# 形式的設計検証のための分岐時間正則時相論理†

濱口清治†† 平石裕実††† 矢島脩三††

論理設計の多様化，複雑化にともない，設計が正しいことを示すための検証手法の確立が重要な課題となっている．対象を順序機械などの有限状態の機械ととらえたとき，設計が仕様を満足するかどうかを厳密に示すために，命題時相論理によって仕様を記述し，モデル検査アルゴリズムを適用する手法が提案されている．命題時相論理には種々の体系が存在するが，このうち Clarke らによって提唱された CTL (Computational Tree Logic) では，有限状態機械の動作を反映する Kripke 構造 $S$ の大きさ (Size ($S$)) と時相論理式 $f$ の長さ (Len ($f$)) の積に対し線形時間でモデル検査を行うことができ，順序機械，通信プロトコルなどの検証に応用されてきている．しかし，本論文で示すように，CTL は事象の繰り返しなどの性質を記述できないといった問題点があり，有限状態機械の仕様の記述に用いることを考えた場合，十分な表現力を持つとは言えない．そこで本論文では CTL と比べ真に高い表現能力を持つ分岐時間正則時相論理 (Branching time Regular Temporal Logic, BRTL) を提案する．またそのモデル検査アルゴリズムを示し，その時間計算量が CTL の場合と同じく，Size ($S$) と Len ($f$) の積に比例することを証明する．

## 1. はじめに

大規模集積回路技術の発達は，論理設計の多様化，複雑化をもたらし，その結果，設計段階における誤りの可能性が高くなってきている．設計された対象が設計者の課した仕様や条件を満足しているかどうかを確認するため，従来シミュレーションの手法が用いられてきた．しかし，シミュレーション手法では，入力されたパターンに対する動作以外については正しさが示されないという問題点があり，これに対して，時相論理，高階論理などによる対象の形式的記述および検証手法が研究されてきている[1)~4)]．

時相論理[5)]は従来の述語論理や命題論理の体系に，「常に」，「次の時点で」，「～が成り立つまで常に」などの概念を表す演算子 □, ○, $U$ を加えた論理体系である．検証の対象を有限状態機械ととらえた場合，特に命題時相論理の種々の体系では，検証に用いることのできるモデル検査問題が決定可能であることから，人手を介さない自動検証において有効であり，順序機械，通信プロトコル，非同期回路などの形式的検証に応用され，成果をあげている[1),2),6),7)]．

命題時相論理を用いた検証手法としては，モデル検査 (model checking)[1)] の手法が用いられてきている．この手法は，設計された有限状態機械の動作を反映する Kripke 構造に対して，仕様として与えられた命題時相論理の式の真偽を判定するというものである．

時相論理には，系列の各状態に対して原始命題への真偽の割当てを定めた線形時間モデルに基づく体系と，無限木の節点に対して原始命題への真偽の割当てを定めた分岐時間モデルに基づく体系がある．モデル検査問題の計算量を考えた場合，線形時間モデルに基づく古典的な命題時相論理 (Propositional Temporal Logic)，またその拡張である拡張時相論理 (Extended Temporal Logic)[8)] や正則時相論理 RTL (Regular Temporal Logic)[9)] では，多項式領域完全 (PSPACE-complete) またはそれ以上の計算量を必要とする[6),10),11)]．これに対して，分岐時間モデルに基づく計算木論理 (Computation Tree Logic, 以下 CTL)[1)] には Kripke 構造 $S$ の大きさ Size($S$) と時相論理式 $f$ の長さ Len($f$) の積に対し，線形時間のモデル検査アルゴリズムがある．

しかしながら，本論文で示すように，CTL では有限状態システムの性質のうち，繰り返しなどの性質は十分に記述することができない．

本論文では，CTL の表現力を改善した分岐時間正則時相論理 (Branching time Regular Temporal Logic, 以下 BRTL) を提案する．BRTL は命題時相論理の一種であり，CTL より真に高い表現能力を持ち，かつ CTL と同じく Size($S$) と Len($f$) の積に比例する時間計算量のモデル検査アルゴリズムを持つ．また，BRTL の拡張として，表現能力は BRTL と等価であるが記述量を少なくしうる体系として，

† Branching Time Regular Temporal Logic for Formal Design Verification by KIYOHARU HAMAGUCHI (Department of Information Science, Faculty of Engineering, Kyoto University), HIROMI HIRAISHI (Department of Information & Communication Sciences, Faculty of Engineering, Kyoto Sangyo University) and SHUZO YAJIMA (Department of Information Science, Faculty of Engineering, Kyoto University).

†† 京都大学工学部情報工学科
††† 京都産業大学工学部情報通信工学科

BRTL† を示す.

BRTL は時相演算子を記述する際, Büchi 型の $\omega$ 有限オートマトン[12]の部分クラスを用いており, BRTL† はオートマトン間にブール演算を許すよう拡張した時相論理である.

以下, 2章では, BRTL を用いた検証手法を概観し, 次に BRTL において, 時相演算子として利用する Büchi 型の $\omega$ 有限オートマトンの部分クラスを定義し, その性質を明らかにした後, BRTL を定義する. 3章では, BRTL が CTL よりも真に高い表現能力を持つことを証明する. 4章では, BRTL に対するモデル検査のアルゴリズムを示し, 時間計算量が Size($S$) と Len($f$) の積に比例することを証明する. 5章では, BRTL を拡張した BRTL† を示す.

## 2. 分岐時間正則時相論理 BRTL

### 2.1 検証の概観

この節では, 設計として順序機械が, 仕様として時相論理式が与えられたとき, 形式的検証がどのように行われるかを概観する (種々の用語の正確な定義は次節以降で示す).

なお, 以下では, 順序機械の入出力として, 0, 1 の2値のものを扱うことにする.

(1) 順序機械から Kripke 構造への変換

図1の(a)の Moore 型の順序機械を Kripke 構造に変換したものが(b)である. (a)中, 初期状態は太い矢印で指し示されている. Kripke 構造は有向グラフで各節点に原始命題の真偽値を割り当てたものであり, 図1では, $x$, $\neg x$ により各節点の $x$ に対する真偽の割当てが $T$ (真), $F$ (偽) であることを表している. ここでは, 順序機械の各入出力の値 0, 1 をそれぞれ $F$ (偽), $T$ (真) に対応させ, 各信号線名を原始命題と同一視している.

変換して得られる Kripke 構造は順序機械の動作を反映するものになり, 例えば, 図1(a)の破線で囲まれた2つの入出力の対の間の遷移が(b)の破線で囲まれた枝 $s_1$ と $s_2$ に対応する. 順序機械の初期状態に対応する Kripke 構造上の節点は太い矢印で示されている. (変換手続きの詳細は文献 6) を参照.)

順序機械を Kripke 構造に変換すると, 最悪の場合, 節点数がもとの順序機械の状態数の2乗のオーダになる. 順序機械のみを対象にした検証を考えた場合, 文献 13) の手法を適用すると順序機械を Kripke 構造に変換する必要は生じないが, CTL では表現能力, 計算量が Kripke 構造に対して論じられており, 本論文では CTL と比較するため, 一貫して Kripke 構造を用いて議論を進める.

