数理解析研究所講究録
第871巻 1994年 219-225

時間制約付 LOTOS の等価性の証明

中田 明夫 Akio NAKATA　東野 輝夫 Teruo HIGASHINO　谷口 健一 Kenichi TANIGUCHI

大阪大学基礎工学部情報工学科

1 まえがき

通信プロトコルや分散システムの仕様を記述するために，CCS[7], CSP[4], ACP[2], LOTOS[5]などのプロセス代数に基づいた形式仕様記述言語が提案されてきた．これらの言語は動作の順序を記述することはできるが，動作の定量的な時間制約，すなわち，各動作がある時点から具体的に何秒後から何秒後までの間に実行すべきか，ということを記述することはできなかった．しかし，実時間システムや実際の通信プロトコルなどでは定量的な時間制約を記述することが必要である．

本研究は，

1. 「動作がいつからいつまでに必ず実行される」という条件を直接指定できる．
2. 隣接しない動作間の時間制約を指定できる．
3. 等価性が決定可能である．

という要求を満足する言語を提案することを目的とする．時間制約を記述できる言語は現在までに多数提案されている[1, 3, 6, 8, 12, 13, 15, 17]．しかし，それらは上の条件のいずれかを満足しない．例えば，文献 [8, 13, 17] の提案は，上の 1. を，文献 [3, 6, 12] の提案は 2. を，文献 [1, 15] は 3. をそれぞれ満足しない．リアルタイムシステムの仕様記述に対して，時間制約を変更したときにシステムの等価性が保たれていることを証明するためには，1〜3を満たすことが望ましい。

本論文では，上の要求をすべて満足する言語 LOTOS/T[10, 9] を提案する．LOTOS/T は Basic LOTOS を時間制約が記述できるように拡張したものである．LOTOS/T では，時間制約は線形不等式の論理結合で記述する．このことによって，上の 1. が実現される．また，線形不等式は充足可能性が機械的に判定可能であるため，3. が実現される．さらに，動作の実行時刻が変数に代入可能で，その変数を後続の任意の動作の時間制約式で参照できるため，2. が実現される．

また，本論文では LOTOS と同様に LOTOS/T の動作式に対応するラベル付き遷移システム (LTS) を定義する推論規則を与える．単位時間進行は動作 tic で表現される．tic によって時間性を意味づける方法は，従来の意味論の簡潔で自然な拡張であり，他の多くの提案も同様の意味論を採用している．我々の提案はさらに，推論規則が適用できるかどうかの判定を線形不等式の充足可能性に帰着することにより機械的に行えるという特徴を持つ．従って，線形不等式の充足可能性を判定する手続きを利用すれば，LOTOS/T の動作式から機械的に LTS（一般には状態数が無限になる場合もあるが）を構成することができる．

LOTOS/T の記述に対して，時間的強 (弱) 双模倣等価性と，非時間的強 (弱) 双模倣等価性という 2 種類の等価性を導入する．時間的強 (弱) 双模倣等価性は，tic を観測可能としたときに，観測的に等価である 2 つの記述を等価とみなす．一方，非時間的強 (弱) 双模倣等価性は，tic を観測不能としたときに 2 つのシステムが観測的に等価であれば等価とみなす．もし動作式から構成した LTS が有限であれば，これらの等価性は文献 [14] のアルゴリズムを用いて判定できる（上述 3. の等価性が決定可能というのはこの意味であり，導出した LTS が無限の場合は LOTOS の場合と同様に必ずしも決定できるとは限らない）．しかし，LOTOS/T では時間制約の与え方によっては状態数が無限になり得る．しかも，多くの典型的な場合においては，本質的でない冗長な状態によって状態爆発が引き起こされる．そこで等価性の判定できるクラスを広げるために，構造が同じで時間制約のみが異なる LOTOS/T 記述に対して，等価な状態をまとめ，実用上多くの LOTOS/T 記述の LTS が有限になるような工夫を行なっている．時間制約を取り除いて Basic LOTOS 記述とみなした時に LTS が有限であるような LOTOS/T 記述であれば，実用上ほとんどの場合，この工夫によって有限 LTS を構成することができる．

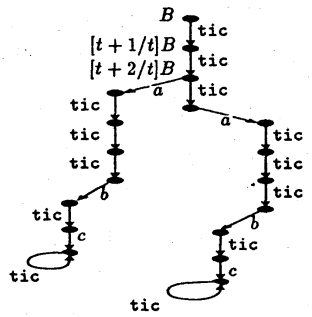

図 1: 動作式 Bの意味

本論文は以下のように構成される．2では言語 LOTOS/T の概略と構文について説明する．3では LOTOS/T の操作的意味を定義する推論規則について説明する．4では時間性に関連した等価性の定義を示し，等価性の機械的証明法，および，LTS の状態数を有限に抑えるための工夫を述べる．5では本論文のまとめと今後の課題を述べる．

2 構文

まず最初に，LOTOS/T の概略を説明する．

例 1

$$B = a[2 \leq t \leq 3 \wedge x_0 = t]; b[t = x_0 + 3]; c[t = x_0 + 4]; \mathbf{stop}$$

動作式 Bは動作 a を 2 単位時間後から 3 単位時間後の間に必ず実行し，動作 b を a の実行時から 3 単位時間後に実行し，さらに動作 c を a の実行時から 4 単位時間後に実行するプロセスを表す． □

