関係代数によるUNITYループの意味づけ

徐 良為[†] 武市正人[††] 岩崎英哉[†††]

関係は一対多の対応を記述するものである．関係代数はプログラム，特に，ある入力値に対する可能な出力値が複数あるような非決定的なプログラムを記述する数学モデルとして重要な役割を果たしてきた．一方，UNITY（Unbounded Nondeterministic Iterative Transformations）は並列計算プログラムの設計，検証に有用なアプローチとして知られている．従来，UNITYに対する意味定義，推論は一種の時制論理（Temporal Logic）を基本としてきたが，その記法はvariable-wiseスタイルであって簡潔さに欠ける．本論文では，関係代数により抽象化された局所エンゼリック選択（locally angelic choice）の定義を利用して，UNITYループに対して形式的な意味定義を与える．さらに，UNITYループ固有の性質を関係代数を用いて推論し証明する方法を，簡単な例とともに示す．

Relational Definition of UNITY Loop

LIANGWEI XU,[†] MASATO TAKEICHI[††] and HIDEYA IWASAKI[†††]

Relation is a set of pairs which is suitable for describing one-input-multiple-output correspondences. Relational algebra as a mathematical model has been playing important role in specification of nondeterministic programs. UNITY (Unbounded Nondeterministic Iterative Transformations), on the other hand, is an approach to specification, design and verification of parallel programs. UNITY loop is usually reasoned by some kind of temporal logic which is in variable-wise style. In this paper, we give a formal definition of the UNITY loop with the help of the relational definition of the locally angelic choice. The purpose of this paper is to make a variable-free definition and give a concise proof tool on UNITY loop by means of relations.

1. はじめに

局所エンゼリック（locally angelic）選択は，非決定的なプログラムの設計にきわめて有用な概念といわれている．局所エンゼリック選択（$\Diamond$）は，与えられた2つの式の中から1つを非決定的に選んで評価する．もし，片方の式の評価が停止しなければ，もう片方を選ぶ．

$$
\begin{aligned}
&\mathsf{e}_1 \Diamond \mathsf{e}_2 \\
&\quad = \text{either } \mathsf{e}_1 \text{ or } \mathsf{e}_2 && \textbf{if } \mathsf{e}_1 \neq \bot \wedge \mathsf{e}_2 \neq \bot \\
&\quad = \mathsf{e}_1 && \textbf{if } \mathsf{e}_2 = \bot \\
&\quad = \mathsf{e}_2 && \textbf{if } \mathsf{e}_1 = \bot
\end{aligned}
$$

ここで $\bot$ は評価が停止しないことを指す．この定義はMcCarthyのambiguity演算子[6] *amb* としてよく知られている．この1つの実装として e_1 と e_2 の両方を並列に評価し，先に正常に終わるものを解として返す方法がある．本論文では $\Diamond$ を次のように，2つの関数 f，g の間の選択に抽象化する．

$$\mathsf{f} \,\square\, \mathsf{g} = \lambda x.\ (\mathsf{f}\,x) \Diamond (\mathsf{g}\,x)$$

選択 $\square$ を含むようなプログラムの関係代数を用いた意味づけはXu[9]に述べてある．その基本的な考え方はfを非決定的なプログラムとし，fの意味を2つの関係 $\mathcal{A}[\![\mathsf{f}]\!]$ と $\mathcal{D}[\![\mathsf{f}]\!]$ で表すというものである．これらは，入力 x と，それに対する出力 y の対 (y,x) の集合（関係）で，以下のように構成される．「$(y,x) \in \mathcal{A}[\![\mathsf{f}]\!]$」は，入力 x に対して y が $\mathsf{f}x$ を実行した後の可能な停止状態であるような y と x の対である．一方，$(y,x) \in \mathcal{D}[\![\mathsf{f}]\!]$ は，y は入力 x に対する $\mathsf{f}x$ の実行後の可能な停止状態であるが $\mathsf{f}x$ がどのような計算経路をたどっても必ず停止するような y と x の対である．$\square$ の意味は次のように定義される．

$$
\begin{aligned}
&(y,x) \in \mathcal{A}[\![\mathsf{f} \,\square\, \mathsf{g}]\!] \\
&\quad \equiv (y,x) \in \mathcal{A}[\![\mathsf{f}]\!] \vee (y,x) \in \mathcal{A}[\![\mathsf{g}]\!]
\end{aligned}
$$

† 株式会社数理システム
Mathematical Systems Inc.
†† 東京大学大学院工学系研究科
Faculty of Engineering, University of Tokyo
††† 東京農工大学工学部
Faculty of Technology, Tokyo University of Agriculture and Technology

$(y,x) \in \mathcal{D}[\![\mathsf{f} \,\square\, \mathsf{g}]\!]$
$\equiv (y,x) \in \mathcal{A}[\![\mathsf{f} \,\square\, \mathsf{g}]\!] \wedge$
$x \in dom(\mathcal{D}[\![\mathsf{f}]\!]) \cup dom(\mathcal{D}[\![\mathsf{g}]\!])$

ただし，$dom()$ は関係の定義域を表す．$\mathcal{A}[\![\mathsf{f}]\!]$ を f のエンゼリック（angelic）側面，$\mathcal{D}[\![\mathsf{f}]\!]$ を f のデモニック（demonic）側面と呼ぶ．明らかに，$\mathcal{D}[\![\mathsf{f}]\!]$ は $\mathcal{A}[\![\mathsf{f}]\!]$ の部分集合である．また，すべての x に対して $\mathsf{f}x$ の結果が唯一（f が関数）であれば，$\mathcal{A}[\![\mathsf{f}]\!] = \mathcal{D}[\![\mathsf{f}]\!]$ となる．$\mathcal{A}$ と $\mathcal{D}$ の違いを次の例で説明する．$\mathsf{f}x = 1$，$\mathsf{g}x = 0$ は整数上の定数関数，$(1/)$ を $(1/)x = 1/x$ となるような関数，$\circ$ を関数合成とする．$\mathsf{h} = (1/) \circ (\mathsf{f} \,\square\, \mathsf{g})$ の2つの側面 $\mathcal{A}[\![\mathsf{h}]\!]$ と $\mathcal{D}[\![\mathsf{h}]\!]$ は次のように定義される．

$\mathcal{A}[\![\mathsf{h}]\!] = \{\ (1,n) \mid n : \text{整数}\ \}$
$\mathcal{D}[\![\mathsf{h}]\!] = \{\ \}$

$(1,n) \in \mathcal{A}[\![\mathsf{h}]\!]$ であることは，任意の入力 n に対して1が $\mathsf{h}n$ の可能な解であることを表している．$\mathcal{D}[\![\mathsf{h}]\!] = \{\}$ であることは，いかなる入力 n に対しても $\mathsf{h}n$ は異常終了（0による割算）する可能性があることを表している．

本論文は以下のことを目的とする．

- 現実的な問題（UNITY）に対する具体的な適用例を提示し，文献9)の手法が有効であることを示すこと．
- UNITYは並列プログラムの仕様・設計と検証のために作られた代表的な計算モデルである．従来UNITYのような非決定的なプログラムの意味定義・推論（証明）は一種の時制論理（Temporal Logic）を用いて行われてきた[4]が，本論文で関係代数を用いて具体的に記述することにより，互いの関連を明らかにすること．

関係代数を用いることによってUNITYループの定義が関数レベルに抽象化され，変数のない（variable-free）推論[1]が可能となることによって，より簡潔になる．

本論文の構成は以下のとおり．関係代数を簡単に紹介したあと非決定的な基本的な言語を定義する．それを用いてUNITYループを定義し，その意味と代数的性質を示し，簡単なUNITYループの証明例をあげる．最後はまとめと関連研究を述べる．

2. 関係代数

2.1 基本定義

$\mathcal{U}$ を普遍的（universal）な集合とする．$\mathcal{U}$ 上の二項関係は，直積 $\mathcal{R} = \mathcal{U} \times \mathcal{U}$ の部分集合である．$\mathcal{U}$ 上のすべての関係は完備束（complete lattice）となり，集合の包含関係 $\subseteq$ は束の順序関係となる．関係の和 $\cup$ と積 $\cap$ はそれぞれ最小上限と最大下限となる．束の最大要素 $\top$ は全域関係 $\mathcal{U} \times \mathcal{U}$ 自身であり，最小要素は空関係 $\emptyset$ である．恒等関係は $I = \{(x,x) \mid x \in \mathcal{U}\}$ と定義される．