(2) 仕様記述

仕様として, 順序機械の入出力の値の系列, すなわち(1)で得られた Kripke 構造が満足すべき条件を記述する. 例として, 次の仕様を分岐時間正則時相論理で記述する.

「入力 $x$ が1になれば, 次の時刻で出力 $z$ の値は反転するという性質が常に成立する」

$$\forall\ \mathrm{Always}(x \Rightarrow ((z \Rightarrow \forall\ \mathrm{Next}(\neg z)) \wedge (\neg z \Rightarrow \forall\ \mathrm{Next}(z))))$$

ここで, $\forall$ は「(Kripke 構造上の) すべての経路上で」を意味する経路限定子である. また, Always($f$), Next($f$) はそれぞれ「常に $f$ が成立する」,「次の時刻で $f$ が成立する」を意味する. 分岐時間正則時相論理では, Always と Next はそれぞれ図2と図3に示される有限オートマトンによって記述される. このオートマトンは枝の遷移条件として命題論理式を持ち, Kripke 構造上の無限経路を入力とする有限オートマトンである. (遷移先としては, Kripke 構造の節点に割り当てられた原始命題の真偽値のもとで, 真になる論理式をラベルされた枝が選ばれる. また無限系

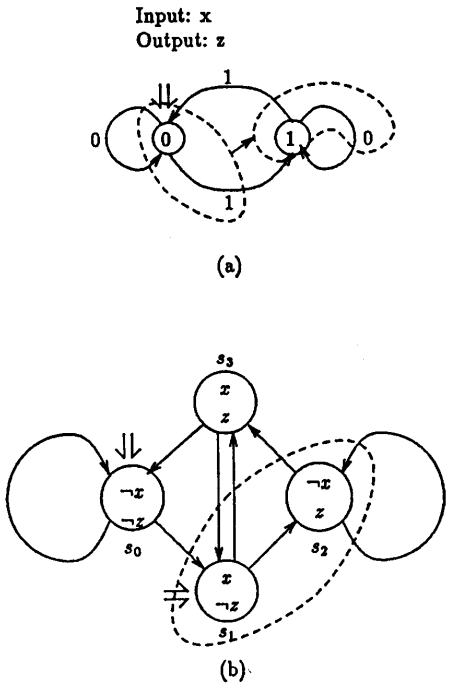


図1 順序機械から Kripke 構造への変換
Fig. 1 Transformation of a sequential machine to a Kripke structure.

列の受理／非受理は，少なくとも1つの受理状態を無限回通過するかどうかによって決定される.)

(3) 検　　証

本論文での検証問題は，Kripke 構造の初期節点において，仕様として与えられた時相論理式の規定している性質が満足されているかどうかを調べることである．この問題は Kripke 構造の初期節点に対する論理式の真偽を判定する問題（モデル検査問題）として 2.3 節で定式化される.

分岐時間正則時相論理の論理式の真偽は，Kripke 構造上の各節点に対して定まる．有限オートマトンを用いた論理式の真偽は，∀(∃) が先行する場合は，各節点からはじまるすべての（ある）無限経路が，そのオートマトンによって受理されるかどうかによって定義される.

例えば，図1(b)においては，∀Next(z) は節点 $s_2$ で真になる．これは $s_2$ から始まるすべての経路上の次の節点 $s_2$, $s_3$ で $z$ が真になり，図3の機械により受理されるからである．同様に，$g \stackrel{\text{def}}{=} x \Rightarrow ((z \Rightarrow \forall \text{Next}(\neg z)) \wedge (\neg z \Rightarrow \forall \text{Next}(z)))$ は，節点 $s_0$, $s_1$, $s_2$, $s_3$ で真になる．次に $g$ に着目すると，Kripke 構造の任意の初期節点から始まるすべての経路に対して「常に $g$」が成立する，すなわち，すべての経路が（各部分式の各節点での真偽を原始命題の真偽と同様にみなしたとき）図2の有限オートマトンによって受理される.

これにより，設計が仕様を満足することが示されたことになる.

### 2.2 論理型決定性 ω 有限オートマトン

本節では，分岐時間モデルに基づく時相論理 BRTL (Branching time Regular Temporal Logic) を定義する.

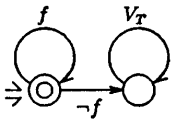


図 2 Always($f$) を表現するオートマトン演算子
Fig. 2 Automaton connective representing Always($f$).

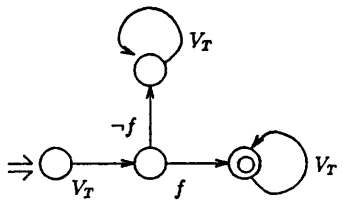


図 3 Next($f$) を表現するオートマトン演算子
Fig. 3 Automaton connective representing Next($f$).

BRTL は時間概念を表現するための時相演算子の記述に，次に定義する ω 有限オートマトンを利用している．直観的には，このオートマトンは決定性完全指定の Büchi 型 ω 有限オートマトンに変更を加えたもので，遷移の枝に記号の代わりに命題論理式をラベル付けしたものになっている．受理／非受理は，各命題変数への真偽値の割当ての無限の列に対して決定される．(BRTL で用いられるオートマトンはさらに，以下に示す条件1を満足するものに制限されている.)

時相演算子を定義する際，他のクラスの有限オートマトンを用いることも考えられるが，これについては計算量の議論を行ったのち，4.2 節で考察することとする.

以下では，命題論理における真と偽をそれぞれ $T$, $F$ で表現する．命題論理式 $f$ が命題変数の集合 $P=\{p_1, \cdots, p_n\}$ から構成される時，各命題変数への真偽値の割当て $v \in \{F, T\}^*$ に対する $f$ の値を $f(v)$ と記す.

また，集合 $X$ に対して，その要素の個数を $|X|$ と記述する．$A \Leftrightarrow B$ は $A$ が $B$ であるための必要十分条件であることを表す．集合 $X$ のある（すべての）要素 $x$ に関して $C$ という条件が成り立つことを $\exists x \in X.\, C$($\forall x \in X.\, C$) と記述する．$X^\omega$ および $X^+$ によって $X$ の要素の無限系列，長さ1以上の有限系列の集合をそれぞれ表す．有限系列 $\alpha \in X^+$ に対して $\beta, \gamma \in X^+$ が存在し，$\alpha$ が $\beta$ と $\gamma$ をこの順序で連接した系列であるとき，$\beta$ を $\alpha$ のプレフィックスという.

**定義 1** 論理型決定性 ω 有限オートマトン (logic-type deterministic ω finite automaton, 以下，ldo-fa と記す) $A=(Q, P, Br, q_0, F)$ を次のように定義する.

$Q$ は有限個の状態の集合，$P=\{p_1, \cdots, p_n\}$ は命題変数の集合.