tは動作式が表すプロセスが動作を開始してからの時刻を保持する特別な変数である．述語 $x_0 = t$は，動作 a を実行した時刻を変数 x_0

に代入することを表す．我々はBの意味モデルのLTSが図1のようになることを意図している．このLTSは以下のようにして得られる．図1の木状のLTSにおいて，根節点は動作式Bに対応する．動作式Bでは最初に単位時間進行動作tickのみが実行可能である．なぜなら，Bの先頭動作aに付随する述語「$2 \leq t \leq 3 \wedge x_0 = t$」($a$の時間制約を表す)は時刻0，すなわち$t=0$で偽である．ゆえに，枝$\xrightarrow{\texttt{tic}}$が根節点から出される．tickが実行されると時刻が1単位時間進行する．このとき，我々はtickの実行後の状態を動作式$[t+1/t]B$，すなわち$a[2 \leq t+1 \leq 3 \wedge x_0 = t+1]; b[t+1 = x_0+3]; c[t+1 = x_0+4]; \mathbf{stop}$で表す方法を採用する．ここで，$[e/x]B$は動作式$B$中の変数$x$のすべての出現を式$e$で置き換えた動作式を表す．すなわち，例えば動作$a$は$B$においてはあと2～3単位時間経過すれば実行可能になるが，Bから1単位時間進行した状態ではあと1～2単位時間経過すれば実行可能になるため，tを$t+1$で置き換えることによりこのことを表現する．これに伴い，動作aの実行可能性は常にaに付随する述語が$t=0$で充足可能であるか否かで決定される．このように，tは構文の上では時刻を表すが，意味論の上では時間制約を時間進行に合わせて動的に更新するための目印として働く．しかし，記述者はそのような働きを意識する必要はなく，単にtを時刻と扱えば良い．

状態$[t+1/t]B$でも同様にticのみが実行可能である．よって，$[t+1/t]B \xrightarrow{\texttt{tic}} [t+2/t]B$が付け加えられる．状態$[t+2/t]B$では，ticと$a$が実行可能である．もしticが実行されるなら，$[t+3/t]B$，つまり，$a[2 \leq t+3 \leq 3 \wedge x_0 = t+3]; b[t+3 = x_0+3]; c[t+3 = x_0+4]; \mathbf{stop}$が得られる．ここで，動作$a$は$B$の表す状態から3単位時間後までには必ず実行されなければならないという制約のため，ticは$[t+3/t]B$において実行できない．この場合，動作aはこの時点で緊急性(urgency[3, 6])を持つという．このことは，aに付随する述語「$2 \leq t+3 \leq 3 \wedge x_0 = t+3$」が1以上の$t$の充足解を持たないことより決定される．よって，$[t+3/t]B$では$a$のみが実行可能である．もし$a$が実行されたなら，変数$x_0$に$t=0$における$x_0$の充足解が代入される．この場合では$x_0 = t+3$であるから，その結果，変数$x_0$には3という値が代入される．そして，$b[t+3 = 3+3]; c[t+3 = 3+4]; \mathbf{stop}$が次の状態として得られる．したがって，$b$はこの瞬間から3単位時間後に実行され，$c$はこの瞬間から4単位時間後に実行されることになる．ここで，aに付随する述語の最初の形「$2 \leq t \leq 3 \wedge x_0 = t$」は$t=3$で充足可能であることに注意されたい．すなわち，この述語が$t=3$で充足可能であることは，この述語のtを$t+3$で置き換えた述語が$t=0$で充足可能であることと等価である．したがって，x_0に3が代入されることは，最初の形の述語の意味とも合致する．

次に，LOTOS/Tによる，時間制約のある動作とない動作の混在した記述，および，再帰プロセスの記述例を示す．

例 2

1. $E = a[x = t]; b; c[t \geq x+2]; stop$
2. $P = a[t = 5]; stop [] b[t = 1]; P$

最初の動作式は，時間制約のない動作が時間制約のある動作の間にはさまれている例である．時間制約のない動作bはいつでも実行可能，すなわち，時間制約として$true$が与えられているとみなされる．この例ではさらに，無限区間$(t \geq x+2)$が時間制約として与えられている．2番目の動作式は，再帰プロセスの記述例である．Pが呼ばれるたびに時刻は0にリセットされる．一般に時刻は各プロセス毎に（自分自身を呼び出した時にはそれぞれのインスタンス毎に）異なり，常にそのプロセスが走りはじめた時刻が0となる．対応するLTSを図2に示す． □

LOTOS/Tの構文の形式的定義は以下に与えられる．

定義 1 LOTOS/Tの動作式は以下のように定義される(演算子の優先順位はLOTOSと同様)．

$$
\begin{aligned}
E \ ::= \ & \mathbf{stop} \quad \text{(非時間的デッドロック)} \\
| \ & \mathbf{exit} \quad \text{(正常終了)} \\
| \ & a; E \quad \text{(非時間的動作プレフィクス)} \\
| \ & a[P(t,\bar{x})]; E \quad \text{(時間的動作プレフィクス)} \\
| \ & E[]E \quad \text{(選択)} \\
| \ & E|||E \quad \text{(非同期並列)} \\
| \ & E||E \quad \text{(同期並列)} \\
| \ & E|[A]|E \quad \text{(並列合成)} \\
| \ & E[> E \quad \text{(割り込み)} \\
| \ & E >> E \quad \text{(逐次合成)} \\
| \ & \mathbf{hide}\ A\ \mathbf{in}\ E \quad \text{(隠蔽)} \\
| \ & P[g_1, \ldots, g_k](\bar{e}) \quad \text{(プロセス呼び出し)}
\end{aligned}
$$

ただし，$a \in Act \cup \{i\}$ (Actはすべての観測可能な動作の有限集合を，iは内部動作を表す)，$A \subset Act$, $k \in \mathbf{N}$, そして，$P(t,\bar{x})$は，t(動作式が実行を始めてからの経過時間を表す)と$\bar{x}$ ($\bar{x}$は変数のベクトルを表す)を自由変数[1]として持つ述語を表す．$\bar{e}$は式のベクトルを表す．Varは述語で用いるすべての変数の集合を指す． □

3 操作的意味

本節では，動作式間の遷移関係を与える推論規則によって，LOTOS/Tの操作的意味を形式的に定義する．LOTOS/Tの操作的意味はBasic LOTOSの拡張になっている．その違いは動作ticに関する状態遷移の扱いにある．

3.1 無動作プロセス

LOTOS/Tでは，動作式**stop**が**非時間的デッドロック**(non-temporal deadlock[8])を表わすように意味を拡張している[表1規則 (1)]．ここで非時間的デッドロックとは，無限に時間が経過する(すなわち，無限回ticを実行する)以外の如何なる動作も行なえない状態を指す．動作式**exit**の意味は，δ[2]を実行する前に何回でもticを実行できるように拡張される[規則 (3)]．