関係 R に対して，$(b,a) \in R$ を bRa と略記する．ここで a は関係 R の入力とし b はその出力とする．関係 R の補関係は $\tilde{R} = \{(y,x) \mid \neg(yRx)\}$ と定義し，逆関係は $R^\circ = \{(y,x) \mid xRy\}$ と定義する．関係合成には通常の関係合成とデモニック合成の2種類がある．

関係 R と S に対して，通常の関係合成 $(R \bullet S)$ は次のように定義される．

$$\forall a,c \in \mathcal{U}:\ a(R \bullet S)\,c \equiv \exists b \in \mathcal{U}:\ aRb \wedge bSc$$

合成演算子 $\bullet$ は結合律と $\cup$ に対する分配律を満たす．また，デモニック合成の定義はモノタイプを定義したあとに述べる．

関係 f がもし単純 $(f \bullet f^\circ \subseteq I)$ かつ全域定義 $(I \subseteq f^\circ \bullet f)$ なら f を関数と呼ぶ．関数は単純であるので関数適用 $f\,x$ は唯一に定まる．

関係 A はもし $A \subseteq I$ を満たすならばモノタイプ（monotype[1]）と呼ばれる．任意のモノタイプ $A, B \subseteq I$ と関係 R は，次の性質を満す．

$$A \cap B = A \bullet B \tag{1}$$
$$A \bullet B = B \bullet A \tag{2}$$
$$R \bullet A \subseteq R \tag{3}$$
$$A \bullet R \subseteq R \tag{4}$$
$$A \bullet A = A \tag{5}$$

モノタイプと述語の間には一対一の対応がある．述語 p に対応するモノタイプを $p? = \{(x,x) \mid px\}$ と定義する．モノタイプ A の否定はモノタイプであって $\overline{A} = \tilde{A} \cap I$ である．

関係 R の定義域（$R\square = \{\ (x,x) \mid \exists y : yRx\ \}$）と値域（$\square R = \{\ (y,y) \mid \exists x : yRx\ \}$）はモノタイプである．関係の定義域と値域は次のような性質を持つ．

$$R = R \bullet R\square = \square R \bullet R \tag{6}$$
$$\square(R \cup S) = \square R \cup \square S \tag{7}$$
$$(R \bullet S)\square = (R\square \bullet S)\square \tag{8}$$
$$\square(R \bullet S) = \square(R \bullet \square S) \tag{9}$$

2つの関係 R と S のデモニック合成（$\circledcirc$）は次のように定義される．

$$R \circledcirc S = R \bullet S \bullet (R \triangleright S)$$
$$R \triangleright S = \bigcup \{\ A \subseteq I \mid \square(S \bullet A) \subseteq R\square\ \}$$

$\triangleright$ はデモニック制限と呼ばれ，次のような性質を持つようなモノタイプである．

$$\forall A \subseteq I:\ \ A \subseteq R \triangleright S \equiv \square(S \bullet A) \subseteq R\square$$

関係のデモニック合成は次のような性質を持つ．

$$(R \circledcirc S)\Box = S\Box \bullet (R \triangleright S) \quad (10)$$
$$R \circledcirc S = R \bullet S \bullet (R \circledcirc S)\Box \quad (11)$$
$$(R \bullet S = R \circledcirc S) \Leftarrow (R\Box \supseteq \Box S) \quad (12)$$
$$R \bullet P = R \circledcirc P \quad (P \text{ は単純の場合}) \quad (13)$$
$$R = I \circledcirc R = R \circledcirc I \quad (14)$$
$$(R \circledcirc S) \circledcirc T = R \circledcirc (S \circledcirc T) \quad (15)$$
$$(R \triangleright S) \triangleright T = R \triangleright (S \bullet T) \quad (16)$$

関係上の条件式 $(R \rightarrow S, T)$ を次のように定義する．

$$R \rightarrow S, T = U \cup V$$
$$U = S \bullet ((true =)? \bullet R)\Box$$
$$V = T \bullet ((false =)? \bullet R)\Box$$

2.2 再帰定義と不動点

関係上の関数 Φ は，次のような性質を満たすとき，順序 $\leq$ に対して単調であるという．

$$\forall R, S \in \mathcal{R}:\ S \leq R \Rightarrow \Phi(S) \leq \Phi(R)$$

今後，関数 Φ がある順序 $\leq$ に対して単調であるということを $\leq$-単調と書く．たとえば，デモニック制限は次のような性質を満たす．

関数 $\lambda X.\ X \triangleright R$ は $\subseteq$-単調 (17)

関数 $\lambda X R \triangleright X$ は $\subseteq$-逆単調 (18)

完備束のうえでは，Tarski 定理により，Φ が $\leq$ に対して単調である場合，$\Phi(X) \leq X$ に最小解が存在する．我々はそのような最小解を $\mu_{\leq} X.\ \Phi(X)$ と記す．ここでいう最小の意味は次のようなものである．

$$\forall Y:\ \Phi(Y) \leq Y \Rightarrow \mu_{\leq} X.\Phi(X) \leq Y$$

2.3 Preorder 閉包

関係 S がもし $(I \subseteq S)$ と $(S \bullet S \subseteq S)$ を満たすなら S を Preorder と呼ぶ．任意の関係 R に対して R を含む最小の Preorder R^* が存在し，それは次のように定義される：

$$R^* = \mu_{\subseteq} X.I \cup X \bullet R$$

R^* を R の Preorder 閉包（closure）と呼ぶ．Backhouse ら[1]によると，R^* は次のような性質を持つ．

$$S \bullet R^* = \mu_{\subseteq} X.\ S \cup X \bullet R \quad (19)$$
$$R^* \bullet S = \mu_{\subseteq} X.\ S \cup R \bullet X \quad (20)$$
$$(R^*)^* = R^* = R^* \bullet R^* \quad (21)$$
$$R^* \bullet R = R \bullet R^* \subseteq R^* \quad (22)$$
$$\forall n \geq 0:\ R^n \subseteq R^* \quad (23)$$
$$R \subseteq S \Rightarrow R^* \subseteq S^* \quad (24)$$

ここの R^n は次のように定義される．

$$R^0 = I$$
$$R^1 = R$$
$$R^{n+1} = R \bullet R^n$$

2.4 デモニック合成とその単調順序

再帰的に定義されたプログラムの不動点の存在を保証するにはプログラム中に出現する演算子のデモニック側面が単調であるような順序関係が必要になる．

定義 1 任意の関係 $R, S \in \mathcal{R}$ の間に，次のような関係 $\sqsubseteq$ を定義する．

$$R \sqsubseteq S \equiv R \subseteq S \wedge S \bullet R\Box \subseteq R$$

$\sqsubseteq$ は $\mathcal{R}$ 上の順序関係であり，次のような定理が成り立つ．その証明は Xu[9]を参照されたい．

定理 1 $(\mathcal{R}, \sqsubseteq)$ は CPO である．

3. 簡単な言語

本章では UNITY ループを記述するために簡単な言語を定義する．

3.1 文 法

この言語は決定的な関数（単純関係）とその間のエンゼリック選択，合成，条件式および再帰定義で構成される．

E	::=	D	(単純 (simple) 関係)
	|	F	(変数)
	|	E ∘ E	(合成)
	|	E→E, E	(条件式)
	|	E ▯ E	(エンゼリック選択)
	|	μF.E	(再帰定義)

ここの $\mathsf{p} \rightarrow \mathsf{f}, \mathsf{g}$ は条件式を表し，ある入力 a に対して $\mathsf{p}a$ が真であるなら $\mathsf{f}a$ を実行し，そうでなければ，$\mathsf{g}a$ を実行する．