$Br : Q \times Q \to BF$ ($BF$ は $P$ から構成される命題論理式の集合）は状態の2つ組に命題論理式を割り当てる部分関数（したがって，命題論理式が割り当てられない状態の2つ組も存在しうる）である．ただし，$Br$ は次の条件を満たす.

$q \in Q$ に対して，$Br(q, Q)=\{f \mid \exists q'.\, Br(q, q')=f\}$ を考えた時，「任意の $f_1 \neq f_2$ なる $f_1, f_2 \in Br(q, Q)$ に対して，$f_1 \wedge f_2$ は恒偽」かつ「$\vee_{f \in Br(q,Q)} f$ は恒真」.

$\text{Edges}(A) \stackrel{\text{def}}{=} \{(q, q') \mid \exists f.\, Br(q, q')=f\}$ と定義しておく.

$q_0$ は初期状態であり，$F$ は受理状態の集合である．$Q-F$ の要素を非受理状態と呼ぶ．

$A$ は $\Sigma=2^P$ の要素の無限系列に対して，受理／非受理を決定する．遷移関数 $\delta: Q\times\Sigma\to Q$ は $q, q'\in Q, v\in\Sigma$ に対して，$\delta(q,v)=q'\Leftrightarrow Br(q,q')(v)=T$ となるように定められる．無限系列 $\phi\in\Sigma^\omega$ を $A$ に入力した時，無限回通過する状態の集合を $Inf(\phi)$ と表す．$A$ が受理する言語を $\{\phi|Inf(\phi)\cap F$ が空でない$\}$ と定義し，$\langle A\rangle$ と記すこととする． □

図 4 に ldo-fa の例を示す．太い矢印が初期状態を指し示しており，◎は受理状態を表現している．$p$, $q$ は命題変数である．

BRTL 式では，上述の $\omega$ 有限オートマトンを論理式の一部として用いるため，オートマトンに対して否定演算を定義できなければ不自然である．しかし，決定性の Büchi 型 $\omega$ オートマトンは，否定に対して閉じていない[14]ため，ldo-fa $A$ に対して，$\Sigma^\omega-\langle A\rangle$ を受理する ldo-fa を構成することは，一般にできない．そこで，ldo-fa に以下のような制限を加えたオートマトンを用いることとする．

**定義 2** ldo-fa $A=(Q,P,Br,\delta,q_0,F)$ が次の条件 1 を満足するとき，ldo-fa-1 と呼ぶ． □

条件 1 は，どの非受理状態 $q_r$ からも，受理状態を経由して再び $q_r$ へ遷移させる入力系列が存在しないことを示している．

**条件 1** $A$ のどの $q_r\in(Q-F)$ および任意の有限系列 $\alpha\in\Sigma^+$ と $\alpha$ の任意のプレフィックス $\beta\in\Sigma^+$ に対して，$\delta(q_r,\beta)=q_a$ かつ $\delta(q_r,\alpha)=q_r$ となるような $q_a\in F$ は存在しない． □

このとき，次の補題が成立する．

**補題 1** ldo-fa-1 $A=(Q,P,Br,\delta,q_0,F)$ に対して，$\Sigma^\omega-\langle A\rangle$ を受理する ldo-fa-1 $A$ は，$A$ の受理状態と非受理状態を交換することによって得られる．

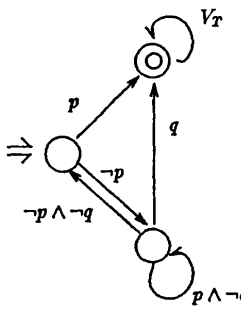


図 4 ldo-fa の例
Fig. 4 An example of an ldo-fa.

(証明) 条件 1 が成立すれば，どの受理状態 $q_a$ からも，非受理状態を経由して再び $q_a$ へ遷移させる入力系列が存在しないことに注意する．

$\phi\in\langle A\rangle$ ならば，$Inf(\phi)\subseteq F$ である．なぜなら，もしも，非受理状態 $q_r$ が $Inf(\phi)$ に含まれるならば，$\phi=\phi_1\phi_2\cdots\phi_i\cdots(\phi_i\in\Sigma^+, i\geq 1)$ なる無限個の有限系列 $\phi_i$ が存在して，$\delta(q_0,\phi_1)=q_r$ かつ $\delta(q_r,\phi_j)=q_r$ $(j\geq 2)$ となる．条件 1 より，$q_r$ から $q_r$ へ $\phi_j$ によって遷移する際，受理状態を経由することはない．したがって，$Inf(\phi)$ は受理状態を含まないことになって，$\phi\in\langle A\rangle$ に反する．ゆえに，$Inf(\phi)\cap(Q-F)$ は空．したがって，$\phi\notin\langle\bar{A}\rangle$.

同様に $\phi\notin\langle A\rangle$ ならば，$\phi\in\langle\bar{A}\rangle$ も証明することができる．また，明らかに $\bar{A}$ は条件 1 を満足している． □

**補題 2** 2 つの ldo-fa-1 $A_i=(Q_i,P,Br_i,\delta_i,q_{i0},F_i)$ $(i=1,2)$ に対して，$\langle A_1\rangle\cup\langle A_2\rangle$ を受理する ldo-fa-1 $A_1|A_2=(Q,P,Br,\delta,q_0,F)$ は次のように構成することができる．

- $Q=Q_1\times Q_2=\{(q_1,q_2)|q_1\in Q_1, q_2\in Q_2\}$
- $Br: Q\times Q\to BF$ は次の条件を満たす．$q_i, q'_i\in Q_i$ $(i=1,2)$ とする．
  $Br((q_1,q_2),(q'_1,q'_2))$ が定義されるのは，$Br_i(q_i,q'_i)$ が，$i=1,2$ のどちらの場合にも定義されている時かつその時に限り，$Br((q_1,q_2),(q'_1,q'_2))=Br_1(q_1,q'_1)\wedge Br_1(q_2,q'_2)$.
- $q_0=(q_{10},q_{20})$.
- $F=\{(q_1,q_2)|q_1\in F_1$ または $q_2\in F_2\}$. □

したがって，次の定理が導かれる．

**定理 1** ldo-fa-1 が受理する言語のクラスは，集合上のブール演算に関して閉じている． □

ldo-fa-1 以外のオートマトンを用いた場合については，4.2 節で考察する．

### 2.3 BRTL の構文と意味

BRTL の論理式の真偽は Kripke 構造に対して定まる．

$AP=\{p_1,p_2,\cdots,p_n\}$ を原始命題の集合とする．

**定義 3** $S=\langle\Sigma,I,R,\Sigma_0\rangle$ を Kripke 構造 (Kripke structure) と呼ぶ．ここで，$\Sigma$ は節点の集合，$I:\Sigma\to 2^{AP}$ は各節点に対し，原始命題の真偽を割り当てる関数，$R\subseteq\Sigma\times\Sigma$ は $\Sigma$ 上の有向枝の集合（任意の節点から少なくとも 1 本の枝が出ているとする）であり，$\Sigma_0$ は初期節点の集合である．

Kripke 構造の大きさ Size($S$) を $|\Sigma|+|R|$ と定義

する. □

BRTL の構文は BRTL 式と時相演算子を表現するためのオートマトン演算子 (automaton connective)[8] からなる.