3.2 動作プレフィクス

動作式$a[P(t,\bar{x})]; B$は，ある$\bar{x}$への代入値$\bar{c}$が存在して，$P(0,\bar{c})$が成り立つ時，動作aが実行可能で，動作式$[\bar{c}/\bar{x}]B$へ遷移すると定義する[規則 (4)]．

また，1以上のあるtの値に対して$P(t,\bar{x})$が充足可能となる時，動作ticが実行可能で，tを$t+1$で置き換えた動作式に遷移すると定義する[規則 (5)]．

時間制約なしの動作式$a; B$の意味は，$a[true]; B$と同じである[規則 (6),(7)]

[1] 自由変数，束縛変数の定義は1階述語論理における定義と同様である．
[2] 動作δはBasic LOTOS同様に正常終了を表す．

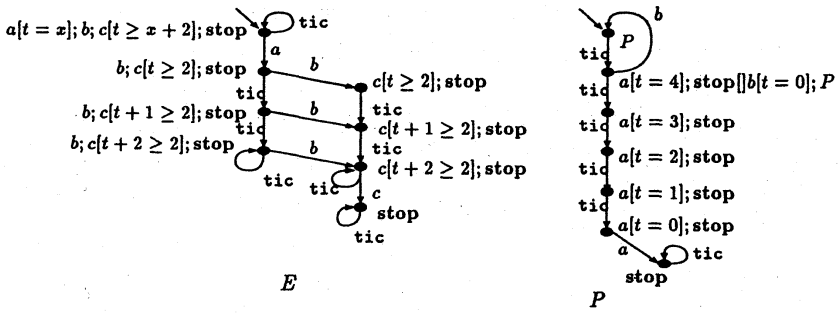

図 2: 動作式 E, Pの意味

表 1: 動作の遷移関係を導出する推論規則 (1)

Inaction			
$\dfrac{}{\mathbf{stop} \xrightarrow{\mathtt{tic}} \mathbf{stop}}$	(1)	$\dfrac{}{\mathbf{exit} \xrightarrow{\delta} \mathbf{stop}}$	(2)
$\dfrac{}{\mathbf{exit} \xrightarrow{\mathtt{tic}} \mathbf{exit}}$	(3)		
Action Prefix			
$\dfrac{P(0,\bar{c})}{a[P(t,\bar{x})];B \xrightarrow{a} [\bar{c}/\bar{x}]B}$	(4)	$\dfrac{\exists t'\exists \bar{x}[t'>0 \wedge P(t',\bar{x})]}{a[P(t,\bar{x})];B \xrightarrow{\mathtt{tic}} a[P(t+1,\bar{x})];[t+1/t]B}$	(5)
$\dfrac{}{a;B \xrightarrow{a} B}$	(6)	$\dfrac{}{a;B \xrightarrow{\mathtt{tic}} a;[t+1/t]B}$	(7)
Internal Action			
$\dfrac{P(0,\bar{c})}{i[P(t,\bar{x})];B \xrightarrow{i} [\bar{c}/\bar{x}]B}$	(8)	$\dfrac{\neg P(0,\bar{x}) \quad \mathcal{F}P(1)}{i[P(t,\bar{x})];B \xrightarrow{\mathtt{tic}} i[P(t+1,\bar{x})];[t+1/t]B}$	(9)
$\dfrac{}{i;B \xrightarrow{i} B}$	(10)		
Choice			
$\dfrac{B_1 \xrightarrow{\beta} B_1'}{B_1[]B_2 \xrightarrow{\beta} B_1'}$ iff $\beta \in Act \cup \{\delta, i\}$	(11)	$\dfrac{B_2 \xrightarrow{\beta} B_2'}{B_1[]B_2 \xrightarrow{\beta} B_2'}$ iff $\beta \in Act \cup \{\delta, i\}$	(12)
$\dfrac{B_1 \xrightarrow{\mathtt{tic}} B_1' \quad B_2 \xrightarrow{\mathtt{tic}} B_2'}{B_1[]B_2 \xrightarrow{\mathtt{tic}} B_1'[]B_2'}$	(13)		
$\dfrac{B_1 \xrightarrow{\mathtt{tic}} B_1' \quad B_2 \not\xrightarrow{\mathtt{tic}}}{B_1[]B_2 \xrightarrow{\mathtt{tic}} B_1'}$	(14)	$\dfrac{B_2 \xrightarrow{\mathtt{tic}} B_2' \quad B_1 \not\xrightarrow{\mathtt{tic}}}{B_1[]B_2 \xrightarrow{\mathtt{tic}} B_2'}$	(15)

3.3 内部動作プレフィクス

動作式 $i[...];B$などにおける内部動作 i は常に緊急性を持つと仮定する (maximal progress assumption[17]). したがって, 内部動作はいったん実行可能になると, 常に tic より先に実行される [規則 (9),(10)]. それ以外は観測可能動作と同じである.

3.4 選択

我々は選択演算子を弱選択 (weak-choice[8]) と定義する. 例えば, 動作式 "$a[t = 1];$**stop**" と "$b[t = 2];$**stop**" は, あえて非形式的に従来の LOTOS の記法を用いれば, それぞれ "**tic**;a;**stop**" と "**tic**;**tic**;b;**stop**" と表現できる. しかしながら, "$a[t = 1];$**stop**$[]b[t = 2];$**stop**" は "**tic**;a;**stop**[]**tic**;**tic**;b;**stop**" とは表現されない. なぜなら, 時刻 0 で非決定的選択が生じるからである. そうではなくて, この動作式は "**tic**; (a;**stop** [] **tic**;b;**stop**)" と表現される必要がある. 規則 (11)〜(15) は後者の意味モデルを構成するために導入される.