混乱を避けるため，この文法によるプログラムはサンセリフフォントで（E のように），関係はイタリックフォントで（R のように）記す．

3.2 意味定義

我々は非決定的なプログラムの意味を記述するために 2 つの意味関数[9]：エンゼリック意味関数とデモニック意味関数を使う．エンゼリック意味関数を $\mathcal{A}[\![\_]\!] \in \mathcal{R} \leftarrow Env \leftarrow \mathsf{E}$ と，デモニック意味関数を $\mathcal{D}[\![\_]\!] \in \mathcal{R} \leftarrow Env \leftarrow Env \leftarrow \mathsf{E}$ と記述する．ここで，Env は環境，すなわち，変数と値（関係）の組の集合を表す．また我々はプリミティブ関数（組込み関数など）に対する意味関数がすでに存在する（$\mathcal{I}[\![\_]\!]$ と記す）と仮定する．$\mathcal{I}[\![\_]\!]$ はプリミティブ関数を入力とし $\mathcal{U}$ 上の単純関係を返すような関数である．たとえば，恒等関数 I に対しては $\mathcal{I}[\![\mathsf{I}]\!] = I$ と定義される．以後，簡単のため $\mathcal{I}[\![\mathsf{f}]\!]$ を同じ文字のイタリックフォントを用いて f と表すことにする．意味関数の具体的な定義とその正当性の証明などは Xu ら[9]を参照されたい．ここでは，定義についてのみ述べる．

次の定義の中の関係 $\mathcal{A}[\![\mathsf{e}_1]\!]\sigma$，$\mathcal{A}[\![\mathsf{e}_2]\!]\sigma$ と $\mathcal{A}[\![\mathsf{e}_3]\!]\sigma$

を，それぞれ R_1，R_2 と R_3 と，関係 $\mathcal{D}[\![\mathsf{e}_1]\!]\sigma\rho$，$\mathcal{D}[\![\mathsf{e}_2]\!]\sigma\rho$ と $\mathcal{D}[\![\mathsf{e}_3]\!]\sigma\rho$ を，それぞれ S_1，S_2 と S_3 と略記する．

(a1) $\mathcal{A}[\![\mathsf{d}]\!]\sigma = \mathcal{I}[\![\mathsf{d}]\!] = d \quad (\mathsf{d} \in \mathsf{D})$

(a2) $\mathcal{A}[\![\mathsf{f}]\!]\sigma = \sigma[\![\mathsf{f}]\!] \quad (\mathsf{f} \in \mathsf{F})$

(a3) $\mathcal{A}[\![\mathsf{e}_1 \circ \mathsf{e}_2]\!]\sigma = R_1 \bullet R_2$

(a4) $\mathcal{A}[\![\mathsf{e}_1 \to \mathsf{e}_2, \mathsf{e}_3]\!]\sigma = R_1 \to R_2, R_3$

(a5) $\mathcal{A}[\![\mathsf{e}_1 \,[]\, \mathsf{e}_2]\!]\sigma = R_1 \cup R_2$

(a6) $\mathcal{A}[\![\mu\mathsf{f}.\mathsf{e}]\!]\sigma = \mu_{\subseteq} X.\ \mathcal{A}[\![\mathsf{e}]\!](\sigma \cup \{(\mathsf{f}, X)\})$

(d1) $\mathcal{D}[\![\mathsf{d}]\!]\sigma\rho = \mathcal{I}[\![\mathsf{d}]\!] = d \quad (\mathsf{d} \in \mathsf{D})$

(d2) $\mathcal{D}[\![\mathsf{f}]\!]\sigma\rho = \rho[\![\mathsf{f}]\!] \quad (\mathsf{f} \in \mathsf{F})$

(d3) $\mathcal{D}[\![\mathsf{e}_1 \circ \mathsf{e}_2]\!]\sigma\rho = S_1 \circledcirc S_2$

(d4) $\mathcal{D}[\![\mathsf{e}_1 \to \mathsf{e}_2, \mathsf{e}_3]\!]\sigma\rho = (\mathcal{A}[\![\mathsf{e}_1 \to \mathsf{e}_2, \mathsf{e}_3]\!]\sigma) \bullet (S_1 \to S_2\Box, S_3\Box)$

(d5) $\mathcal{D}[\![\mathsf{e}_1 \,[]\, \mathsf{e}_2]\!]\sigma\rho = (\mathcal{A}[\![\mathsf{e}_1 \,[]\, \mathsf{e}_2]\!]\sigma) \bullet (S_1\Box \cup S_2\Box)$

(d6) $\mathcal{D}[\![\mu\mathsf{f}.\mathsf{e}]\!]\sigma\rho = \mu_{\sqsubseteq} Y.\ \mathcal{D}[\![\mathsf{e}]\!](\sigma \cup \{(\mathsf{f}, \mathcal{A}[\![\mu\mathsf{f}.\mathsf{e}]\!]\sigma)\})(\rho \cup \{(\mathsf{f}, Y)\})$

3.3 エンゼリック側面とデモニック側面との関係

エンゼリック側面とデモニック側面は次のような定理[9]で結ばれる．

定理 2 任意のプログラム e に対して次のような関係が成り立つ．

$$\mathcal{D}[\![\mathsf{e}]\!] \sqsubseteq \mathcal{A}[\![\mathsf{e}]\!]$$

$\mathcal{A}[\![\_]\!]$ と $\mathcal{D}[\![\_]\!]$ を用いてプログラムの停止性を検証することができる．プログラム e と入力値 x に対し：

- $(x,x) \notin \mathcal{A}[\![\mathsf{e}]\!]\Box$ ならば $\mathsf{e}x$ はどんな計算経路においても止まらない．
- $(x,x) \in \mathcal{A}[\![\mathsf{e}]\!]\Box$ しかも $(x,x) \notin \mathcal{D}[\![\mathsf{e}]\!]\Box$ ならば $\mathsf{e}x$ は止まることも，止まらないこともある．
- $(x,x) \in \mathcal{D}[\![\mathsf{e}]\!]\Box$ ならば $\mathsf{e}x$ はつねに止まる．

たとえば，f は大域的に定義されたプログラムとするとき，プログラム

$$\mathsf{g} = \mu\mathsf{k}.\mathsf{I}\,[]\,(\mathsf{k} \circ \mathsf{f})$$

に対して，(a1)～(a6) と (d1)～(d6) によって以下の意味定義が得られる．

$$\mathcal{A}[\![\mathsf{g}]\!] = \mu_{\subseteq} X.\ I \cup X \bullet f = f^*$$

$$\mathcal{D}[\![\mathsf{g}]\!] = \mu_{\sqsubseteq} Y.\ f^* \bullet (I \cup Y \circledcirc f)\Box = f^*$$

ゆえに任意の入力 x に対して (x,x) は $\mathcal{D}[\![\mathsf{g}]\!]\Box = I$ の要素であるので $\mathsf{g}x$ はつねに止まる．実際には，選択 $\mathsf{I}\,[]\,(\mathsf{k} \circ \mathsf{f})$ において，もしつねに $(\mathsf{k} \circ \mathsf{f})$ を選べば計算は停止しなくなってしまう $(\bot)$ が，片方の計算が $\bot$ であればもう一方を選ぶという $[]$ の定義によって，g の計算はつねに止まることになる．もちろん，$\mathsf{g}x$ の計算において何回 f を x に適用するのかは非決定的である．

3.4 プログラムの等価と洗練関係

前節で述べたようなプログラムの意味を用いると，2つのプログラム f と g の等価性は次のように定義される：

$$\mathsf{f} = \mathsf{g} \equiv \mathcal{A}[\![\mathsf{f}]\!] = \mathcal{A}[\![\mathsf{g}]\!] \wedge \mathcal{D}[\![\mathsf{f}]\!] = \mathcal{D}[\![\mathsf{g}]\!]$$

我々はプログラム f がプログラム g より洗練されたものであることを $\mathsf{f} \preceq \mathsf{g}$ で表し，次のように定義する．

$$\mathsf{f} \preceq \mathsf{g} \equiv \mathcal{A}[\![\mathsf{f}]\!] \subseteq \mathcal{A}[\![\mathsf{g}]\!] \wedge \mathcal{D}[\![\mathsf{f}]\!]\Box \supseteq \mathcal{D}[\![\mathsf{g}]\!]\Box$$

直観的には，$\mathsf{f} \preceq \mathsf{g}$ を満たすということは，f は g より決定的でかつ f のつねに停止する定義域は g より広いことを意味する．

4. UNITY ループ

4.1 直観的定義

Chandy と Misra の UNITY (Unbounded Nondeterministic Iterative Transformations)[4] は並列プログラムの仕様・設計と検証のために作られた抽象的な計算モデルである．