**定義 4 構文**

BRTL 式: 以下では, その集合を $BRF$ と記す.

$p\in AP, f, g\in BRF$ また, $A$ がオートマトン演算子ならば, $p, (\neg f), (f\vee g), (\exists A)\in BRF$ である.

オートマトン演算子

$A$ を $P$ を命題変数の集合とする ldo-fa-1 とする. $\{f_1, f_2, \cdots, f_n\}\subseteq BRF$ とする. このとき, $A(f_1, f_2, \cdots, f_n)$ は, $A$ 中の各命題変数 $p_1, p_2, \cdots p_n\in AP$ を同時にそれぞれ $f_1, f_2, \cdots, f_n$ で置き換えたものを表すとするこの時, $Br(q, q')$ も代入後の値をとるものとする.

このとき, $A(f_1, f_2, \cdots, f_n)$ および $\neg A(f_1, f_2, \cdots, f_n)$ はオートマトン演算子である.

$A(f_1, f_2, \cdots, f_n)$ に対しても, ldo-fa-1 に対する否定演算を拡張する. $\overline{A(f_1, f_2, \cdots, f_n)}$ によって, $A$ の受理状態と非受理状態を交換したものを表すとする.

BRTL の意味は, Kripke 構造 $S=\langle\Sigma, I, R, \Sigma_0\rangle$ に対して定義する. $S, s\vDash f$ は BRTL 式 $f$ が節点 $s\in\Sigma$ に対して真であることを意味する. $p\in AP$, $f, g$, $f_1, f_2, \cdots, f_n$ は BRTL 式, $A$ は ldo-fa-1 とする.

**定義 5** BRTL の意味

- $S, s\vDash p\Leftrightarrow p\in I(s)$
- $S, s\vDash(f\vee g)\Leftrightarrow S, s\vDash f$ または $S, s\vDash g$
- $S, s\vDash(\neg f)\Leftrightarrow S, s\nvDash f$
- $S, s\vDash(\exists A(f_1, f_2, \cdots, f_n))\Leftrightarrow$Kripke 構造 $S$ 上の節点 $s$ から始まる無限系列 $\sigma=s_0s_1s_2\cdots$ と $A$ の状態の無限系列 $q_0q_1q_2\cdots$ があって次の条件を満足している. $S, s_i\vDash Br(q_i, q_{i+1})$ (ただし, $i=0, 1, 2, \cdots$ とする) かつ $q_0, q_1, \cdots$ のうち, 無限回出現する状態がすくなくとも 1 つ $A$ の受理集合に含まれる.
- $S, s\vDash(\exists\neg A(f_1, f_2, \cdots, f_n))\Leftrightarrow$ $S, s\vDash(\exists\overline{A(f_1, f_2, \cdots, f_n)})$

すべての $s\in\Sigma_0$ に対し $S, s\vDash f$ の時, $S\vDash f$ と書く. □

ブール演算子 $\wedge$, $\equiv$ および $\Rightarrow$ をそれぞれ論理積, 同値, 包含の意味で用いるものとし, $V_T$ は恒真式を表すとする. また $\forall A\overset{\text{def}}{=}\neg\exists\neg A$ および $\forall\neg A\overset{\text{def}}{=}\neg\exists A$ とする. $\forall$, $\exists$ は, それぞれ Kripke 構造上のすべての経路およびある経路を意味する経路限定子である.

単項演算子は二項演算子よりも高い優先順位を持つものとし, 混乱の恐れのない時は '(', ')' を省略することもある.

次に, BRTL 式の長さを定義する.

**定義 6** BRTL 式 $f$ およびオートマトン演算子 $A(f_1, f_2, \cdots, f_n)$ に対して, その部分式の集合 SF($f$), 長さ Len($f$) は次のように再帰的に定義される.

以下で, $f, f_1, f_2, \cdots, f_n, g, g_1, g_2$ は BRTL 式とする.

- $f$ が原始命題の場合. SF($f$)=$\{f\}$. Len($f$)=1.
- $f=\neg g$ の場合. それぞれ SF($f$)=SF($g$)$\cup\{\neg g\}$, Len($f$)=Len($g$)+1.
- $f=g_1\vee g_2$ の場合. それぞれ SF($f$)=SF($g_1$)$\cup$ SF($g_2$)$\cup\{g_1\vee g_2\}$, Len($f$)=Len($g_1$)+Len($g_2$)+1.
- $f=\exists A(f_1, f_2, \cdots, f_n)$ の場合. SF($f$)=$\{\exists A(f_1, f_2, \cdots, f_n)\}\cup\cup_{1\le i\le n}$SF($f_i$) および Len($f$)=Size($A(f_1, f_2, \cdots, f_n)$). ここで, Size($A(f_1, f_2, \cdots, f_n)$) $\overset{\text{def}}{=}|Q|+|\text{Edges}(A)|+\sum_{g\in Br'(Q,Q)}\text{Len}(g)+\sum_{1\le i\le n}$ Len($f_i$) である. ただし, $Br'(Q\times Q)$ は $f_1, f_2, \cdots, f_n$ を代入する前の ldo-fa-1 $A$ にラベル付けされている命題論理式の集合を表す.
  すなわち, Size($A(f_1, f_2, \cdots, f_n)$) は, $A$ の状態数と枝の数, 代入前の $A$ の枝にラベルされている命題論理式の長さの総和と代入される BRTL 式の長さの総和の合計となっている.
- $f$ が $\exists\neg A(f_1, f_2, \cdots, f_n)$ の場合. SF($f$)=$\{\exists\neg A(f_1, f_2, \cdots, f_n)\}\cup\cup_{1\le i\le n}$SF($f_i$) および Len($f$) =Len($A(f_1, f_2, \cdots, f_n)$)+1. □

## 3. BRTL の表現能力

本章では, BRTL と CTL の表現能力を比較する. まず, CTL の定義の概略を示す.

**定義 7** CTL の論理式の真偽値は, Kripke 構造の各節点に対して定まる.

BRTL では $\exists A(f_1, f_2, \cdots, f_n)$, $\exists\neg A(f_1, f_2, \cdots, f_n)$ の形式で時間の概念を表現しているが, CTL では, $\forall Xf$, $\exists Xf$, $\forall[f_1Uf_2]$ および $\exists[f_1Uf_2]$ のみが用いられる. これらの意味は次のとおりである.