3.5 並列合成

LOTOS/T では, 並列合成 (|||, ||, |[A]|) において tic は常に同期実行する [表 2規則 (17)]. その結果, 複数のプロセスで同期する動作の時間制約は, その動作の各プロセスにおける時間制約の論理積をとったものとなる.

例 3　動作式 $a;b[2 \le t \le 4];$**stop**$|[b]|c;b[3 \le t \le 5];$**stop** において, 動作 b の時間制約は $3 \le t \le 4$.

3.6 割り込み

割り込みにおいては, tic は本プロセスと割り込みプロセスの間で常に同期し, 動作の選択には関与しない (すなわち, パーシステント) [規則 (25)]. 割り込みを発生させる明示的な動作が指定されることが普通と判断し, このように定義した. その結果, 割り込みプロセスの先頭動作に緊急性をもつ動作を指定すると, 本プロセスの動作はその動作の実行と同時に打ち切られる.

表 2: 動作の遷移関係を導出する推論規則 (2)

Parallel	
$\dfrac{B_1 \xrightarrow{\beta} B_1' \quad B_2 \xrightarrow{\beta} B_2'}{B_1 \|[A]\| B_2 \xrightarrow{\beta} B_1' \|[A]\| B_2'}$ iff $\beta \in A \cup \{\delta\}$ (16)	$\dfrac{B_1 \xrightarrow{\mathtt{tic}} B_1' \quad B_2 \xrightarrow{\mathtt{tic}} B_2'}{B_1 \|[A]\| B_2 \xrightarrow{\mathtt{tic}} B_1' \|[A]\| B_2'}$ (17)
$\dfrac{B_1 \xrightarrow{a} B_1'}{B_1 \|[A]\| B_2 \xrightarrow{a} B_1' \|[A]\| B_2}$ iff $a \notin A \vee a = i$ (18)	$\dfrac{B_2 \xrightarrow{a} B_2'}{B_1 \|[A]\| B_2 \xrightarrow{a} B_1 \|[A]\| B_2'}$ iff $a \notin A \vee a = i$ (19)
$\dfrac{B_1 \|[\emptyset]\| B_2 \xrightarrow{\alpha} B'}{B_1 \|\| B_2 \xrightarrow{\alpha} B'}$ iff $\alpha \in Act \cup \{\delta, \mathtt{tic}, i\}$ (20)	$\dfrac{B_1 \|[Act]\| B_2 \xrightarrow{\alpha} B'}{B_1 \|\| B_2 \xrightarrow{\alpha} B'}$ iff $\alpha \in Act \cup \{\delta, \mathtt{tic}, i\}$ (21)
Disable	
$\dfrac{B_1 \xrightarrow{a} B_1'}{B_1 [> B_2 \xrightarrow{a} B_1' [> B_2}$ (22)	$\dfrac{B_2 \xrightarrow{\beta} B_2'}{B_1 [> B_2 \xrightarrow{\beta} B_2'}$ iff $\beta \in Act \cup \{\delta, i\}$ (23)
$\dfrac{B_1 \xrightarrow{\delta} B_1'}{B_1 [> B_2 \xrightarrow{\delta} B_1'}$ (24)	$\dfrac{B_1 \xrightarrow{\mathtt{tic}} B_1' \quad B_2 \xrightarrow{\mathtt{tic}} B_2'}{B_1 [> B_2 \xrightarrow{\mathtt{tic}} B_1' [> B_2'}$ (25)
Enable	
$\dfrac{B_1 \xrightarrow{a} B_1'}{B_1 >> B_2 \xrightarrow{a} B_1' >> B_2}$ (26)	$\dfrac{B_1 \xrightarrow{\delta} B_1'}{B_1 >> B_2 \xrightarrow{i} B_2}$ (27)
$\dfrac{B_1 \xrightarrow{\mathtt{tic}} B_1' \quad B_2 \xrightarrow{\mathtt{tic}} B_2' \quad B_1 \not\xrightarrow{\delta}}{B_1 >> B_2 \xrightarrow{\mathtt{tic}} B_1' >> B_2'}$ (28)	
Hide	
$\dfrac{B \xrightarrow{\beta} B'}{\mathtt{hide}\ A\ \mathtt{in}\ B \xrightarrow{\beta} \mathtt{hide}\ A\ \mathtt{in}\ B'}$ iff $\beta \in (Act - A) \cup \{\delta, i\}$ (29)	
$\dfrac{B \xrightarrow{a} B'}{\mathtt{hide}\ A\ \mathtt{in}\ B \xrightarrow{i} \mathtt{hide}\ A\ \mathtt{in}\ B'}$ iff $a \in A$ (30)	$\dfrac{B \xrightarrow{\mathtt{tic}} B' \quad B \not\xrightarrow{\delta}\ \text{for all } a \in A}{\mathtt{hide}\ A\ \mathtt{in}\ B \xrightarrow{\mathtt{tic}} \mathtt{hide}\ A\ \mathtt{in}\ B'}$ (31)
Process Invocation	
$\dfrac{[\bar{e}/\bar{x}]B\{g_1'/g_1, \ldots, g_k'/g_k\} \xrightarrow{\alpha} B'}{P[g_1', \ldots, g_k'](\bar{e}) \xrightarrow{\alpha} B'}$ iff $\alpha \in Act \cup \{\mathtt{tic}, \delta, i\} \wedge P[g_1, \ldots, g_k](\bar{x}) := B$ is a definition (32)	

例 4　動作式 $P\,[>\, b[t = 10]; \mathbf{exit}$ において，プロセス P の実行は時刻 10 で動作 b によって打ち切られる．

3.7 逐次合成

逐次合成においても，tic は先行プロセスと後続プロセスとの間で同期する [規則 (28)]．さらに，後続プロセスの呼び出しは，常に緊急性を持って（すなわち，tic に先行して）実行される [規則 (28)]．

3.8 隠蔽

hide 文による動作の隠蔽では，隠蔽された動作は緊急性をもつ [規則 (31)]．これによって，同期通信における最小遅延 (minimal delay[6]) の概念をモデル化することができる．