UNITY プログラムは本質的には代入式の空でない有限集合で記述される．代入は単純（決定的）かつつねに止まる（全域定義）ようなものと仮定される．プログラムの実行はある初期状態から出発し，次のような操作を繰り返し行う．代入式の集合から1つの代入式を選び実行する．代入式の選択は非決定的かつ公平に行われる．選択が公平であるというのは任意の代入式が永遠に選ばれないことがないような選択を指す．言い換えれば，任意の代入式は任意の計算時点のあとの有限回の選択の中に必ず選ばれる．

UNITY では単体プログラムの実行に関しては最終状態がないが，そのかわりにプログラムが不動状態，つまり，いずれの代入が行われても状態が変わらなく

なったら，そのプログラムは終了したと見なす．

4.2 関係代数による UNITY ループの記述

UNITY の代入は単純かつつねに止まると仮定されるので，代入を状態遷移関数として記述することができる．状態遷移関数の入力値は代入前の状態とし，関数の結果は代入直後の状態とする．よって，UNITY プログラムは状態遷移関数の集合で表すことができる．ある初期状態 s_0 から出発し，プログラムの集合 $\mathsf{L} = \{\mathsf{f}_1, \ldots, \mathsf{f}_n\}$ (f_i $(1 \leq i \leq n)$ は状態遷移関数) による可能な実行経路の 1 つは次のような関数合成の列となる：

$$(\ldots \mathsf{g}_3 \circ \mathsf{g}_2 \circ \mathsf{g}_1)s_0 \qquad (\forall i \geq 1 :\ \mathsf{g}_i \in \mathsf{L})$$

上の分析から，状態遷移関数の集合 L に対して，すべての可能な計算経路 $\{\mathsf{L}\}$ を次のような再帰的な式で表すことができる．

$$\{\mathsf{L}\} = \mu\mathsf{k}.\mathsf{fix}_{\mathsf{L}} \to \mathsf{I}, \mathsf{k} \circ \mathsf{Body}$$

Body はループの本体であり，以下で定義する．$\{\mathsf{L}\}$ を UNITY ループと呼ぶ．fix は次のように定義される．

$$\mathsf{fix}_{\{\mathsf{f}_1,\ldots,\mathsf{f}_n\}} = \lambda x.\ (x = \mathsf{f}_1 x) \wedge \ldots \wedge (x = \mathsf{f}_n x)$$

"公平な選択" を表すためには，まず，"関数の任意の有限回の合成" を表さなければならない．ここでは関数間のエンゼリック選択演算子の性質を利用して，関数 f の有限回の合成 g を以下のように定義する．

$$\mathsf{g} = \mu\mathsf{k}.\mathsf{I} \,[]\, (\mathsf{k} \circ \mathsf{f})$$

3.3 節で述べたように，g は非決定的な関数で，入力 a に対し $\mathsf{g}a$ の可能な結果は f を有限回 a に適用した結果となる．以後，簡単のため g を f^* と略記する．この記法を用いると $\mathsf{L} = \{\mathsf{f}_1, \ldots, \mathsf{f}_n\}$ からの有限回の関数の選択は

$$\mu\mathsf{k}.\mathsf{I} \,[]\, (\mathsf{k} \circ (\mathsf{f}_1 \,[]\, \ldots \,[]\, \mathsf{f}_n)) = (\mathsf{f}_1 \,[]\, \ldots \,[]\, \mathsf{f}_n)^*$$

となる．我々は上の式を L^* と記すことにする．

次に "L 中の任意の関数は有限回以内に必ず選ばれる" ようなプログラム <L> を定義する．

$$<\{\}> = \mathsf{I}$$

$$<\mathsf{L}> = \,[]_{\mathsf{f} \in \mathsf{L}} \mathsf{f} \circ (\mathsf{L} - \mathsf{f})^* \circ <\mathsf{L} - \mathsf{f}>$$

ここで

$$[]_{\mathsf{f} \in \{\mathsf{f}_1,\ldots,\mathsf{f}_n\}} \Phi(\mathsf{f}) = \Phi(\mathsf{f}_1) \,[]\, \ldots \,[]\, \Phi(\mathsf{f}_n)$$

とし，"−" は集合から要素を取り除く演算子とする．たとえば，2 つの関数の場合 $<\{\mathsf{f}, \mathsf{g}\}>$ は次のようになる．

$$<\{\mathsf{f}, \mathsf{g}\}> = \mathsf{f} \circ \mathsf{g}^* \circ \mathsf{g} \,[]\, \mathsf{g} \circ \mathsf{f}^* \circ \mathsf{f}$$

g が先に有限回 (1 回以上) 実行 $(\mathsf{g}^* \circ \mathsf{g})$ されたあと必ず f が実行される．また逆も同様である．3 つの関数の場合 $<\{\mathsf{f}, \mathsf{g}, \mathsf{h}\}>$ は以下のとおり．

$$\begin{aligned} <\{\mathsf{f}, \mathsf{g}, \mathsf{h}\}> = {} & \mathsf{f} \circ \{\mathsf{g}, \mathsf{h}\}^* \circ <\{\mathsf{g}, \mathsf{h}\}> \,[]\, \\ & \mathsf{g} \circ \{\mathsf{f}, \mathsf{h}\}^* \circ <\{\mathsf{f}, \mathsf{h}\}> \,[]\, \\ & \mathsf{h} \circ \{\mathsf{f}, \mathsf{g}\}^* \circ <\{\mathsf{f}, \mathsf{g}\}> \end{aligned}$$

上の記法を用いると，すべての関数が任意の計算時点以後，有限回以内に必ず選ばれることは，次のように表現できる．

$$\mathsf{Body} = <\mathsf{L}> \circ \mathsf{L}^*$$

ゆえに，UNITY ループは次のように定義することができる：

$$\{\mathsf{L}\} = \mu\mathsf{k}.\ \mathsf{fix}_{\mathsf{L}} \to \mathsf{I}, \mathsf{k} \circ <\mathsf{L}> \circ \mathsf{L}^* \qquad (25)$$

4.3 UNITY ループの意味定義

本節では関係代数を用いて，$\{\mathsf{L}\}$ に対して 2 つの側面から意味を与え，その代数的性質を探る．まず，L 中の状態遷移はすべて全域定義という前提から $\mathcal{A}[\![<\mathsf{L}>]\!] = \mathcal{D}[\![<\mathsf{L}>]\!]$ が成り立つので，以後，我々は $\mathcal{A}[\![<\mathsf{L}>]\!]$ を $<L>$ と書き，L を

$$\begin{aligned} L &= \{\ \mathcal{I}[\![\mathsf{f}_1]\!], \ldots, \mathcal{I}[\![\mathsf{f}_n]\!]\ \} \\ &= \{\ f_1, \ldots, f_n\ \} \end{aligned}$$

とする．また，$(\bigcup L)^*$ を L^* と略記する．

(a1)〜(a6) と (d1)〜(d6) によって $<L>$ の定義は次のように得られる．

$$<\{\}> = I$$

$$<L> = \bigcup_{f \in L} f \bullet (L - f)^* \bullet <L - f>$$

任意の $f \in L$ に対して $<L>$ に関する次の性質が成り立つ．これらの証明は付録で述べる．

$$<L - f> \bullet f \subseteq <L> \qquad (26)$$

$$<L> \bullet L^* \subseteq L^* \bullet <L> \qquad (27)$$

UNITY ループの意味定義は，定義 (25) と [] の意味定義から次のように得られる．

$$\begin{aligned} &\mathcal{A}[\![\{\mathsf{L}\}]\!] \\ &\quad = \mu_{\subseteq} X.\ B_{\mathsf{L}} \cup X \bullet <L> \bullet L^* \bullet \overline{B_{\mathsf{L}}} \\ &\qquad \mathcal{D}[\![\{\mathsf{L}\}]\!] \\ &\quad = \mu_{\sqsubseteq} X.\ \mathcal{A}[\![\{\mathsf{L}\}]\!] \bullet \\ &\qquad (B_{\mathsf{L}} \cup X \circledcirc <L> \circledcirc L^* \circledcirc \overline{B_{\mathsf{L}}}) \square \end{aligned}$$

where

$$B_{\mathsf{L}} = \mathcal{I}[\![\mathsf{fix}_{\mathsf{L}}]\!]?$$

4.4 UNITY ループの代数的性質

任意の関数 $f \in L$，関係 $R \subseteq L^*$ に対して，次のような代数的性質が成り立つ．

$$f \bullet B_{\mathsf{L}} = B_{\mathsf{L}} \qquad (28)$$

$$B_{\mathsf{L}} \bullet f \subseteq B_{\mathsf{L}} \bullet <L> \qquad (29)$$

$$<L> \bullet B_{\mathsf{L}} \subseteq B_{\mathsf{L}} \subseteq \mathcal{D}[\![\{\mathsf{L}\}]\!] \qquad (30)$$

$$L^* \bullet B_{\mathsf{L}} \subseteq B_{\mathsf{L}} \subseteq \mathcal{D}[\![\{\mathsf{L}\}]\!] \qquad (31)$$

$$B_{\mathsf{L}} \subseteq \mathcal{D}[\![\{\mathsf{L}\}]\!] \triangleright <L> \quad (32)$$

$$\{\mathsf{L}\} = \{\mathsf{L} \cup \{\mathsf{I}\}\} \quad (33)$$

$$\mathcal{D}[\![\{\mathsf{L}\}]\!]\square \subseteq \mathcal{D}[\![\{\mathsf{L}\}]\!] \triangleright R \quad (34)$$

