム $Prog$ を考える．$\Delta = \emptyset$ とする．拡張ゴール G_0 と Δ に対して，unification アトム展開の規則が適用でき，G_0 は，

$$\langle \texttt{pro(X,Y)}, \emptyset\rangle \blacktriangleright \langle \texttt{ob1(X1),ob2(Y)}, ob_1\texttt{(X,Y,A,X1)}, \{\texttt{X=[A|X1]}\}\rangle$$

にリダクションされ，この拡張ゴールを G_1 とする．$\Delta = \{ob\texttt{(X,Y,A,X1)}\ \texttt{:-}\ |\ ob_1\texttt{(X,Y,A,X1)}\}$ である．

(2) **(被観測ゴールの guard 展開)** もし，E_c が節 c に対するゴール $b(\in P)$ の最汎環境で，b が B にリダクションされ，E_c が $E_P\cup E_O$ と矛盾しないとし，$E_P\cup E_O \sim_{\overline{V}} E'\cup E'_c$ なる E', E'_c を考える．ここでは，E' は $var(B)$ を具体化しない最も大きな unification アトムの集合である．その時，
   (a) $E_c \succeq_{\overline{V}} E'_c$ ならば，$\langle P, E_P\rangle \blacktriangleright \langle O, ob(\overline{V}), E_O\rangle$ は
   $\langle P\setminus\{b\}\cup\{B\}, E_P\cup E_c\rangle \blacktriangleright \langle O, ob(\overline{V}), E_O\rangle$ にリダクションされ，Δ は変更されない．あるいは，
   (b) $E'_c \succeq_{\overline{V}} E_c$，すなわち，$E'_c\cup E''_c \sim_{\overline{V}} E_c$ ならば，$\langle P, E_P\rangle \blacktriangleright \langle O, ob(\overline{V}), E_O\rangle$ は
   $\langle P\setminus\{b\}\cup\{B\}, E_P\rangle \blacktriangleright \langle O, ob'(\overline{V'}), E_O\cup E''_c\rangle$ にリダクションされ，
   $\Delta := \Delta\cup\{ob(\overline{V})\ \texttt{:-}\ |E''_c, ob'(\overline{V'})\}$ である．ここで，$\overline{V'} = \overline{V}\cup var(E''_c)$ である．

例 4.3 例 4.2 の拡張ゴール G_1 には，被観測ゴールの guard 展開が適用でき，G_1 は，

$$\langle \texttt{pro(X1,Y1),Y=[b|Y1]}, \emptyset\rangle \blacktriangleright \langle \texttt{ob1(X1),ob2(Y)}, ob_2\texttt{(X,Y,A,X1)}, \{\texttt{X=[A|X1], A=a}\}\rangle$$

にリダクションされ，この拡張ゴールを G_2 とする．

$$\Delta := \Delta\cup\{ob_1\texttt{(X,Y,A,X1)}\ \texttt{:-}\ |\ \texttt{A=a},\ ob_2\texttt{(X,Y,A,X1)}\}$$

である．

(3) **(観測ゴールの guard 展開)** もし，E_c が節 c の適当な renaming に対するゴール $b(\in O)$ の最汎環境で，b が B にリダクションされ，E_c が $E_P\cup E_O$ と矛盾しないとし，$E_P\cup E_O \sim_{\overline{V}} E'\cup E'_c$ なる E', E'_c を考える．E' は $var(B)$ を具体化しない最も大きな unification アトムの集合である．その時，$E_c \succeq_{\overline{V}} E'_c$ ならば，
   $\langle P, E_P\rangle \blacktriangleright \langle O, ob(\overline{V}), E_O\rangle$ は $\langle P, E_P\rangle \blacktriangleright \langle O\setminus\{b\}\cup\{B\}, ob'(V\cup var(E_c)), E_O\cup E_c\rangle$ にリダクションされ，
   $\Delta := \Delta\cup\{ob(\overline{V})\ \texttt{:-}\ E_c\ |\ ob'(\overline{V}\cup var(E_c))\}$ である．