例 5　動作式 $a[t \geq 5]; \mathbf{stop} \|[a]\| a[t \geq 10]; \mathbf{stop}$ では，動作 a は時刻 10 以降いつでも実行可能である．しかし，動作式 hide a in $(a[t \geq 5]; \mathbf{stop} \|[a]\| a[t \geq 10]; \mathbf{stop})$ では，動作 a は実行可能になってすぐ (すなわち，時刻 10 に) 実行される．

3.9 プロセス呼び出し

プロセス呼び出しではプロセスパラメータが同じであれば，いかなる時刻で実行されても正確に同じ動作をすることを保証するため，プロセスの実行開始時に時刻は 0 にリセットされる．

4 等価性とその検証法

本節では，時間性に関連して，双模倣等価性[11]に基づいた 2 種類の等価性を導入する．

4.1 時間的双模倣等価性

時間的双模倣等価性は 2 つのプロセスの動作の実行時刻が一致することを要求する等価性である．定義は文献 [11] の強双模倣等価性と同様である．

定義 2　動作式集合上の関係 $\mathcal{R}$ が以下の条件を満たす時，$\mathcal{R}$ を**時間的強双模倣関係**と呼ぶ．

- もし $B_1 \mathcal{R} B_2$ ならば，任意の $a \in Act \cup \{\delta, \mathtt{tic}\}$ に対して以下の 2 条件が成立する：
  1. もし $B_1 \xrightarrow{a} B_1'$ ならば，$\exists B_2' [B_2 \xrightarrow{a} B_2'$ かつ $B_1' \mathcal{R} B_2']$
  2. もし $B_2 \xrightarrow{a} B_2'$ ならば，$\exists B_1' [B_1 \xrightarrow{a} B_1'$ かつ $B_1' \mathcal{R} B_2']$ □

定義 3　動作式 B, B' に対して，もし，$B \mathcal{R} B'$ となるような時間的強双模倣関係 $\mathcal{R}$ が存在するならば，B と B' は**時間的強双模倣等価**であるといい，$B \sim_t B'$ と表記する．□

観測不能な内部動作を考慮した時間的弱双模倣等価性（$\approx_t$ と表記する）も従来と同様に定義される．

4.2 非時間的双模倣等価性

本節では，tic を内部動作とみなした等価性である，非時間的双模倣等価性を定義する．この等価性は，時間制約を記述した二つの動作式が，細かい時間的タイミングを無視した時に観測的に区別できないならば，それらを等価と見なす．

定義 4 任意の $a \in (Act \cup \{\delta\} - \{\mathtt{tic}\}) \cup \{\epsilon\}$ に対して，関係 $\stackrel{a}{\Longrightarrow}$ を以下のように定義する．

$$B \stackrel{a}{\Longrightarrow} B' \stackrel{\mathrm{def}}{=} \begin{cases} B(\stackrel{\mathtt{tic}}{\longrightarrow})^* \stackrel{a}{\longrightarrow} (\stackrel{\mathtt{tic}}{\longrightarrow})^* B', \\ \quad a \in Act \cup \{\delta\} - \{\mathtt{tic}\} \\ B(\stackrel{\mathtt{tic}}{\longrightarrow})^* B', \qquad a = \epsilon \end{cases}$$

□

定義 5 動作式集合上の関係 $\mathcal{R}$ が以下の条件を満たす時，$\mathcal{R}$ を非時間的強双模倣関係と呼ぶ：

- もし $B_1 \mathcal{R} B_2$ ならば，任意の $a \in (Act \cup \{\delta\} - \{\mathtt{tic}\}) \cup \{\epsilon\}$ に対して，以下の条件が成り立つ：
  1. もし $B_1 \stackrel{a}{\Longrightarrow} B_1'$ ならば，$\exists B_2'[B_2 \stackrel{a}{\Longrightarrow} B_2'$ かつ $B_1' \mathcal{R} B_2']$
  2. もし $B_2 \stackrel{a}{\Longrightarrow} B_2'$ ならば，$\exists B_1'[B_1 \stackrel{a}{\Longrightarrow} B_1'$ かつ $B_1' \mathcal{R} B_2']$ □

定義 6 動作式 B, B' に対して，もし，$B \mathcal{R} B'$ となるような非時間的強双模倣関係 $\mathcal{R}$ が存在するならば，B と B' は**非時間的強双模倣等価**であるといい，$B \sim_u B'$ と表記する． □

観測不能な内部動作を考慮した非時間的弱双模倣等価性（$\approx_u$ と表記する）も従来と同様に定義される．

命題 1 時間的強 (弱) 双模倣等価である動作式は非時間的強 (弱) 双模倣等価である．すなわち，

$$\begin{aligned} B \sim_t B' &\Rightarrow B \sim_u B' \\ B \approx_t B' &\Rightarrow B \approx_u B' \end{aligned}$$

□

命題 2 $\sim_u$ は合同関係でない．例えば：

$$\exists B, B_1, B_2[(B_1 \sim_u B_2) \wedge (B[]B_1 \not\sim_u B[]B_2)]$$

(証明) $B_1 = a[t=0];\mathbf{stop}$, $B_2 = a[t=2];\mathbf{stop}$, $B = b[t=1];\mathbf{stop}$ と選べばよい． □

命題 2より，非時間的双模倣等価性は公理的な方法で証明することは困難である．

4.3 検証法

述語 $P(t,\bar{x})$ と $\exists t' \exists \bar{x}[t' > 0 \wedge P(t',\bar{x})]$ はプレスブルガー文であるので，いずれも充足可能性を機械的に判定できる．したがって，表 1,2の推論規則を適用すれば，遷移の有無が $P(t,\bar{x})$ や $\exists t' \exists \bar{x}[t' > 0 \wedge P(t',\bar{x})]$ の充足可能性判定に帰着するため，各節点が動作式に対応する LTS（一般には状態数無限であるが）を機械的に構成できる．まず，図 2のプロセス E に対する規則の適用例を以下に示す．