性質(28)は，停止状態に対して，f を何回適用してもその値は変わらないということを意味する．性質(29)はある状態 x に対して fx が停止状態であるなら fx は <L>x の可能な解であることを意味する．性質(30)は性質(29)の拡張である．性質(32)は停止状態 x に対して <L> は状態を変えないこと，性質(33)はUNITYループに恒等関数を入れてもその意味が不変であることを意味する．性質(34)は $\mathcal{D}[\![\{\mathsf{L}\}]\!]$ の定義域に属す状態に対して，L 中の任意の関数を何回適用してもその結果は $\mathcal{D}[\![\{\mathsf{L}\}]\!]$ の定義域に属すことを意味する．

性質(28)～(33)は B_{L} の定義より直接得られる．性質(34)の証明は付録で述べる．

上の性質を利用して，UNITYループの意味定義は以下のように簡単化される．証明は付録を参照されたい．

定理 3

$$\mathcal{A}[\![\{\mathsf{L}\}]\!] = B_{\mathsf{L}} \bullet L^*$$

$$\mathcal{D}[\![\{\mathsf{L}\}]\!]\square = \mu_{\subseteq} X.B_{\mathsf{L}} \cup X \triangleright <L>$$

最後に，次の性質(35)は性質(32)と定理(3)から得られる．

$$\mathcal{D}[\![\{\mathsf{L}\}]\!]\square = \mathcal{D}[\![\{\mathsf{L}\}]\!] \triangleright <L> \quad (35)$$

性質(35)は x が $\mathcal{D}[\![\{\mathsf{L}\}]\!]$ の定義域に属すならば，<L>x も $\mathcal{D}[\![\{\mathsf{L}\}]\!]$ の定義域に属し，さらに，その逆もいえることを示す．

5. 並列探索への応用

次のプログラム α は，与えられた整数入力値から出発し，述語 p に対して pa が真になるような整数 a を求める簡単なプログラムである．

$$\alpha = \{\mathsf{g}\} \,[]\, \{\mathsf{h}\}$$

$$\mathsf{g} = \mathsf{p} \rightarrow \mathsf{I}, \mathsf{succ}$$

$$\mathsf{h} = \mathsf{p} \rightarrow \mathsf{I}, \mathsf{pred}$$

ただし，succ と pred はプリミティブ関数とし，その意味関数を

$$succ = \mathcal{I}[\![\mathsf{succ}]\!] = \{(n+1, n) \mid n \text{ は整数}\}$$

$$pred = \mathcal{I}[\![\mathsf{pred}]\!] = \{(n-1, n) \mid n \text{ は整数}\}$$

とする．

このプログラムは入力より大きい方向と小さい方向の両側を並列に探し，先に見つかるものを返す．このプログラムを1つのUNITYループにまとめ，一方で条件を満たすものが見つかったら他方へ明示的に知らせるようにすると，次のようになる．

$$\beta = \pi_1 \circ \{\mathsf{L}_\beta\} \circ (\mathsf{I} \vartriangle \mathsf{I})$$

$$\mathsf{L}_\beta = \{\mathsf{g}_1, \mathsf{h}_1\}$$

$$\mathsf{g}_1 = (\mathsf{p} \circ \pi_1) \rightarrow (\pi_1 \vartriangle \pi_1), (\mathsf{succ} \times \mathsf{I})$$

$$\mathsf{h}_1 = (\mathsf{p} \circ \pi_2) \rightarrow (\pi_2 \vartriangle \pi_2), (\mathsf{I} \times \mathsf{pred})$$

ここでは，関数 π_1 と π_2，関数上の演算子 $\vartriangle$ と $\times$ を使った．s と t を関数とするとき，これらの定義は次のようになる．

$$\begin{aligned} \pi_1(a,b) &= a \\ \pi_2(a,b) &= b \\ (\mathsf{s} \vartriangle \mathsf{t})\,a &= (\mathsf{s}a, \mathsf{t}a) \\ (\mathsf{s} \times \mathsf{t})(a,b) &= (\mathsf{s}a, \mathsf{t}b) \end{aligned}$$

ここで，4.1節まで紹介した道具を使い β は α の洗練されたものであること（$\beta \preceq \alpha$）を証明する．

3.2節から，各関数の意味定義が次のように得られる．

$$\begin{aligned} \mathcal{A}[\![\mathsf{g}]\!] &= p? \cup succ \bullet \overline{p?} = g \\ \mathcal{A}[\![\mathsf{h}]\!] &= p? \cup pred \bullet \overline{p?} = h \\ \mathcal{A}[\![\mathsf{g}_1]\!] &= (\pi_1 \vartriangle \pi_1) \bullet (p \bullet \pi_1)? \cup \\ &\qquad (succ \times I) \bullet \overline{(p \bullet \pi_1)?} \\ &= (p? \times p?) \bullet (\pi_1 \vartriangle \pi_1) \cup \\ &\qquad ((succ \bullet \overline{p?}) \times I) \\ &= g_1 \\ \mathcal{A}[\![\mathsf{h}_1]\!] &= (\pi_2 \vartriangle \pi_2) \bullet (p \bullet \pi_2)? \cup \\ &\qquad (I \times pred) \bullet \overline{(p \bullet \pi_2)?} \\ &= (p? \times p?) \bullet (\pi_2 \vartriangle \pi_2) \cup \\ &\qquad (I \times (pred \bullet \overline{p?})) \\ &= h_1 \\ B_{\mathsf{L}_\beta} &= (p? \times p?) \bullet (=)? \\ \mathcal{A}[\![\alpha]\!] &= p? \bullet g^* \cup p? \bullet h^* \\ \mathcal{A}[\![\beta]\!] &= \pi_1 \bullet B_{\mathsf{L}_\beta} \bullet L_\beta^* \bullet (I \vartriangle I) \\ \mathcal{D}[\![\alpha]\!]\square &= \mathcal{D}[\![\{\mathsf{g}\}]\!]\square \cup \mathcal{D}[\![\{\mathsf{h}\}]\!]\square \\ &= (\mu_{\subseteq} X.p? \cup X \triangleright g) \cup \\ &\qquad (\mu_{\subseteq} X.p? \cup X \triangleright h) \\ \mathcal{D}[\![\beta]\!]\square &= (\pi_1 \circledcirc \mathcal{D}[\![\{\mathsf{L}_\beta\}]\!] \circledcirc (I \vartriangle I))\square \\ &= (\mathcal{D}[\![\{\mathsf{L}_\beta\}]\!])\square \triangleright (I \vartriangle I) \\ &\qquad \{\pi_1,\ (I \vartriangle I) \text{ 全域定義}\} \end{aligned}$$

まず，$\mathcal{A}[\![\beta]\!] \subseteq \mathcal{A}[\![\alpha]\!]$ を証明する．

$$\begin{aligned} &\mathcal{A}[\![\beta]\!] \subseteq \mathcal{A}[\![\alpha]\!] \\ \equiv\ &\{\text{ 定義 }\} \\ &\pi_1 \bullet B_{\mathsf{L}_\beta} \bullet L_\beta^* \bullet (I \vartriangle I) \\ &\subseteq \mathcal{A}[\![\alpha]\!] \\ \Leftarrow\ &\{\ B_{\mathsf{L}_\beta} \text{ の定義 }\} \end{aligned}$$