- $S, s\vDash\forall Xf(\exists Xf)\Leftrightarrow$Kripke 構造 $S$ 上の $s=s_0$ なるすべての (ある) 無限系列 $s_0s_1s_2\cdots$ に対して, $S, s_1\vDash f$
- $S, s\vDash\forall[f_1Uf_2](\exists[f_1Uf_2])\Leftrightarrow$Kripke 構造 $S$ 上の $s=s_0$ なるすべての (ある) 無限系列 $s_0s_1s_2\cdots$

に対して，$S, s_0 \models f_2$ または，ある $n \geq 0$ が存在して $S, s_i \models f_1$ $(i=0, 1, 2, \cdots, n)$ かつ $S, s_{n+1} \models f_2$

□

**定理 2** 任意の CTL 式 $f$，任意の Kripke 構造 $S=\langle \Sigma, I, R, \Sigma_0 \rangle$ および $s \in \Sigma$ に対して，ある BRTL 式 $f'$ が存在して，

$S, s \models f \Leftrightarrow S, s \models f'$

(略証) 図3の Next($f$) および**図5**の Until($f_1, f_2$) をオートマトン演算子として用いると次に示すように CTL 式と等価な BRTL 式が構成できる．

- $\exists Xf = \exists \mathrm{Next}(f)$
- $\forall Xf = \neg \exists \neg \mathrm{Next}(f)$
- $\exists [f_1 U f_2] = \exists \mathrm{Until}(f_1, f_2)$
- $\forall [f_1 U f_2] = \neg \exists \neg \mathrm{Until}(f_1, f_2)$ □

定理2は CTL で表現できることは BRTL においてもまた表現できることを示している．次に，この逆は成立しないこと，すなわち，BRTL では表現可能であるが，CTL では表現できない性質が存在することを示す．

以下の証明は，文献 8) の定理 4.1 および系 4.2 を Kripke 構造に拡張したものになっている．

次の $Re_i(S, s_0)$ は $s_0$ から Kripke 構造 $S$ 上の枝を $i$ 回遷移して到達することのできる節点すべての集合を表す．

**定義 8** Kripke 構造 $S=\langle \Sigma, I, R, \Sigma_0 \rangle$，節点 $s_0 \in \Sigma$ および $i$ $(i=0, 1, 2, \cdots)$ に対して，

$Re_i(S, s_0) = \{s_i \mid \exists s_0, s_1, \cdots s_i. (s_k, s_{k+1}) \in R \ (k=0, 1, \cdots, i-1)\}$ □

**補題 3** $p$ を原始命題とする．$C_i$ を次のように定める．

$C_i \overset{\mathrm{def}}{=} \{(S, s) \mid$ Kripke 構造 $S=\langle \Sigma, I, R, \Sigma_0 \rangle$ と節点 $s \in \Sigma$ は次の条件を満足する．$\}$

1. $\forall s' \in Re_j(S, s). S, s' \models p$ $(j=0, 1, \cdots, i)$,

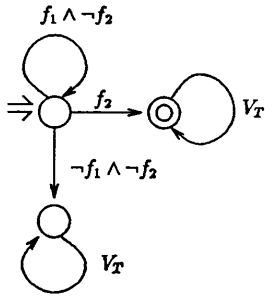


**図 5** Until($f_1, f_2$) を表現するオートマトン演算子
Fig. 5 Automaton connective representing Until($f_1, f_2$).

2. $\forall s'' \in Re_{i+1}(S, s). S, s'' \models \neg p$,
3. $\forall s''' \in Re_j(S, s). S, s''' \models p$ $(j=i+1, i+2, \cdots)$

CTL 式 $f$ がせいぜい $n$ 個の '$X$' しか含まないと仮定すると，$i, i' > n$ であるような任意の $C_i$, $C_{i'}$ から選んだ任意の $(S_i, s_i) \in C_i, (S_{i'}, s_{i'}) \in C_{i'}$ に対して，

$S_i, s_i \models f \Leftrightarrow S_{i'}, s_{i'} \models f$ (*)

すなわち，任意の $i > n$ に対して，$f$ の真偽値は変化しない．

(証明) 部分式と $i$ の値に関する帰納法によって証明する．

1. $f$ が原始命題の場合は明らか．
2. $f$ が $\neg g$ または $g_1 \vee g_2$ の場合は，$g, g_1, g_2$ に対して，帰納法の仮定より (*) が成立することから，$g$ に対しても成立する．
3. $f$ が $\forall Xg$ の場合．$(S, s) \in C_i$ に対して $S, s \models \forall Xg \Leftrightarrow$ すべての $(s, s') \in R$ であるような $(S, s')$ に対して $S, s' \models \forall g$ となる．
   $g$ に含まれる '$X$' の個数はせいぜい $n-1$ 個，$(S, s') \in C_{i-1}$，$i-1 > n-1$ であることより，帰納法の仮定から，$S, s' \models \forall g$ か否かは，$i$ に依存せずに決まる．ここで考えている $(S, s')$ では，$S, s' \models \forall g$ か $S, s' \not\models \forall g$ かが，どの $s'$ を選んでも同じであることより適当な $(s, s') \in R$ なる $s'$ に対して $S, s \models \forall Xg \Leftrightarrow S, s' \models \forall g$ ゆえに，$S, s \models \forall Xg$ か否かは $i$ に依存しない．
4. $f$ が $\forall [g_1 U g_2]$ の場合．$(S, s_i) \in C_i$ とする．
   $S, s_i \models \forall [g_1 U g_2] \Leftrightarrow$
   $s_i$ から始まるすべての無限系列 $s_i s_{i-1} s_{i-2} \cdots s_{n+1} \sigma_n$ $(s_j \in \Sigma$ (ただし，$j=i, i-1, \cdots)$ かつ $\sigma_n$ は $(s_{n+1}, s_n) \in R$ なる節点 $s_n$ から始まる $S$ 上の節点の無限系列) に対して，
   $(S, s_i \models g_2)$ または $((S, s_i \models g_1)$ かつ
   $((S, s_{i-1} \models g_2)$ または $((S, s_{i-1} \models g_1)$ かつ
   $\cdots$
   $((S, s_{n+1} \models g_2)$ または $((S, s_{n+1} \models g_1)$ かつ
   $(S, s_n \models \forall [g_1 U g_2])) \cdots)))$
   $g_1, g_2$ に対する帰納法の仮定より，$k=1, 2$ に対し，
   $S, s_j \models g_k$ $(j=n+1, n+2, \cdots, i-1, i) \Leftrightarrow$
   $S, s_n \models g_k$ となる．
   ゆえに，
   $S, s_i \models \forall [g_1 U g_2]$ か否かは $i$ に依存せずに定まる．
5. $f$ が $\exists h$ $(h=[g_1 U g_2]$ または $Xg)$ の場合．

$(S,s)\in C_i$ に対しては $C_i$ の性質より，$S,s\models \exists h \Leftrightarrow S,s\models \forall h$ が成立することから，$\forall h$ の場合に帰着される． □

次の定理は，CTL では与えられた $m\geq 2$ に対して，Kripke 構造上のすべての（またはある）経路 $s_0s_1\cdots$ において，$s_i$ $(i=km, k=0,1,\cdots)$ に対し $p$ が真になるという性質は記述できないことを示している．

**定理 3** $m\geq 2$ が与えられたとする．Kripke 構造と節点の組 $(S,s)$ の集合 $D=\{(S,s)\mid \forall s'\in Re_{km}(S,s).\ S,s'\models p, (k=0,1,\cdots)\}$ を考える．