- $E = a[t=x];b;c[t \geq x+2];\mathbf{stop} \stackrel{a}{\longrightarrow} b;c[t \geq 2];\mathbf{stop}$ [規則 (4) より],
- $b;c[t \geq 2];\mathbf{stop} \stackrel{\mathtt{tic}}{\longrightarrow} b;c[t+1 \geq 2];\mathbf{stop}$ [規則 (5) より],

図 2のプロセス P に対しては次のようになる．

- $P \stackrel{\mathtt{tic}}{\longrightarrow} a[t=4];\mathbf{stop}[]b[t=0];P$ [規則 (32), (13), (5) より],
- $a[t=4];\mathbf{stop}[]b[t=0];P \stackrel{\mathtt{tic}}{\longrightarrow} a[t=3];\mathbf{stop}$ [規則 (14), (5) より],

ところが，プロセス E のように先頭動作の時間制約が有界でない場合，規則 (5) の適用によって，$[t+1/t]E \stackrel{\mathtt{tic}}{\longrightarrow} [t+2/t]E \stackrel{\mathtt{tic}}{\longrightarrow} \ldots$ という，無限列が生成される．各 $[t+k/t]E$ $(k=1,2,\ldots)$ はいずれも文字の系列として等しくないので，動作式そのものを状態とみなす 3の意味論では，プロセス E のような単純なプロセスに対してさえも状態爆発が起こる．しかし，各 $[t+k/t]E$ は時間制約の述語の部分以外は全く同じ形をしていて，かつ，任意の非負整数 t に対する述語の充足可能性は変化していない．そこで，以下では時間制約の述語の充足可能性が一致する場合に，異なる動作式を 1つの状態に縮約する方法を提案する．

もし，E と $[t+1/t]E$ の対応する述語の充足可能性が $0 \leq t < \infty$ の範囲の t の値に対して等しいならば，その二つの動作式は同じ状態を表現しているとみなすことができる．例えば，図 2の E に関しては，推論規則より $a[t=x];b;c[t \geq x+2];\mathbf{stop} \stackrel{\mathtt{tic}}{\longrightarrow} a[t+1=x];b;c[t+1 \geq x+2];\mathbf{stop}$ が導かれる．ここで，動作 a に付随する二つの述語 $t=x, t+1=x$ の充足可能性は次の意味で等しい．

$$\forall t[0 \leq t \Rightarrow \exists x[t=x] \equiv \exists x'[t+1=x']]$$

("$t=x$" の中の x と "$t+1=x$" の中の x は同じ値を表さない．したがって，我々は後者の述語を "$t+1=x'$" と書く)．

さらに，上式を充足する，x と x' に対する任意の値の割り当て v,v' に対して，以下の式が成り立つ．

$$\forall t'[0 \leq t' \Rightarrow [t' \geq v+2] \equiv [t'+1 \geq v'+2]]$$

まとめると，

$$\begin{aligned} &\forall t_1[0 \leq t_1 \Rightarrow [\exists x(t_1=x) \equiv \exists x'(t_1+1=x')] \wedge \\ &\quad \forall x \forall x'[(t_1=x) \wedge (t_1+1=x') \Rightarrow \\ &\quad \forall t_2[0 \leq t_2 \Rightarrow \\ &\qquad [(t_2 \geq x+2) \equiv (t_2+1 \geq x'+2)]]]] \end{aligned}$$

が E と $[t+1/t]E$ の対応する述語の充足可能性が $0 \leq t \leq \infty$ の範囲の t の値に対して等しいことを表す論理式となる．以下この論理式を $\Phi(E,[t+1/t]E)$ と書く．この論理式が充足可能ならば，$E = a[t=x];b;c[t \geq x+2];\mathbf{stop}$ と $[t+1/t]E = a[t+1=x];b;c[t+1 \geq x+2];\mathbf{stop}$ が等価な状態を表していることになる．この論理式はプレスブルガー文であるので，充足可能性は決定可能である．この論理式の一般形 $\Phi(B_1,B_2)$ は文献 [9] に示す．

このように，tic 遷移の前後の動作式が意味的に等価であるかどうかを機械的に検証することが可能であるため，状態爆発を防ぎながら LTS を機械的に構成することが可能である．このようにして得られた LTS が有限であれば，文献 [14] のアルゴリズムを適用することによって，時間的双模倣等価性，および，非時間的双模倣等価性を検証することが可能である．

ただし，全ての動作式が有限 LTS に変換出来るわけではない．時間制約を記述しない動作式 (Basic LOTOS) でも状態数が無限になり得るのは既知の結果である．また，有限 LTS に変換可能な Basic LOTOS の動作式に時間制約を付加した場合でも，対応する有限 LTS が存在しない，すなわち，状態数が本質的に無限になる場合がある．以下の例を参照されたい．

例 6 図 3に示す例において，状態 $b[t=1];\mathbf{stop}$ は決して $b[t=2];\mathbf{stop}$ や $b[t=3];\mathbf{stop}$ などと強等価にならないため，無限に新しい状態が導入される． □

$$
\begin{array}{rcl}
a[x=t];b[t-2x=1];\mathtt{stop} & \xrightarrow{a} & b[t=1];\mathtt{stop} \\
\downarrow \mathtt{tic} & & \\
a[x=t+1];b[t+1-2x=1];\mathtt{stop} & \xrightarrow{a} & b[t=2];\mathtt{stop} \\
\downarrow \mathtt{tic} & & \\
a[x=t+2];b[t+2-2x=1];\mathtt{stop} & \xrightarrow{a} & b[t=3];\mathtt{stop} \\
\vdots & &
\end{array}
$$

図 3: 状態数が本質的に無限になる例

```
ONE_KEY_CONTROLLER[p,r,lc,sc,dc]
     := p[t1p=t];
          (lc[t1p+d1<=t<=t1p+d4];r;ONE_KEY_CONTROLLER
           [] r[t<t1p+d1];
               (p[t<t1p+d2 and t2p=t];
                     (r[t<t2p+d3];dc[t2p+d3<=t<=t1p+d4];
                                        ONE_KEY_CONTROLLER
                     [] slc[t2p+d3<=t<=t1p+d4];
                                r;ONE_KEY_CONTROLLER)
                [] sc[t1p+d2<=t<=t1p+d4];exit)
          )
+ variables
  t1p: time when the first press occurred.
  t2p: time when the second press occurred.
+ constants
  d1: threshold for the first short or long click
  d2: timeout for the second click
  d3: threshold for the second short or long click
  d4: required maximum total delay between button press
      and result action
```