$\{\ \pi_1 \bullet (I \vartriangle I) = \pi_2 \bullet (I \vartriangle I) = I\ \}$
$\{$ 両側に $I \vartriangle I$ を結合 $\}$
$\pi_1 \bullet (p? \times p?) \bullet (=)? \bullet L_\beta^*$
$\subseteq\ \mathcal{A}[\![\alpha]\!] \bullet (\pi_1 \cup \pi_2)$

$\Leftarrow$ $\{$ 性質 (19)，Tarski 定理 $\}$
$\pi_1 \bullet (p? \times p?) \bullet (=)? \cup$
$\mathcal{A}[\![\alpha]\!] \bullet (\pi_1 \cup \pi_2) \bullet (\cup L_\beta)$
$\subseteq\ \mathcal{A}[\![\alpha]\!] \bullet (\pi_1 \cup \pi_2)$

$\equiv$ $\{\mathcal{A}[\![\alpha]\!], g, h, g_1, h_1$ の定義，性質 (3) $\}$
$\{$ 閉包の性質 $\}$
$true$

次は，$\mathcal{D}[\![\alpha]\!]\Box \subseteq \mathcal{D}[\![\beta]\!]\Box$ を証明する．

$\mathcal{D}[\![\alpha]\!]\Box \subseteq \mathcal{D}[\![\beta]\!]\Box$

$\equiv$ $\{$ 定義，定理 (3) $\}$
$(\mu_\subseteq X.p? \cup X \triangleright g) \cup (\mu_\subseteq X.p? \cup X \triangleright h)$
$\subseteq\ \mathcal{D}[\![\beta]\!]\Box$

$\Leftarrow$ $\{$ 対称関係 $\}$
$\mu_\subseteq X.p? \cup X \triangleright g \subseteq \mathcal{D}[\![\beta]\!]\Box$

$\Leftarrow$ $\{$ 性質 (17)，Tarski 定理 $\}$
$p? \cup \mathcal{D}[\![\beta]\!]\Box \triangleright g \subseteq \mathcal{D}[\![\beta]\!]\Box$

$\Leftarrow$ $\{\ \mathcal{D}[\![\beta]\!]\Box$ の定義より $p? \subseteq \mathcal{D}[\![\beta]\!]\Box\ \}$
$\mathcal{D}[\![\beta]\!]\Box \triangleright g \subseteq \mathcal{D}[\![\beta]\!]\Box$

$\Leftarrow$ $\{\ g$ の定義，$p? \subseteq \mathcal{D}[\![\beta]\!]\Box\ \}$
$\mathcal{D}[\![\beta]\!]\Box \triangleright (succ \bullet \overline{p?}) \subseteq \mathcal{D}[\![\beta]\!]\Box$

$\Leftarrow$ $\{$ 次の (b) の証明 $\}$
$\mathcal{D}[\![\beta]\!]\Box \subseteq \mathcal{D}[\![\beta]\!]\Box \triangleright (pred \bullet \overline{p?})$

$\equiv$ $\{\ \mathcal{D}[\![\beta]\!]\Box$ の定義，性質 (16) $\}$
$\mathcal{D}[\![\beta]\!]\Box \subseteq$
$\mathcal{D}[\![\{\mathsf{L}_\beta\}]\!]\Box \triangleright ((I \vartriangle I) \bullet pred \bullet \overline{p?})$

$\equiv$ $\{\ fp = pred \bullet \overline{p?}$ とする $\}$
$\{\ \vartriangle$，$\times$ の性質 $\}$
$\mathcal{D}[\![\beta]\!]\Box \subseteq$
$\mathcal{D}[\![\{\mathsf{L}_\beta\}]\!]\Box \triangleright ((fp \times fp) \bullet (I \vartriangle I))$

$\Leftarrow$ $\{$ 性質 (35) $\}$
$\{\ K := (<L_\beta> \bullet (fp \times fp) \bullet (I \vartriangle I))\ \}$
$\mathcal{D}[\![\beta]\!]\Box \subseteq \mathcal{D}[\![\{\mathsf{L}_\beta\}]\!]\Box \triangleright K$

$\Leftarrow$ $\{$ 次の (c) の証明 $\}$
$\mathcal{D}[\![\beta]\!]\Box \subseteq \mathcal{D}[\![\{\mathsf{L}_\beta\}]\!]\Box \triangleright (I \vartriangle I)$

$\equiv$ $\{\ \mathcal{D}[\![\beta]\!]\Box$ の定義 $\}$
$true$

(b) $\mathcal{D}[\![\beta]\!]\Box \triangleright (succ \bullet \overline{p?}) \subseteq \mathcal{D}[\![\beta]\!]\Box$
$\Leftarrow$
$\mathcal{D}[\![\beta]\!]\Box \subseteq \mathcal{D}[\![\beta]\!]\Box \triangleright (pred \bullet \overline{p?})$ の証明

$\mathcal{D}[\![\beta]\!]\Box \triangleright (succ \bullet \overline{p?}) \subseteq \mathcal{D}[\![\beta]\!]\Box$

$\Leftarrow$ $\{$ 前提： $\}$
$\{\ \mathcal{D}[\![\beta]\!]\Box \subseteq \mathcal{D}[\![\beta]\!]\Box \triangleright (pred \bullet \overline{p?})\ \}$
$\{\ \lambda X.\ X \triangleright R$ は $\subseteq$-単調 $\}$
$(\mathcal{D}[\![\beta]\!]\Box \triangleright (pred \bullet \overline{p?})) \triangleright (succ \bullet \overline{p?})$
$\subseteq\ \mathcal{D}[\![\beta]\!]\Box$

$\equiv$ $\{\ (R \triangleright S) \triangleright T\ =\ R \triangleright (S \bullet T)\ \}$
$\mathcal{D}[\![\beta]\!]\Box \triangleright (pred \bullet \overline{p?} \bullet succ \bullet \overline{p?})$
$\subseteq\ \mathcal{D}[\![\beta]\!]\Box$

$\equiv$ $\{\ \overline{p?} \bullet succ = succ \bullet \overline{(p \bullet succ)?}\ \}$
$\{\ pred \bullet succ = I\ \}$
$\mathcal{D}[\![\beta]\!]\Box \triangleright (\overline{(p \bullet succ)?} \bullet \overline{p?}) \subseteq \mathcal{D}[\![\beta]\!]\Box$

$\Leftarrow$ $\{\ A \subseteq R\Box \Rightarrow R\Box \triangleright \overline{A} \subseteq R\Box\ \}$
$\{\ \overline{(\overline{(p \bullet succ)?} \bullet \overline{p?})} = p? \cup (p \bullet succ)?\ \}$
$p? \cup (p \bullet succ)? \subseteq \mathcal{D}[\![\beta]\!]\Box$

$\Leftarrow$ $\{\ \mathcal{D}[\![\beta]\!]$ の定義 $\}$
$true$

(c) $\mathcal{D}[\![\beta]\!]\Box \subseteq \mathcal{D}[\![\{\mathsf{L}_\beta\}]\!]\Box \triangleright$
$(<L_\beta> \bullet (fp \times fp) \bullet (I \vartriangle I))$
$\Leftarrow$
$\mathcal{D}[\![\beta]\!]\Box \subseteq \mathcal{D}[\![\{\mathsf{L}_\beta\}]\!]\Box \triangleright (I \vartriangle I)$ の証明

$\mathcal{D}[\![\beta]\!]\Box \subseteq \mathcal{D}[\![\{\mathsf{L}_\beta\}]\!]\Box \triangleright$
$(<L_\beta> \bullet (fp \times fp) \bullet (I \vartriangle I))$

$\Leftarrow$ $\{$ 定義，閉包性質より $\}$
$\{\ <L_\beta> \bullet (fp \times fp) \bullet (I \vartriangle I)\ \}$
$\{\ \subseteq L_\beta^* \bullet (I \vartriangle I)\ \}$
$\{\ \lambda X.\ R \triangleright X$ は逆 $\subseteq$-単調 $\}$
$\mathcal{D}[\![\beta]\!]\Box \subseteq \mathcal{D}[\![\{\mathsf{L}_\beta\}]\!]\Box \triangleright (L_\beta^* \bullet (I \vartriangle I))$

$\Leftarrow$ $\{\ (R \triangleright S) \triangleright T\ =\ R \triangleright (S \bullet T)\ \}$
$\mathcal{D}[\![\beta]\!]\Box \subseteq (\mathcal{D}[\![\{\mathsf{L}_\beta\}]\!]\Box \triangleright L_\beta^*) \triangleright (I \vartriangle I))$

$\Leftarrow$ $\{$ 性質 (34): $\}$
$\{\ \mathcal{D}[\![\{\mathsf{L}_\beta\}]\!]\Box \subseteq \mathcal{D}[\![\{\mathsf{L}_\beta\}]\!]\Box \triangleright L_\beta^*\ \}$
$\{\ \lambda X.\ X \triangleright R$ は $\subseteq$-単調 $\}$
$\mathcal{D}[\![\beta]\!]\Box \subseteq \mathcal{D}[\![\{\mathsf{L}_\beta\}]\!]\Box \triangleright (I \vartriangle I))$

6. 議　　論

本論文は従来の UNITY ループに対する定義・推論手法とは異なり，プログラムの入出力のみに注目し，それらの対の集合（関係）について議論した．関係の固有の数学的性質を用いて（関係・プログラムの）合成的（compositional）に，状態のうえでの演算ではなく関係上の演算によって簡潔的な証明ができるよう

になった.

