# 時間の論理の束モデルを用いた並行プログラム系の検証*

4J−2

塩 雅之（筑波大学 工学研究科）† 五十嵐 滋（筑波大学 電子・情報工学系）‡
辻 尚史（筑波大学 電子・情報工学系）‡ 水谷 哲也（筑波大学 電子・情報工学系）‡
白銀 哲也（筑波大学 工学研究科）†

## 1 序

有理数時間を扱うプログラム検証のための形式的体系である envelope system[?, ?] を提示する。時刻で真理値の変化する命題の真理集合－解釈が真である時刻の集合－を考え、閉包概念を特殊化した envelope と呼ぶ集合演算子を定義した。これにより、プログラム検証を容易かつ厳密に遂行できるようになった。最後に、並行プログラム系についての基本的かつ高度な概念と問題点を持つ Dekker の解の飢餓回避問題について検証した。

## 2 envelope system

envelope system は並行プログラム系の検証を行うための形式的体系である。$T_c$ を適当な理論 $T$ に可算個の定数をプログラム変数として扱うために加えて、その解釈を与えたものとする。このとき $T$ のモデル $\mathcal{M}$ の自然な拡張を $\mathcal{M}_c$ とする。

### 2.1 構文

envelope system の論理式、集合（es 集合と呼ぶ）を以下のように再帰的に定義する。

$$
\begin{array}{rcl}
\langle\text{formula}\rangle & ::= & \langle\text{set}\rangle = \mathcal{I} \mid \neg\langle\text{formula}\rangle \\
& & \mid \langle\text{formula}\rangle \vee \langle\text{formula}\rangle \\
\langle\text{set}\rangle & ::= & \langle T_c \text{ の formula}\rangle \\
& & \mid \langle\text{set}\rangle^c \mid \langle\text{set}\rangle / \\
& & \mid \langle\text{set}\rangle \cup \langle\text{set}\rangle \mid \langle\text{set}\rangle \cap \langle\text{set}\rangle
\end{array}
$$

ただし ⟨set⟩ $= \mathcal{I}$ の $= \mathcal{I}$ は省略してよい。$\wedge, \supset, \equiv$ は普通の略記法に従う。集合 $A, B$ について $(A\cap B)\cup(A^c\cap B^c) = \mathcal{I}$ を $A \ominus B$ と表す。$\ominus$ は集合間の普通の等号と考えてよい。以下、混乱の恐れの 無い限り = で表す。

### 2.2 解釈

envelope system の解釈は軌跡 $\chi$、集合の解釈 ~ と論理式の解釈 − の三つ組によって与えられる。

$Q^{>0}$ を正の有理数全体の集合とし、$\mathcal{M}_c$ 全体の集合を Mdl とする。このとき $\chi : Q^{>0} \to$ Mdl を軌跡と呼ぶ。$\chi$, $t$ で論理式 $P$ が真であることを $\chi(t) \models P$ と表す。$Q^{>0}$ の部分集合に対し、その特徴関数が左連続で端点が離散的であるものを locus 型とするとき、locus 型である集合 $S(\subseteq Q^{>0})$ について、その閉包を $S$ を含む最小の上切片と定義して、$Q^{>0}$ を位相づけることができる。この位相における閉包を特に envelope と呼ぶ。集合 $S$ の envelope を $S/$ と表す。/ の結合は $\cup, \cap, ^c$ より弱いものとする。es 集合 $A$ は、観察時刻に従って有理数の部分集合として解釈される。$P$ を $T_c$ の論理式、$A$, $B$ を es 集合としたとき es 集合の解釈 ~ の定義は次の通りである。

$$
\begin{array}{rcl}
\widetilde{P}^t & = & \{u \mid t < u,\ \chi(u) \models P\} \\
\widetilde{A^c}^t & = & \{u \mid t < u,\ u \notin \widetilde{A}^t\} \\
\widetilde{A/}^t & = & \begin{cases} \phi & \text{if} \widetilde{A}^t = \phi \\ \{u \mid \inf \widetilde{A}^t < u\} & \text{otherwise} \end{cases} \\
\widetilde{A \cup B}^t & = & \{u \mid u \in \widetilde{A}^t \text{ or } u \in \widetilde{B}^t\} \\
\widetilde{A \cap B}^t & = & \{u \mid u \in \widetilde{A}^t \text{ and } u \in \widetilde{B}^t\}
\end{array}
$$