$(S,s)\in D \Leftrightarrow S,s\models f$

を満足する CTL 式 $f$ は存在しない．

（証明） 上記の条件を満足するような $f$ が存在するとする．$f$ が $l$ 個の 'X' を含んでいるとする．

補題より，$km-1>l$ であるような任意の $(S,s)\in C_{km}$, $(S',s')\in C_{km-1}$ に対して，$S,s\models f \Leftrightarrow S',s'\models f$ となるが $(S,s)\in D$ かつ $(S',s')\notin D$ であることから，仮定に反する． □

オートマトン演算子を用いれば，条件

$S,s\in D \Leftrightarrow S,s\models f$

を満足する BRTL 式 $f=\forall A_R$ は容易に構成することができる．$A_R$ を**図 6** に示す．

## 4. BRTL のモデル検査アルゴリズム

### 4.1 モデル検査アルゴリズム

**定義 9** Kripke 構造 $S=\langle \Sigma, I, R, \Sigma_0\rangle$ および BRTL 式 $f$ が与えられた時，$S\models f$ であるかどうかを判定する問題を，BRTL のモデル検査問題 (model checking problem) と呼ぶ． □

2.1 節で示した手法により，上記のモデル検査問題を用いて有限状態機械が仕様を満足していることを示すことができる．

モデル検査アルゴリズムを示す前に，正当性の証明に必要な定義および補題を示す．

以下では，特にことわりがない限り，Kripke 構造 $S=\langle \Sigma, I, R, \Sigma_0\rangle$ とオートマトン演算子 $A(f_1, f_2, \cdots, f_n)$（ただし，$A$ は $A=(Q,P,Br,q_0,F)$ なる ldo-fa-1 である）を固定して用いる．

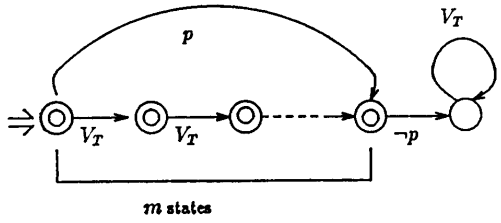


**図 6** 繰り返しを記述するためのオートマトン演算子
Fig. 6 Automaton connective representing repetition.

BRTL の意味の定義と ldo-fa-1 の性質より次の補題が導かれる．

**補題 4** $S,s\models \exists A(f_1,f_2,\cdots,f_n)\Leftrightarrow$

Kripke 構造 $S$ 上の節点とオートマトン演算子 $A$ の状態の組の有限系列 $(s_0,q_0)(s_1,q_1)\cdots(s_k,q_k)(s_{k+1},q_{k+1})\cdots(s_n,q_n)$（ここで，$s_0=s$）が存在して，条件 2 および条件 3 を満足する．

**条件 2**

$(s_i,s_{i+1})\in R (i=0,1,\cdots,n-1)$

かつ $S,s_i\models Br(q_i,q_{i+1})$ $(i=0,1,\cdots,n-1)$

**条件 3**

$s_k=s_n$ かつ $q_k=q_n$ かつ $q_j\in F$ $(j=k,k+1,\cdots,n)$ □

つぎに BRTL のモデル検査アルゴリズムを示す．

**アルゴリズム 1** モデル検査アルゴリズム

- 入力：Kripke 構造 $S$ および BRTL 式 $f$
- 出力：$S\models f$ ならば，yes. そうでなければ，no.
- 方法：

1. $f$ 中のオートマトン演算子を 2 章で示した方法により，'¬' を含まない形に変換する．すなわち，オートマトン演算子を含む部分式は $\exists A(f_1,f_2,\cdots,f_n)$ の形式のもののみになるよう変換する．
2. 以下の（a）-（c）の場合にしたがい，$f$ の部分式 $g_i\in SF(f)$ に関して原始命題からボトムアップに，次の条件を満足する $S$ の節点 $s$ に $g_i$ をラベル付けする．

   $S,s\models g_i$

   $f$ まで終了した時点で，すべての $s_0\in\Sigma_0$ に対して，$f$ がラベルされていれば yes を，そうでなければ no を返す．

   （a） $g_i$ が原始命題の場合，$g_i\in I(s)$ なる節点 $s$ に $g_i$ をラベル付けする．

   （b） $g_i$ が $h_1\vee h_2$ または $\neg h$ ($h_1,h_2,h$ は BRTL 式）の場合，すでに $S$ 上にラベル付けされている $h_1,h_2,h$ を利用して，各節点での $g_i$ の真偽を決定する．

   （c） $g_i=\exists A(f_1,f_2,\cdots,f_n)$ の場合．

   i. グラフ $G=(V,E)$ を構成する．ここで，

   $V=\{(s,q)\mid s\in\Sigma, q\in Q\}$

   $E=\{((s,q),(s',q'))\mid (s,s')\in R$

かつ $S, s \models Br(q, q')$}

ii. $G$ の節点集合を $V' = \{(s, q_F) | q_F \in F\}$ に制限した部分グラフ $G'$ の強連結成分の節点集合を求め $V_C$ とする.

iii. $G$ 上で $V_C$ のいずれかの節点に到達可能な節点の集合 $V_R$ を求める.

iv. $(s, q_0) \in V_R$ ($q_0$ は $A$ の初期状態) なる $s \in \Sigma$ に $\exists A(f_1, f_2, \cdots, f_n)$ をラベル付けする.

次にこのアルゴリズムの正当性を示す.

**定理 4** アルゴリズム 1 は $S \models f$ か否かを判定する.

(証明) 部分式に関する帰納法によって証明する.

2. (a) の場合は明らか. 2. (b) の場合, すでに $h_1$, $h_2$, $h$ の $S$ 上の各節点における真偽が確定していることより, $g_i$ に対する真偽値の判定も正しく行うことができる.

2. (c) の場合, $(s, q_0) \in V_R$ が存在したとする. ii. と iii. より, 系列

$(s, q_0)(s_1, q_1)\cdots(s_k, q_k)(s_{k+1}, q_{k+1})\cdots(s_n, q_n)$

が存在して, $s_k = s_k$ かつ $s_n = q_n$ かつ $(s_k, q_k), \cdots, (s_n, q_n) \in V_C$ となる.

$V_C$ の定義より, $q_k, q_{k+1}, \cdots, q_n \in F$. したがって, 補題 4 より, $S, s \models \exists A(f_1, f_2, \cdots, f_n)$.

逆に, $(s, q_0) \notin V_R$ かつ $S, s \models \exists A(f_1, f_2, \cdots, f_n)$ と仮定する.

補題 4 より, 条件 2 を満足する系列

$(s, q_0)(s_1, q_1)\cdots(s_k, q_k)(s_{k+1}, q_{k+1})\cdots(s_n, q_n)$

が存在する. 少なくとも, $(s_k, q_k)(s_{k+1}, q_{k+1})\cdots(s_n, q_n) \in V_C$ であるから, iii. より $(s, q_0) \in V_R$ となり仮定に反する.