本論文の手法を用いてUNITYの主要な推論リレーション "leads-to" を表現することができる. p, q を2つの述語とすると, 直観的には, "p leads-to$_{\mathsf{L}}$ q" が真であるとは初期状態 x ($\mathsf{p}\,x = true$) から, L による繰り返しにより代入を有限回実行した後必ず $\mathsf{q}\,y = true$ となるような状態 y に到達することを表す. 関係を用いて p leads-to$_{\mathsf{L}}$ q を表現すると次のようになる.

$p? \subseteq (q? \circledcirc \mathcal{D}[\![\{\mathsf{L'}\}]\!])\Box$

where

$\mathsf{L'} = \{\, \mathsf{q} \rightarrow \mathsf{I}, \mathsf{f} \mid \mathsf{f} \in \mathsf{L} \,\}$

我々の $\mathcal{A}[\![\_]\!]$ と $\mathcal{D}[\![\_]\!]$ 記法とDijkstraの wlp/wp 記法[5]との関係は次のようになる. f を非決定的なプログラムとし, Q はその後条件 (post-condition) とする. 任意の入力値 x に対して, 次の関係式が成り立つ.

$(wlp\ \mathsf{f}\ \mathsf{Q})x$

$\equiv \forall y : (y, x) \in \mathcal{A}[\![\mathsf{f}]\!] \Rightarrow (\mathsf{Q}\,y)$

$(wp\ \mathsf{f}\ \mathsf{Q})x$

$\equiv \exists y : (y, x) \in \mathcal{D}[\![\mathsf{f}]\!] \wedge (wlp\ \mathsf{f}\ \mathsf{Q})x$

本論文で述べたプログラムはすべて total command[3]である. プログラム f が total であるのは次の条件を満たすようなものを指す.

$wp\ \mathsf{f}\ false = false.$

Partial command の上のエンゼリック選択と関数の Preorder 閉包を用いて, UNITY の公平選択を (2変数のみの場合) 表現するアイディアは Broy ら[3]にも述べられている. しかし, UNITY ループは本論文で述べたような total command で十分定義できるので, Broy のような partial command のための繁雑な定義は必要ない. さらに, エンゼリックに対する Broy の定義に使う道具は本論文と違うので, その証明方法なども当然異なってくる.

7. おわりに

本論文では関係代数によって局所エンゼリック選択とデータ構造定義を合わせたプログラムの意味づけを行い, その応用例として UNITY ループを記述した. 関係代数の定義からさまざまな UNITY ループを含むような並列アルゴリズムを導出するのは将来の課題となろう.

参考文献

1) Backhouse, R. and Hoogendijk, P.: Elements of Relational Theory of Datatypes, *Lecture Notes in Computer Science*, Vol.755, Springer-Verlag (1993).
2) Backhouse, R. and van der Woude, J.: Demonic Operators and Monotype Factors, *Mathematical Structures in Computer Science*, Vol.3, No.4, pp.417–433, Springer-Verlag (1993).
3) Broy, M. and Nelson, G.: Adding Fair Choice to Dijkstra's Calculus, *ACM Trans. Prog. Lang. Syst.*, Vol.16, No.3, pp.924–938 (1994).
4) Chandy, K.M. and Misra, J.: *Parallel Program Design: A Foundation*, Addison Wesley, Reading, MA (1988).
5) Dijkstra, E.W. and Scholten, C.S.: *Predicate Calculus and Program Semantics*, Springer-Verlag, Berlin (1990).
6) McCarthy, J.: Towards a Mathematical Science of Computation, *Computer Programming and Formal Systems*, Braffort, P. and Hirschberg, D., (Eds.), pp.33–70, North-Holland (1963).
7) Sondergaard, H. and Sestoft, P.: Nondeterminism in Functional Languages, *The Computer Journal*, Vol.35, No.5, pp.514–523 (1992).
8) Tarski, A.: On the calculus of relations, *Journal of Symbolic Logic*, Vol.6, No.3, pp.73–89 (1941).
9) Xu, L., Takeichi, M., and Iwasaki, H.: Relational Semantics for Locally Nondeterministic Programs, *New Generation Computing*, Vol.15, No.3, pp.339–362 (1997).

付録

ここでは本論文に現れた定理, 性質などを証明する.

性質 (26) の証明 L の大きさ関する帰納法を用いて証明する.

$L = \{f\}$ の場合：$<L>$ の定義により明らかに成り立つ.

L の大きさが2以上の場合：

左辺

$= \{$ 定義 $\}$

$$\bigcup_{g \in L-f} g \bullet (L-g-f)^* \bullet <L-g-f> \bullet f$$

$\subseteq \{$ 帰納法の仮定 $\}$

$$\bigcup_{g \in L-f} g \bullet (L-g-f)^* \bullet <L-g>$$

$\subseteq \{$ 性質 (24): $\}$

$\{\, (L-g-f)^* \subseteq (L-g)^* \,\}$

$$\bigcup_{g\in L-f} g\bullet(L-g)^*\bullet{<}L-g{>}$$
$\subseteq$ { 和集合 }
$$\bigcup_{g\in L} g\bullet(L-g)^*\bullet{<}L-g{>}$$
$=$ { 定義 }
右辺

性質 (27) の証明

${<}L{>}\bullet L^* \subseteq L^*\bullet{<}L{>}$
$\equiv$ { 性質 (19) }
$\mu_{\subseteq}X.\ {<}L{>}\cup X\bullet(\bigcup L)\subseteq L^*\bullet{<}L{>}$
$\Leftarrow$ { Tarski 定理 }
${<}L{>}\cup L^*\bullet{<}L{>}\bullet(\bigcup L)\subseteq L^*\bullet{<}L{>}$
$\Leftarrow$ { ${<}L{>}\subseteq L^*\bullet{<}L{>}$ }
$\forall f\in L:\ L^*\bullet{<}L{>}\bullet f\subseteq L^*\bullet{<}L{>}$
$\equiv$ { 次の (a) の証明 }
true

(a) $\forall f\in L:\ L^*\bullet{<}L{>}\bullet f\subseteq L^*\bullet{<}L{>}$ の証明
$L(\neq\emptyset)$ の要素数に関する帰納法を用いて証明する．
$L=\{f\}$ の場合：定義と性質 (22) によって明らかである．
L の大きさが 2 以上の場合：

左辺
$=$ { 定義 }
$$L^*\bullet\bigcup_{g\in L} g\bullet(L-g)^*\bullet{<}L-g{>}\bullet f$$
$=$ {$g=f$ と $g\neq f$ を場合分け }
$$L^*\bullet f\bullet(L-f)^*\bullet{<}L-f{>}\bullet f\cup L^*\bullet\bigcup_{g\in L-f} g\bullet(L-g)^*\bullet{<}L-g{>}\bullet f$$
$\subseteq$ { 性質 (22)：$L^*\bullet f\bullet(L-f)^*\subseteq L^*$ }
{ 帰納法の仮定 }
$$L^*\bullet{<}L-f{>}\bullet f\cup L^*\bullet\bigcup_{g\in L-f} g\bullet(L-g)^*\bullet{<}L-g{>}$$
$\subseteq$ { 性質 (26)，${<}L{>}$ の定義 }
$L^*\bullet{<}L{>}\cup L^*\bullet{<}L{>}$
$=$ { 和集合 }
右辺