論理式は時刻を与えられて真理値を得る。

$$
\begin{array}{rcl}
(A = \mathcal{I})_t^- = \text{true} & \overset{\text{def}}{\Leftrightarrow} & \widetilde{A}^t = \{u \mid t < u\} \\
(\neg F)_t^- = \text{true} & \overset{\text{def}}{\Leftrightarrow} & F_t^- = \text{false} \\
(F \vee G)_t^- = \text{true} & \overset{\text{def}}{\Leftrightarrow} & F_t^- = \text{true or } G_t^- = \text{true}
\end{array}
$$

### 2.3 推論体系

推論体系としては $LK$ を採用する。

定義 sequent

$F_1, \cdots, F_m, G_1, \cdots, G_n$ を論理式とすると

$$F_1, \cdots, F_m \to G_1, \cdots, G_n$$

が sequent であり、次の条件が成立するときにのみ、この sequent が成立する。

$$\forall t(\ (\ (F_1)_t^- = \text{true} \wedge \cdots \wedge (F_m)_t^- = \text{true}\ )$$

$$\Rightarrow (\ (G_1)_t^- = \text{true} \vee \cdots \vee (G_n)_t^- = \text{true}\ )\ )$$

公理は、等号に関する公理（反射律、代入則）、束の演算子に関する公理、閉包の公理に以下のものを加える。

- 論理演算子と束の公理
  $(A/ = \mathcal{I}) \vee (B/ = \mathcal{I}) \equiv A/ \cup (B/) = \mathcal{I}$

- envelope の公理
  $A/ \supseteq B/ \vee B/ \supseteq A/$

*Verifications of Parallel Program Systems using the Latice Model of the Tense Logic

†Masayuki Shio, Tetsuya Shirogane , Doctoral Program of Engeneering, University of Tsukuba

‡Shigeru Igarashi, Takashi Tsuji and Tetsuya Mizutani, Institute of Information Sciences , University of Tsukuba

<u>定義　monogenic</u>

es 集合 $A$ について

$\forall t,\ u,\ v \in Q^{>0}(t+u+v \in \widetilde{A}^{t} \Leftrightarrow t+v \in \widetilde{A}^{t+u})$ が成立するとき、$A$ は monogenic であるという。集合の解釈～の定義より、$T_c$ の論理式 $P$ は monogenic である。また、$M_1, \cdots, M_n$ が monogenic のとき $M_1/\cap\cdots/\cap M_n$ を mono-closure といい、$M_1;\cdots;M_n$ と略記する。

推論規則は LK に加え、以下の 3 つを採用する。
$M,\ M_1,\cdots,M_m,\ M_1',\cdots,M_n'$ は monogenic 型とする。

(monogenic rule )

$$\frac{M_1/\ \rightarrow\ M_2/}{\rightarrow\ M_1/\subseteq M_2/}$$

(mono-closure rule 1 (mcr1))

$$\frac{M/\ \rightarrow\ M_1;\cdots;M_m/=M_1';\cdots;M_n'/}{\rightarrow\ M;M_1;\cdots;M_m/=M;M_1';\cdots;M_n'/}$$

(mono-closure rule 2 (mcr2))

$$\frac{M/\rightarrow M_1;\cdots;M_m/\ \subsetneqq\ M_1';\cdots;M_n'/}{M\neq\emptyset\ \rightarrow\ M;M_1;\cdots;M_m/\ \subsetneqq\ M;M_1';\cdots;M_n'/}$$

## 2.4 拍車

拍車は、プロセスのスケジューラを一般化した概念であり、envelope system では ($T_c$ の論理式で表される) es 集合として表現される。

拍車 $\dot{\gamma}$ について $(\dot{\gamma}/\cap\dot{\gamma}^c)/$ を $\gamma$ と表す。混乱が無ければ $\gamma$ を拍車と呼ぶことがある。

# 3 並行プログラム系の例

並行プログラム系の例として Dekker の解の飢餓回避問題 [1], [2] をとりあげる。以下に Dekker の解を簡単にしたもの (図) を示す。各プロセスは 4 つのブロックからなり、命題変数 $L_L$, $M_L$, $P_L$, $S_L$, $L_R$, $M_R$, $P_R$, $S_R$ がそれぞれ対応づけられている。プログラム変数 $T_L$ も命題変数として扱う。また、ブロック間の遷移について次の制限を満たすものとする。$F_1$, $F_2$ を以下のように定義する。

$$F_1 \stackrel{\mathrm{def}}{=} \neg(L_R \vee M_R)$$
$$F_2 \stackrel{\mathrm{def}}{=} (L_R \vee M_R)\wedge\neg T_L$$