$(s, q_0) \notin V_R$ ならば $S, s \not\models \exists A(f_1, f_2, \cdots, f_n)$ が証明された. □

**定理 5** BRTL に対するアルゴリズム 1 によるモデル検査の時間計算量は, $O(Size(S) \times Len(f))$ である.

(証明)

1. ではオートマトン演算子には, '¬' のみしか許されていないため, $f$ 中に出現するオートマトン演算子の状態数の総和に比例する時間すなわち $O(Len(f))$ で, '¬' を含まない BRTL 式に変形することができる.

2. (a) の場合は明らか. 2. (b) では $\xi = h_1, h_2$ または $h$ に対して $O(Size(S) \times Len(\xi))$ の時間で, $S$ へのラベル付けが可能とすると, $h_1 \vee h_2$ または $\neg h$ に対してはどちらも $O(|\Sigma|)$ の時間でラベル付けができる. したがって, $g_i = h_1 \vee h_2$ または $\neg h$ に対し $O(Size(S) \times Len(g_i))$ の時間を要する.

2. (c) では, i. において, $S, s \models Br(q, q')$ を判定するため, $Br(Q \times Q)$ の要素すべて, すなわち $A$ の枝にラベルされている BRTL 式すべてに対して, $S$ へのラベル付けを行う. この時, まず, $f_1, \cdots, f_n$ に対するラベル付けを行い, 次に, $Br(Q \times Q)$ の要素についてラベル付けを行うと, 時間計算量が $O(Size(S) \times (\sum_{g \in Br'(Q \times Q)} Len(g) + \sum_{1 \le i \le n} Len(f_i)))$ となる.

i. の $G$ の $|V|$ は $O(|\Sigma| \times |Q|)$, $|E|$ は $O(|R| \times |Edges(A)|)$ の大きさであることから, $Br(Q \times Q)$ の要素のラベル付けが終了していれば, $G$ は $O(Size(S) \times (|Q| + |Edges(A)|))$ の時間で構成でき, $S, s \models Br(q, q')$ の判定と合わせて, $O(Size(S) \times Len(\exists A(f_1, f_2, \cdots, f_n)))$ の時間で構成できる.

ii. では文献 15) で示されている手法により, $O(\min(|V|, |E|))$ すなわち $O(Size(S) \times (|Q| + |Edges(A)|))$ の時間で $V_C$ を求めることができる.

iii. および iv. もまた $O(Size(S) \times (|Q| + Edges(A)|))$ の時間で $V_R$ を求めることができる.

したがって, 2. (c) は $O(Size(S) \times (Len(\exists A(f_1, f_2, \cdots, f_n))))$, すなわち, $O(Size(S) \times Len(g_i))$ の時間で処理することができる. □

### 4.2 他のオートマトンを用いた場合との比較

BRTL で時相演算子を定義する際に用いた ldo-fa-1 は, Büchi 型決定性 $\omega$ 有限オートマトンの部分クラスを用いているが, 別のクラスの有限オートマトンを用いることについて, 以下で考察する.

有限系列を受理する通常の有限オートマトンを用いれば, 本稿と同様の形式で時相論理を構築することができる. しかし, 有限系列を用いて条件を記述した場合, 記述が繁雑になる場合があり[16], ここでは, $\omega$ 有限オートマトンを採用している.

一般の Büchi 型決定性 $\omega$ 有限オートマトンが受理する言語のクラスは集合上のブール演算に関して閉じていないことから[14], これを用いると論理体系に取り込むことは難しくなる. また, Büchi 型非決定性 $\omega$ 有限オートマトンのクラスは集合上のブール演算に関して閉じているが, その補集合を受理する $\omega$ 有限オートマトンはもとの状態数の指数オーダの状態数を持つため, モデル検査問題のアルゴリズムとして計算量の小さいものを構築することができなくなる. また, 決定性 $\omega$ 有限オートマトンのクラスで Büchi 型非決定 $\omega$ 有限オートマトンのクラスに等価なものとして, Muller 型 $\omega$ 有限オートマトン[17]が知られているが,

受理条件を状態集合のべき集合を用いて記述しなければならないことから，記述量が状態数の指数となる可能性があり実用上問題がある．

以上の事実を考慮して，BRTLではオートマトン演算子として，補集合を受理するオートマトンへの変換が容易な ldo-fa-1 を用いており，これにより，CTL より強力な表現能力を持つが，モデル検査アルゴリズムの時間計算量はCTLと同等の時相論理になっている．

## 5. BRTL の拡張

本章では，BRTL の拡張である BRTL$^\dagger$ について述べる．BRTL$^\dagger$ はヨなどの経路限定子の有効範囲内で，オートマトン演算子に対するブール演算を許したものである．BRTL$^\dagger$ の表現能力は BRTL と同じであるが，記述を容易にすることができる．

構文はオートマトン演算子の部分のみが変更される．

**定義 10** オートマトン演算子

$B$ を ldo-fa-1, $\{f_1, f_2, \cdots, f_n\} \subseteq BRF$, $A, A_1, A_2$ をオートマトン演算子とする．このとき，(1) $B(f_1, f_2, \cdots, f_n)$ (2) $\neg A$ (3) $A_1 \vee A_2$ を再帰的に適用して得られるすべての式をオートマトン演算子と定義する．

□

上述のオートマトン演算子に対して，補題1と2で述べた手続きを機械的に適用して，$\neg A$ ならば受理状態と非受理状態を交換し，$A_1 \vee A_2$ ならば直積機械を構成することにより，否定も論理和も含まないオートマトン演算子に変換することができる．このようにBRTL$^\dagger$ のあるオートマトン演算子 $A$ (否定および論理和を含みうる) とその部分式に対し，補題1と2で述べた手続きを適用できるかぎり適用して変形した結果を Trans($A$) と記すことにする．このとき，BRTL$^\dagger$ の意味は次のように定義される．

**定義 11**

- $S, s \models (\exists A)$ iff $S, s \models (\exists\ \mathrm{Trans}(A))$ □

表現能力について，定理1より次の定理を証明することができる．

**定理 6** 任意の BRTL$^\dagger$ 式 $f$, 任意の Kripke 構造 $S = \langle \Sigma, I, R, \Sigma_0 \rangle$ および $s \in \Sigma$ に対して，ある BRTL 式 $f'$ が存在して，

$S, s \models f \Leftrightarrow S, s \models f'$ □

したがって，BRTL$^\dagger$ は BRTL と同じ表現能力を持つ．また，モデル検査アルゴリズムについては，オートマトン演算子を否定および論理和演算を含まない形式に変換することにより，アルゴリズム1を用いることができる．BRTL$^\dagger$ に対するモデル検査アルゴリズムの時間計算量については，次の結果が成立する．

**定理 7** 与えられた BRTL$^\dagger$ 式 $f$ に対して，$f$ で用いられている各オートマトン演算子 $A_i$ $(0 \le i \le n)$ が ldo-fa-1 $A_{i1}, A_{i2}, \cdots, A_{ik_i}$ から $\neg$ と $\vee$ を用いて構成されているとする．このとき，$k \overset{\mathrm{def}}{=} \max_{0 \le i \le n} k_i$ とすると，モデル検査アルゴリズムの時間計算量は $O(\mathrm{Size}(S) \times (\mathrm{Len}(f))^k)$ となる．