図 4: 1 ボタン制御器の時間制約記述

4.4 記述の実際例と検証の応用

図 4の例は入力として 1 つの押しボタンのみを持ち，ボタンを押す時間的タイミングのパターンに応じて，4 種類の出力動作を行なう遠隔制御器を記述している．時間的タイミングのパターンは以下の 4 種類である：

- 1 回の長いクリック
- 1 回の短いクリック
- 2 回の短いクリック
- 1 回の短いクリックの後，1 回の長いクリック

以上のパターンのうち，1 回の短いクリックはプロセスを終了させるのに用いる．それ以外は何回でも入力を受け付ける．押しボタンの動作は動作p ('press' の略) と r ('release' の略) の系列で表現される．上述の 4 つに対応する出力動作はそれぞれlc ('long click' の略), sc ('short click' の略), dc ('double click' の略) slc ('short and long click' の略) である．

図 4においてd1 は 1 回目のクリックが長いか短いかを区別するしきい値，d2 は 2 回目のクリックを待つタイムアウト値，d3 は 2 回目のクリックが長いか短いかを区別するしきい値，d4 は 1 回目のクリック開始から出力動作が行なわれるまでの遅延時間の最大値である．

```
UNTIMED_ONE_KEY_CONTROLLER[p,r,lc,sc,dc]
     :=p;(i;lc;r;UNTIMED_ONE_KEY_CONTROLLER
          [] r;(p;(r;dc;UNTIMED_ONE_KEY_CONTROLLER
                   []i;slc;r;UNTIMED_ONE_KEY_CONTROLLER)
                []i;sc;exit)
          )
```

図 5: 1 ボタン制御器の時間制約を持たない仕様記述

このプロセスは次のように動作する．まず，最初の動作p(図 4: 2 行目) から時間がd1 以上d4 以下経過したならば，動作lc を実行し，r を待って初期状態に戻る (3 行目)．しかし，もし最初の動作p からd1 経過する前に動作r が実行されると (4 行目)，最初のp の実行時点からd2 以上d4 以下経過したならば，動作sc を実行してプロセスの実行を終了する (8 行目)．あるいは，2 番目のp が最初のp からd2 経過する前に実行されたならば (5 行目)，2 番目のp からd3 以上d4 以下経過した時点で動作slc を実行し，r を待って初期状態に戻る (7 行目)．2 番目のr が 2 番目のp からd3 以内に実行されたならば，動作dc を実行して初期状態に戻る (6 行目)．各動作lc,sc,slc,dc はいずれも最初のp の実行時からd4 経過した時点で緊急性を持つため，遅くとも必ずそれまでに実行される．

図 4の例において，もし d2+d3>d4 ならば，次に行なう動作の時間制約が決して満たされない状態 (時間的デッドロック (temporal deadlock)[8]と呼ぶ) が存在する．また，もしd1>=d2 ならば，2 回目のクリックが動作として認識されない場合が生じる．このことは，上の条件を満たすある値をd1,d2,d3, d4 に代入して，対応する LTS を構成すれば容易に検証できる．LTS においては，時間制約が満たされなくなった状態は出ていく弧をまったく持たない．時間制約によってプロセスの実行可能な動作が変化したか否かは，図 5 に示すような，時間制約を持たない仕様との非時間的双模倣等価性を調べることによって検証できる．

4.5 等価性検証ツールの効率評価

我々の研究グループは本章で提案した等価性判定法を計算機上で実現した．その結果を図 6，表 3に示す．図 4に基づいた 4 番目のテストデータによる判定結果から，複雑な時間制約が指定されていて，時間制約を微妙に変更した場合に等価性が保たれるかどうかが自明でない場合も，比較的短時間で (Sony NEWS-5000 上で 1 分～1 分半) 検証できることがわかった．ただし，LTS の状態数は動作式の複雑さと比べて非常に大きくなっている．このことは，本章の等価性判定法においては，動作式から構成する LTS の大きさが動作式の時間制約に出現する時刻定数の絶対値に比例して大きくなることが原因である．

5 あとがき

本論文では，Basic LOTOS の拡張である言語 LOTOS/T を提案した．LOTOS/T によって，原始的な演算子を複雑に組み合わせることなく，隣接しない動作間の時間制約を整数上の線形不等式を用いて柔軟に記述することが可能になった．同時に，LTS の構成を機械的に行なう枠組を提示し，LTS が有限であれば等価性を機械的に証明することが可能であることを示した．しかし，単位時間進行を tic で表すため，時間制約の記述によっては，LTS が有限状態であっても状態数が非常に大きくなる場合がある．

今後の課題は以下の通りである．

1. 意味論を改良し，時間制約の情報が LTS の枝の数や状態数に反映しないようにする．すなわち，時間進行以外の真の動作の複雑さのみが LTS に反映されるようにする．
2. 時間性に関する，より抽象度の高い構文 (例えば，タイムアウトや「番犬 (watchdog)」など) を導入し，記述性を高める．