左側のプロセスに関して、プログラムカウンタが $L_L(S_L)$ にある時は $F_1$, $F_2$ $(T_L)$ のいずれかが真になるまで拍車 $\dot{\alpha}$ は来ず、いずれかが真になったらそれが偽になる前に必ず拍車 $\dot{\alpha}$ が来るものとする。また、ブロック $M_L$, $P_L$ にあるときには拍車 $\dot{\alpha}$ は有限時間内に来るものとする。右側のプロセスに関しても同様とする。

## 3.1 プログラムの公理

左側のプロセスについてのプログラムの公理を挙げる。右側のプロセスは上と同様な読み替えを行う。

( ラベル間の関係 )

$$L_L/,\ F_1/\ \ \rightarrow\ \ \alpha = M_L/\wedge\alpha\ \supsetneqq\ \neg F_1/ \qquad (1)$$

$$L_L/,\ F_2/\ \ \rightarrow\ \ \alpha = S_L/\wedge\alpha\ \supsetneqq\ \neg F_2/ \qquad (2)$$
$$L_L/\ \ \rightarrow\ \ F_1 \supseteq \alpha,\ F_2/\supseteq\alpha \qquad (3)$$
$$M_L/\ \ \rightarrow\ \ \alpha = P_L/\ \wedge\ \alpha = \neg T_L/ \qquad (4)$$
$$P_L/\ \ \rightarrow\ \ \alpha = L_L/ \qquad (5)$$
$$S_L/,\ T_L/\ \ \rightarrow\ \ \alpha = L_L/ \qquad (6)$$

これらに、拍車の条件やラベルや変数の保存性の公理を加える。

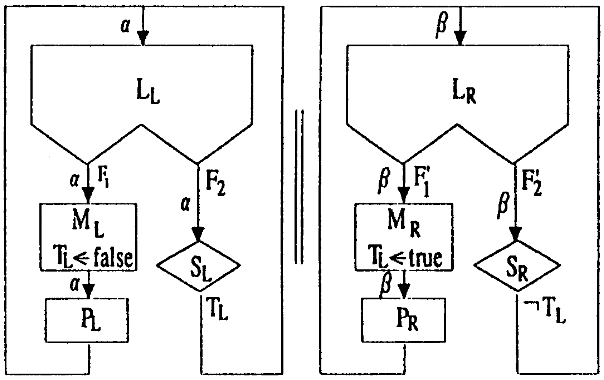


図　　Dekkerの解

## 3.2 飢餓回避

飢餓状態とは、$M_L, M_R$ のいずれかにコントロールが未来永劫到達しなくなることである。以下の 3 つを示すことにより、$M_L$ にいつかはコントロールが到達することを導くことができた。

- $L_L$ から動かないことはない。$(\rightarrow \neg L_L)$
- $S_L$ から動かないことはない。$(\rightarrow \neg S_L)$
- $L_L, S_L$ を繰り返すことはない。

$(S_L/\rightarrow\alpha\alpha=\neg S_L/)$

# 4 結論

束、位相の概念をもとにつくることにより、体系がきれいにまとまり、束論の直感がそのまま持ちこめるようになった。プログラムを公理化することにより数学の立場で算術として検証を行えるため、論理的に行うよりも簡単である。今後の課題としては、相対的な完全性を目指して公理の整備をすることと、実時間システムに対応できるように時刻を陽に扱えるように拡張することが挙げられる。

# References

[1] E. W. Dijkstra: Co-operating Sequential Process, *Programing Languages*, 1968, 43-112.

[2] 水谷 哲也, 五十嵐 滋, 小宮山 弘樹, 辻 尚史 : 一二の並行プロセスの検証問題について, 第 34 回プログラミングシンポジウム報告集, 1993, pp. 105–116.

[3] 白銀 哲也, 時間の論理の束モデルの拡張, 応用数学合同研究集会報告集, 1994,pp. 6-1 – 6-3.

[4] 塩 雅之, 時間の論理の束モデルの 2 次元的解釈, 応用数学合同研究集会報告集, 1994,pp. 5-1 – 5-6.