例 4.4 例 4.3 の拡張ゴール G_2 には，unification アトム展開が適用でき，G_2 は，

$$\langle \texttt{pro(X1,Y1)}, \{\texttt{Y=[b|Y1]}\}\rangle \blacktriangleright \langle \texttt{ob1(X1),ob2(Y)}, ob_3\texttt{(X,Y,A,X1,Y1)}, \{\texttt{X=[A|X1], A=a}\}\rangle$$

にリダクションされ，この拡張ゴールを G_3 とすると，

$$\Delta = \Delta\cup\{ob_2\texttt{(X,Y,A,X1)}\ \texttt{:-}\ |\ ob_3\texttt{(X,Y,A,X1,Y1)}\}$$

である．また，この G_3 には，観測ゴールの guard 展開が適用でき，G_3 は，

$$\langle \texttt{pro(X1,Y1)}, \{\texttt{Y=[b|Y1]}\}\rangle \blacktriangleright \langle \texttt{ob1(X1),ob2(Y1)}, ob_4\texttt{(X,Y,A,X1,Y1)}, \{\texttt{X=[A|X1],A=a,B=b}\}\rangle$$

にリダクションされる．この拡張ゴールを G_4 とする．

$$\Delta = \Delta\cup\{ob_3\texttt{(X,Y,A,X1,Y1)}\ \texttt{:-}\ \texttt{Y=[b|Y1]}\ |\ ob_4\texttt{(X,Y,A,X1,Y1)}\}$$

である． G_4 には再び ob1 に関して観測ゴールの guard 展開が適用でき， G_4 は，

$\langle \mathtt{pro(X1,Y1)}, \{\mathtt{Y=[b|Y1]}\}\rangle \blacktriangleright \langle \mathtt{ob1(X2),X1=[A2|X2],ob2(Y1)},ob_5\mathtt{(X,Y,A,X1,Y1,A2,X2)}, \{\mathtt{X=[A|X1],A=a,B=b}\}\rangle$

にリダクションされる．この拡張ゴールを G_5 とする．

$$\Delta = \Delta \cup \{ob_4\mathtt{(X,Y,A,X1,Y1)\ :-\ |\ }ob_5\mathtt{(X,Y,A,X1,Y1,A2,X2)}\}$$

である．

根ノードのラベルが $\langle P, E_P\rangle \blacktriangleright \langle O, E_O\rangle$ で各ノードのラベルが拡張ゴールであるような木構造で，各親ノードをラベル付けしている拡張ゴールがその子ノードをラベル付けしている拡張ゴールに上記の拡張リダクション規則のいずれかを用いてリダクションされる時，そのような木を $\langle P, E_P\rangle \blacktriangleright \langle O, E_O\rangle$ の拡張リダクション木と呼ぶ．また，ラベルが $\langle \emptyset \blacktriangleright O, E_P \blacktriangleright E_O\rangle$ である時，このようなノードを成功ノードと呼ぶ．

次に，環境からゴールに関係のない unification アトムを取り除く操作を定義する． unification アトム U とゴール G に対して， (1) $var(U) \cap var(G) \neq \emptyset$，または， (2) G に関係のある U' に対して， $var(U) \cap var(U') \neq \emptyset$ である時， U は G に関係があるという．また， unification アトムの集合 C から， G に関係のある unification アトムをすべて取り除いたような unification アトムの集合を， $C\downarrow_G$ で表す．また，観測ゴール O ないのゴール $ob(\overline{V})$ の引数 $\overline{V}$ から $O \cup P$ に関係のない変数を取り除いたゴールを $O\downarrow$ と表す．その時，
$\langle P, E_P\downarrow_P\rangle \blacktriangleright \langle O\downarrow, E_O\downarrow_O\rangle$ を $\langle P, E_P\rangle \blacktriangleright \langle O, E_O\rangle\downarrow$ で表す．

(4) **(ゴール畳み込み)**　拡張リダクション木のあるパス上のノードのラベルを $C_1, \ldots, C_n$ とする． C_1 から C_n までの変換において，必ず，上記の (1) (3) の拡張リダクションがそれぞれ少なくとも一度は行なわれており，観測ゴールの述語以外は $(C_1\downarrow)\eta = C_n\downarrow$ で η が renaming であるならば， C_n は C_1 に畳み込まれるといい， C_n のノードを畳み込まれたノードと呼ぶ．また， C_1, C_n の観測ゴールをそれぞれ $ob_1(\overline{X}_1), ob_n(\overline{X}_n)$ とすると，

$$\Delta := \Delta \cup \{ob_n(\overline{X}_n)\ :-\ |\ ob_1(\overline{X}_1\eta)\}$$