参考文献

[1] Baeten, J. C. M. and Bergstra, J. A.: Real Time Pro-

表 3: 図 6のテストデータに対する等価性判定結果および効率

	状態数 (stop を除く)			LTS 作成時間 (cpu 秒)	等価性判定時間 (cpu 秒) 時間的	非時間的
(1)	A= 4,	B= 4,	計 8	0.020 ~ 0.100 程度	約 0.010(yes)	約 0.010 (yes)
(2)	B= 12,	C= 12,	計 24	0.100 ~ 0.150 程度	約 0.010(yes)	約 0.010 (yes)
(3)	C= 12,	D= 7.	計 19	0.070 ~ 0.150 程度	約 0.010 (no)	約 0.010 (yes)
(4)	P1= 2256,	P2= 1574	計 3830	66.350 ~ 67.260 程度	約 7.880 (no)	約 82.980 (yes)
	P1= 2256,	P3= 696	計 2952	51.550 ~ 53.370 程度	約 2.960 (no)	約 12.600 (no)
	P1= 2256,	P4= 1644	計 3900	67.840 ~ 68.070 程度	約 8.490 (no)	約 63.180 (no)
	P3= 696,	P4= 1664	計 2340	40.590 ~ 40.880 程度	約 2.370 (no)	約 12.100 (no)

```
1)
A:=a[2<=t and t<=3];stop
と
B:=a[t=2];stop [] a[t=3];stop
2)
B:=a[2<=t and t<=3 and x0=t];b[t=x0+3];B
と
C:=a[t=2];b[t=5];C [] a[t=3];b[t=6];C
3)
C:=a[t=2];b[t=5];C [] a[t=3];b[t=6];C
と
D:=a[t=2];stop ||| b[3<=t and t<=5];stop
4)
P:=p[t1p=t and t<2];
    (lc[t1p+d1<=t and t<=t1p+d4];r[t<=t1p+d4];P
     [] r[t<t1p+d1];
        (p[t<t1p+d2 and t2p=t];
             (r[t<t2p+d3];dc[t2p+d3<=t and t<=t1p+d4];P
              [] slc[t2p+d3<=t and t<=t1p+d4];
                    r[t<=t1p+d4];P)
         [] sc[t1p+d2<=t and t<=t1p+d4];exit)
    )
```

において d1,d2,d3,d4 に以下の値を代入したものどうし

```
(P1) d1=4, d2=5, d3=4, d4=12
(P2) d1=4, d2=5, d3=4, d4=10
(P3) d1=4, d2=5, d3=4, d4=7
(P4) d1=6, d2=5, d3=4, d4=10
```

	時間的強等価	非時間的強等価
1)	yes	yes
2)	yes	yes
3)	no	yes
4)		
P1 と P2	no	yes
P1 と P3	no	no
P1 と P4	no	no
P3 と P4	no	no

図 6: 表 3で用いたテストデータと期待される等価性判定結果

cess Algebra, *Journal of Formal Aspects of Computing Science*, Vol. 3,No. 2(1991), pp. 142–188.

[2] Bergstra, J. A. and Klop, J. W.: Algebra of communicating processes with abstraction, *Theor. Comput. Sci.*, Vol. 37(1985), pp. 77–121.

[3] Bolognesi, T., Lucidi, F., and Trigila, S.: From Timed Petri Nets to Timed LOTOS, *Protocol Specification, Testing and Verification, X*(Logrippo, L., Probert, R. L., and Ural, H.(eds.)), IFIP, Elsevier Science Publishers B.V.(North-Holland), 1990, pp. 395–408.

[4] Hoare, C. A. R.: *Communicating Sequential Processes*, Prentice Hall, 1985.

[5] ISO: *Information Processing System, Open Systems Interconnection, LOTOS – A Formal Description Technique Based on the Temporal Ordering of Observational Behaviour*, IS 8807, January 1989.

[6] Leduc, G. and Léonard, L.: A timed LOTOS supporting a dense time domain and including new timed operators, *Formal Description Techniques, V*(Diaz, M. and Groz, R.(eds.)), IFIP, Elsevier Science Publishers B.V.(North-Holland), 1993, pp. 87–102.

[7] Milner, R.: *A Calculus of Communicating Systems*, Lecture Notes in Computer Science, Vol. 92, Springer-Verlag, 1980.

[8] Moller, F. and Tofts, C.: A Temporal Calculus of Communicating Systems, *Proc. of CONCUR '90*(Baeten, J. C. M. and Klop, J. W.(eds.)), Lecture Notes in Computer Science, Vol. 458, Springer-Verlag, 1990, pp. 401–415.

[9] 中田明夫: 非隣接動作間の時間制約を指定可能にするための LOTOS の拡張と等価性の判定, 修士学位論文, 大阪大学大学院基礎工学研究科, 1994-02.

[10] Nakata, A., Higashino, T., and Taniguchi, K.: LOTOS enhancement to specify time constraints among non-adjacent actions using first order logic, *Proc. of 6th IFIP Int'l Conf. on Formal Description Techniques*, IFIP, October 1993.

[11] Park, D.: Concurrency and automata on infinite sequences, *Proc. of 5th GI Conference*(Deussen, P.(ed.)), Lecture Notes in Computer Science, Vol. 104, Springer-Verlag, 1981, pp. 167–183.

[12] Quemada, J., Azcorra, A., and Frutos, D.: TIC: A Timed Calculus for LOTOS, In Vuong [16], pp. 195–209.

[13] 佐藤一郎, 所真理雄: 時間的特性を考慮した並列プロセスの形式的記述, 情報処理学会論文誌, Vol. 34,No. 4(1993), pp. 540–547.

[14] Shiratori, N., Kaminaga, H., Takahashi, K., and Noguchi, S.: A Verification Method for LOTOS Specifications and its application, *Protocol Specification, Testing, and Verification, IX*(Brinksma, E., Scollo, G., and Vissers, C. A.(eds.)), IFIP, Elsevier Science Publishers B.V.(North-Holland), 1990, pp. 59–70.

[15] van Hulzen, W. H. P., Tilanus, P. A. J., and Zuidweg, H.: LOTOS Extended with Clocks, In Vuong [16], pp. 179–194.

[16] Vuong, S. T.(ed.): *Formal Description Techniques, II*, IFIP, Elsevier Science Publishers B.V.(North-Holland), 1990.

[17] Wang, Y.: CCS + Time = an Interleaving Model for Real Time Systems, *Proc. of ICALP '91*(Leach Albert, J., Monien, B., and Rodríguez Artalejo, M.(eds.)), Lecture Notes in Computer Science, Vol. 510, Springer-Verlag, 1991, pp. 217–228.