性質 (34) の証明 以下の証明は $\mathcal{D}[\![\{\mathsf{L}\}]\!]$ を S_D とし，$\mathcal{A}[\![\{\mathsf{L}\}]\!]$ を S_A とする．
$S_D\Box\subseteq S_D\triangleright R$
$\equiv$ { S_D の定義 }
{ $S_D\sqsubseteq S_A$ }
{ ${<}L{>}$ と L^* は全域定義 }
$S_D\Box\subseteq(B_{\mathsf{L}}\cup S_D\triangleright({<}L{>}\bullet L^*))\triangleright R$
$\Leftarrow$ { 性質 (30)，性質 (31) }
{ $B_{\mathsf{L}}\subseteq S_D\triangleright({<}L{>}\bullet L^*)$ }
{ 性質 (16) }
$S_D\Box\subseteq S_D\triangleright({<}L{>}\bullet L^*\bullet R)$
$\Leftarrow$ { $R\subseteq L^*\Rightarrow L^*\bullet R\subseteq L^*$ }
{ 性質 (18) }
$S_D\Box\subseteq S_D\triangleright({<}L{>}\bullet L^*)$
$\Leftarrow$ { $B_{\mathsf{L}}\subseteq S_D\triangleright({<}L{>}\bullet L^*)$ }
$S_D\Box\subseteq B_{\mathsf{L}}\cup S_D\triangleright({<}L{>}\bullet L^*)$
$\equiv$ { 定義 }
true

定理 (3) の証明 以下の証明は $\mathcal{D}[\![\{\mathsf{L}\}]\!]$ を S_D，$\mathcal{A}[\![\{\mathsf{L}\}]\!]$ を S_A と記す．まず，式
$S_A=B_{\mathsf{L}}\bullet L^*$
を証明する．

$S_A\subseteq B_{\mathsf{L}}\bullet L^*$
$\equiv$ { S_A の定義 }
$\mu_{\subseteq}X.B_{\mathsf{L}}\cup X\bullet{<}L{>}\bullet L^*\bullet\overline{B_{\mathsf{L}}}$
$\subseteq B_{\mathsf{L}}\bullet L^*$
$\Leftarrow$ { Tarski 定理 }
$B_{\mathsf{L}}\cup B_{\mathsf{L}}\bullet L^*\bullet{<}L{>}\bullet L^*\bullet\overline{B_{\mathsf{L}}}$
$\subseteq B_{\mathsf{L}}\bullet L^*$
$\Leftarrow$ { $B_{\mathsf{L}}\subseteq B_{\mathsf{L}}\bullet L^*$ }
{ $L^*\bullet{<}L{>}\bullet L^*\bullet\overline{B_{\mathsf{L}}}\subseteq L^*$ }
$B_{\mathsf{L}}\bullet L^*\subseteq B_{\mathsf{L}}\bullet L^*$

逆に，

$B_{\mathsf{L}}\bullet L^*\subseteq S_A$
$\Leftarrow$ { Tarski 定理 }
$B_{\mathsf{L}}\cup S_A\bullet(\bigcup L)\subseteq S_A$
$\equiv$ { $B_{\mathsf{L}}\subseteq S_A$ }
$\forall f\in L:\ S_A\bullet f\subseteq S_A$
$\Leftarrow$ { S_A の定義，性質 (30)，性質 (31) }
{ $S_A\bullet B_{\mathsf{L}}\subseteq B_{\mathsf{L}}$ }
$\forall f\in L:$
$(B_{\mathsf{L}}\cup S_A\bullet{<}L{>}\bullet L^*)\bullet f\subseteq S_A$
$\Leftarrow$ { $B_{\mathsf{L}}\bullet f\subseteq S_A$ }
{ $L^*\bullet f\subseteq L^*$ }

{ S_A の定義 }
$true$

次は,

$S_D\Box = \mu_{\subseteq} X.B_{\lfloor} \cup X \triangleright <L>$

を証明する.

$\mu_{\subseteq} X.B_{\lfloor} \cup X \triangleright <L>$
$\subseteq S_D\Box$
$\Leftarrow$ { 性質 (17), Tarski 定理 }
$B_{\lfloor} \cup S_D \triangleright <L>$
$\subseteq S_D\Box$
$\Leftarrow$ { $B_{\lfloor} \subseteq S_D\Box$ }
{ S_D の定義, }
{ $S_D\Box \supseteq S_D \triangleright (<L> \bullet L^*)$ }
{ $\triangleright$ の性質 }
$\Box(<L> \bullet L^* \bullet S_D \triangleright <L>)$
$\subseteq S_D\Box$
$\Leftarrow$ { 性質 (27) }
{ $\Box(U \bullet V) = \Box(U \bullet \Box V)$ }
{ $\Box(V \bullet U \triangleright V) \subseteq U\Box$ }
$\Box(L^* \bullet S_D\Box)$
$\subseteq S_D\Box$
$\Leftarrow$ { 性質 (34) }
{ $\Box(V \bullet (U \triangleright V)) \subseteq U\Box$ }
$true$

逆に,

$S_D\Box \subseteq \mu_{\subseteq} X.B_{\lfloor} \cup X \triangleright <L>$
$\equiv$ { $\mu_{\subseteq} X.B_{\lfloor} \cup X \triangleright <L>$ を A と記す }
$S_D\Box \subseteq A$
$\Leftarrow$ { $S_D \sqsubseteq S_A$ }
$S_D \subseteq S_A \bullet A$
$\Leftarrow$ { $\sqsubseteq$ の定義 }
$S_D \sqsubseteq S_A \bullet A$
$\Leftarrow$ { S_D の定義, Tarski 定理 }
$S_A \bullet (B_{\lfloor} \cup$
$(S_A \bullet A) \circledcirc <L> \circledcirc L^* \circledcirc \overline{B_{\lfloor}})\Box$
$\sqsubseteq S_A \bullet A$
$\Leftarrow$ { $\sqsubseteq$ の定義 }
$(B_{\lfloor} \cup (S_A \bullet A) \circledcirc$
$<L> \circledcirc L^* \circledcirc \overline{B_{\lfloor}})\Box \subseteq A$
$\Leftarrow$ { $\Box$ の性質, A の定義 }
{ 本証明の前半 $A \subseteq S_D\Box \subseteq S_A\Box$ }
$(A \circledcirc <L> \circledcirc L^* \circledcirc \overline{B_{\lfloor}})\Box \subseteq A$
$\Leftarrow$ { $\circledcirc$ の性質, $<L>$ と L^* 全域定義 }
$A \triangleright (<L> \bullet L^*) \bullet \overline{B_{\lfloor}} \subseteq A$
$\Leftarrow$ { $\overline{B_{\lfloor}} \subseteq I$, }
{ $<L> \subseteq <L> \bullet L^*$ }
{ 性質 (18) }
$A \triangleright <L> \subseteq A$
$\equiv$ { 定義 }
$true$

(平成 9 年 4 月 1 日受付)
(平成 10 年 1 月 16 日採録)

徐　良為（正会員）

1959 年生．1992 年東京大学大学院工学研究科情報工学専攻修士課程修了．1995 年東京大学大学院工学研究科情報工学専攻博士課程退学．同年数理システム（株）入社．プログラミング言語，非決定的なプログラムの意味定義，推論に興味を持つ．

武市　正人（正会員）

1948 年生．1970 年東京大学工学部計数工学科卒業．1972 年同大学大学院工学系研究科修士課程修了．同年同大学工学部計数工学科助手，講師，1977 年電気通信大学電気通信学部計算機科学科講師，助教授，1987 年東京大学教育用計算機センター助教授（工学部併任），1990 年同大学工学部計数工学科助教授，1993 年より同教授（情報処理工学講座），現在に至る．工学博士．プログラミング言語，関数プログラミング，構成的アルゴリズム論の研究・教育に従事．日本応用数理学会，ACM 各会員．

岩崎　英哉（正会員）

1960 年生．1983 年東京大学工学部計数工学科卒業．1988 年同大学大学院工学系研究科情報工学専攻博士課程修了．同年同大学計数工学科助手，1993 年同大学教育用計算機センター助教授を経て，1996 年東京農工大学工学部電子情報工学科助教授．工学博士．日本ソフトウェア科学会，ACM 各会員．