である．

例 4.5　例 4.4 の拡張ゴール G_4

$\langle \mathtt{pro(X1,Y1)}, \{\mathtt{Y=[b|Y1]}\}\rangle \blacktriangleright \langle \mathtt{ob1(X2),X1=[A2|X2],ob2(Y1)},ob_5\mathtt{(X,Y,A,X1,Y1,A2,X2)}, \{\mathtt{X=[A|X1],A=a,B=b}\}\rangle$

は， G_0

$$\langle \mathtt{pro(X,Y)}, \emptyset\rangle \blacktriangleright \langle \mathtt{ob1(X1),X=[A|X1],ob2(Y)},ob\mathtt{(X,Y,A,X1)}, \emptyset\rangle$$

に畳み込まれる．

$$\Delta = \Delta \cup \{ob_5\mathtt{(X,Y,A,X1,Y1,A2,X2)\ :-\ |\ }ob\mathtt{(X1,Y1,A2,X2)}\}$$

である．

例 4.6　例 4.1 4.5 の拡張リダクションの過程で以下のような述語 ob を定義する観測プログラムが生成された．

$ob\mathtt{(X,Y,A,X1)\ :-\ |\ }ob_1\mathtt{(X,Y,A,X1)}.$
$ob_1\mathtt{(X,Y,A,X1)\ :-\ |\ A=a,\ }ob_2\mathtt{(X,Y,A,X1)}.$
$ob_2\mathtt{(X,Y,A,X1)\ :-\ |\ }ob_3\mathtt{(X,Y,A,X1,Y1)}.$
$ob_3\mathtt{(X,Y,A,X1,Y1)\ :-\ Y=[b|Y1]\ |\ }ob_4\mathtt{(X,Y,A,X1,Y1)}.$
$ob_4\mathtt{(X,Y,A,X1,Y1)\ :-\ |\ }ob_5\mathtt{(X,Y,A,X1,Y1,A2,X2)}.$
$ob_5\mathtt{(X,Y,A,X1,Y1,A2,X2)\ :-\ |\ }ob\mathtt{(X1,Y1,A2,X2)}.$

$\langle P \blacktriangleright O, E_P \blacktriangleright E_O\rangle$ の拡張リダクション木のすべての葉ノードが，成功ノードか，または，畳み込まれたノードならば，その拡張リダクション木は normal に閉じているという．

$\Delta = \emptyset$ とし normal に閉じている C_0 の拡張リダクション木のすべての親ノード，子ノードの間で行なわれた各変換ステップ毎に，上記の規則に基づいて生成された節の集合 Δ を C_0 から生成された観測プログラムと呼ぶ．

X

命題: $\langle P, E_P \rangle \blacktriangleright \langle O, ob(\overline{V}), E_O \rangle$ の拡張リダクション木が normal に閉じているならば，観測ゴール $O, ob(\overline{V})$ が観測する被観測ゴール P との transaction がすべて normal transaction であるような観測ゴール $ob(\overline{V})$ が存在し，述語 ob を定義しているのは $\langle P, E_P \rangle \blacktriangleright \langle O, ob(\overline{V}), E_O \rangle$ から生成された観測プログラムである.

おわりに

本稿では，並行論理型言語である Flat GHC プログラムのゴールを，動作を注目したい対象となるゴール (被観測ゴール) と，それらを観測，制御するゴール (観測ゴール) とに区別し，被観測ゴールと観測ゴールが normal な反応的動作をするかどうかを判定する方法を提案し，また，観測ゴールが部分的にしか与えられていないと場合，残りの観測ゴールの合成方法についても述べた.

参考文献

[1] Ueda, K. and K. Furukawa, "Transformation Rules for GHC Programs", Proc. of the International Conference on FGCS'88, pp.582–591, Tokyo, 1988.

[2] Ueda, K., "Designing a Concurrent Programming Language", Proc. of an International Conference organized by the IPSJ to Commemorate the 30th Anniversary: InfoJapan'90, pp.87–94, Tokyo, 1990.