(略証) $A_1 \vee A_2$ を $A_1 | A_2$ に変換する際に要する時間は，$A_1 | A_2$ の構成から明らかなように，$O(\mathrm{Len}(A_1) \times \mathrm{Len}(A_2))$ となる．これと BRTL の場合の議論より，$O(\mathrm{Size}(S) \times (\mathrm{Len}(f))^k)$ の時間計算量が導かれる． □

## 6. おわりに

本稿では，分岐時間モデルに基づく時相論理として，BRTL とその拡張である BRTL$^\dagger$ を提案した．この時相論理は，表現能力が CTL よりも真に大きく，またモデル検査アルゴリズムの時間計算量が式の長さと Kripke 構造の大きさの積に比例し，CTL の場合と同じである．したがって，CTL が適用できる検証例にはかならず適用でき，さらに，繰り返しなどの性質を記述することもできるという特徴を持つ．

今後の課題としては次のような点が考えられる．まず，本稿で示した ldo-fa-1 は，満たさなければならない制約が必ずしも自明ではないことから，記述はあまり容易ではない．実用的には，構文上の制約により ldo-fa-1 のみが記述可能な言語を構築する必要がある．

また，順序回路を考えた場合，フリップフロップが $n$ 個ならば，その状態数は $2^n$ のオーダとなり，Kripke 構造のようなグラフ形式で表現すると節点数がフリップフロップの数の増加につれ，爆発的に増大する．これに対するアプローチとして，時相論理のモデル検査アルゴリズムを二分決定グラフを用いて行う方法があり[2]，比較的規則性を持つ順序機械であれば大規模な例に対しても検証が可能であることが明らかになっている．文献 2) では CTL を用いているが，BRTL にも適用が可能と考えられ，そのアルゴリズムの構築が課題となっている．

**謝辞** 種々の有益なご議論をいただいた本学高木直史博士，大阪大学石浦菜岐佐博士に深謝いたします．なお，本研究は一部文部省科学研究費による．

## 参考文献


1) Clarke, E. M., Emerson, E. A. and Sistla, A. P.: Automatic Verification of Finite State Concurrent Systems Using Temporal Logic Specifications: A Practical Approach, *10th ACM Symposium on Principles of Programming Languages*, pp. 117-126 (1983).
2) Burch, J. R., Clarke, E. M., McMillan, K. L., Dill, D. L. and Hwang, J.: Symbolic Model Checking: $10^{20}$ States and Beyond, *Proc. of 5th Symp. on Logic in Computer Science*, pp. 428-439 (1990).
3) Lowenstein, P. and Dill, D.: Formal Verification of Cache Systems Using Refinement Relations, *Proc. of ICCD '90*, pp. 228-233 (1990).
4) Fujita, M. and Fujisawa, H.: Specification, Verification and Synthesis of Control Circuits with Propositional Temporal Logic, *Proc. 9th Int. Symp. Computer Hardware Description Languages and Their Applications*, pp. 265-279 (1989).
5) Rescher, N. and Urquhart, A.: *Temporal Logic*, Springer-Verlag, Wien (1971).
6) Hiraishi, H.: Design Verification of Sequential Machines Based on a Model Checking Algorithm of $\epsilon$-free Regular Temporal Logic, *Proc. 9th Int. Symp. Computer Hardware Description Languages and Their Applications*, pp. 249-263 (1989).
7) Dill, D. L. and Clarke, E. M.: Automatic Verification of Asynchronous Circuits Using Temporal Logic, *IEE Proceedings*, Vol. 133, No. 5, pp. 276-282 (1986).
8) Wolper, P.: Temporal Logic Can Be More Expressive, *Proc. of 22nd Annual Symposium on Foundations of Computer Science*, pp. 340-348 (1981).
9) 平石，矢島：正則集合と表現等価な正則時相論理 RTL，情報処理学会論文誌，Vol. 28, No. 2, pp. 117-123 (1987).
10) Sistla, A. P. and Clarke, E. M.: The Complexity of Propositional Temporal Logic, *J. ACM*, Vol. 32, No. 3, pp. 733-749 (1985).
11) Sistla, A. P., Vardi, M. Y. and Wolper, P.: The Complementation Problem for Büchi Automata with Applications to Temporal Logic, *Proc. of ICALP '85*, pp. 465-474 (1985).
12) Büchi, J. R.: On a Decision Method in Restricted Second Order Arithmetic, *Logic Methodology and Philosophy of Science*, pp. 1-12, Stanford University Press (1962).
13) Browne, M. C.: An Improved Algorithm for the Automatic Verification of Finite State System Using Temporal Logic, Technical Report CMU-CS-86-156, Carnegie Mellon University (1986).
14) Hoogeboom, H. J. and Rozenberg, G.: Infinitary Languages: Basic Theory and Applications to Concurrent Systems, *Current Trends in Concurrency, LNCS 224*, pp. 266-342 (1986).
15) Aho, A. V., Hopcroft, J. E. and Ullman, J. D.: *The Design and Analysis of Computer Algorithms*, Addison-Wesley (1974).
16) 浜口，平石，矢島：$\omega$ 正則集合と等価な表現能力を持つ時相論理，電子情報通信学会技術研究報告，COMP88-8 (May 1988).
17) Choueka, Y.: Theories of Automata on $\omega$-Tapes: A Simplified Approach, *Computer and System Sciences*, Vol. 8, pp. 117-141 (1974).


(平成3年1月14日受付)
(平成4年1月17日採録)

**濱口 清治**（正会員）

昭和39年生．昭和62年京都大学工学部情報工学科卒業．平成元年同大学院修士課程修了．平成3年京都大学工学部助手となり現在に至る．論理設計検証，論理設計用CADの研究に従事．電子情報通信学会会員．

**平石 裕実**（正会員）

昭和26年生．昭和48年京都大学工学部電子工学科卒業．昭和50年同大学院修士課程（電気工学第二）修了．同年京都大学工学部情報工学教室助手．昭和59年同教室講師．昭和62年同教室助教授．平成3年京都産業大学工学部教授．工学博士．論理設計検証，論理設計用CAD，計算機グラフィクス等の研究に従事．電子情報通信学会会員．

**矢島 脩三**（正会員）

昭和8年生．昭和31年京都大学工学部電気工学科卒業．同大学院博士課程修了．工学博士．昭和36年より京都大学工学部に勤務，昭和46年情報工学科教授．昭和35年京都大学第一号計算機KDC-1を設計稼動，以来，計算機，論理設計，オートマトン等の研究教育に従事．著書は「電子計算機の機能と構造」（岩波，57年）等．本学会元常務理事，元会誌編集委員（地方），元JIP編集委員．電子情報通信学会元評議員およびオートマトンと言語研専元委員長，North-Holland出版 元IPL編集委員，IEEE Senior Member.