4K－6

線形時間のモデルチェックアルゴリズムを持つ正則時相論理と変数代入機構による拡張

濱口 清治　　平石 裕実　　矢島 脩三

京都大学 工学部

1 はじめに

順序機械など有限状態システムの設計が正しいことを保証するための検証手法の確立が重要な課題となっている。我々はこれまで 正則時相論理 (RTL, Regular Temporal Logic)[2] のモデルチェックアルゴリズムを示し、またこれに基づき実際に順序機械のの検証を行なってきた。

RTL は任意の有限オートマトンを特徴づけることができるが、モデルチェックの計算複雑度は非初等的と大きい。また、CTL (Computational Tree Logic) [1] は、有限状態システムの動作を反映する Kripke 構造 S の大きさ ($|S|$) と時相論理式 f の長さ ($|f|$) の積に対し線形時間でモデルチェックを行うことができるが、表現能力が低く有限オートマトンを特徴づけることができない。

本稿ではまず CTL よりも高い表現能力を持つ RRTL (Restricted Regular Temporal Logic) を提案し、CTL と同じく $|S|$ と $|f|$ の積に比例する時間計算量を持つモデルチェックアルゴリズムを示す。次に、時相論理式による記述の簡潔にするため、RRTL にブール変数への代入とそれらの値を参照する機構を導入した ER-RTL(Extended RRTL) を定義し、$O(|S|\times|f|\times 2^{|V|})$ (V はブール変数の集合) の時間計算量を持つモデルチェックアルゴリズムを示す。

2 Restricted Regular Temporal Logic

定義 1 論理型決定性 ω 有限オートマトン (以下、*ldo-fa* と記す) $A = (Q, \Sigma, P, \delta, q_0, F)$ を次のように定義する。Q と $\Sigma = \{0,1\}^n$ は それぞれ状態と入力記号の集合、$\delta : Q \times BF \to Q$ は状態遷移関数 (BF は 原始命題の集合 $P = \{p_1, \cdots, p_n\}$ から構成されるブール式)、q_0 は初期状態、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。

ただし、原始命題への任意の真理値の割り当てによって、1つの状態から出ていく枝にラベルされているブール式2つ以上が同時に真になることはなく、また、状態 q に対して $v \in \{0,1\}^n$ が入力された時、真になるブール式のラベルされた枝を遷移するものとする。

$Inf(x)$ を A に $x \in \Sigma^\omega$ を入力した時無限回通過する状態の集合とすると、A が受理する言語 は $\{x | Inf(x) \cap F$ が空でない $\}$ と定義される。 □

次の条件1を満足する ldo-fa は受理状態と非受理状態の入換え操作のみで、補集合を受理する ldo-fa になる。

条件 1 どの非受理状態からも受理状態への遷移がない。

定義 2 AP を原始命題の集合とする。このとき、$S = \langle \Sigma, I, R, \Sigma_0 \rangle$ を Kripke 構造と呼ぶ。ここで、Σ は状態の集合、$I : \Sigma \to 2^{AP}$ は各状態での原始命題の真偽を割り当てる関数、$R \subseteq \Sigma \times \Sigma$ は Σ 上の有向枝の集合 (任意の状態から少なくとも1本の枝が出ているとする) であり、Σ_0 は初期状態の集合である。 □

定義 3 シンタクス

RRTL 式 (その集合を RF と記す)

$p \in AP$、$f, g \in RF$、また A がオートマトン演算子であれば、$p \in RF$、$\neg f, f \vee g \in RF$、$\exists A$ および $\exists\neg A \in RF$ である。

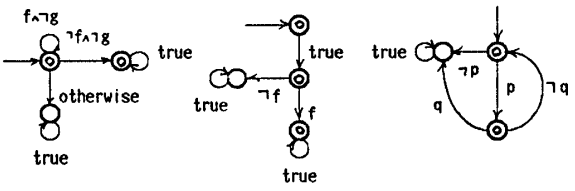

図 1: 時相演算子に対応するオートマトン演算子

オートマトン演算子

A が条件1を満足する ldo-fa ならば、A はオートマトン演算子である。ただし、遷移にラベルづけされる式は RRTL 式であるとする。

セマンティクス

Kripke 構造 $S = \langle \Sigma, I, R, \Sigma_0 \rangle$ に対して定義する。$S, s \models f$ は RRTL 式 f が状態 $s \in \Sigma$ に対して真であることを意味する。$p \in AP$、f と g は RRTL 式、A はオートマトン演算子とする。

- $S, s \models p$ iff $I(s) \ni p$
- $S, s \models f \vee g$ iff $S, s \models f$ or $S, s \models g$
- $S, s \models \neg f$ iff $S, s \not\models f$
- $S, s \models \exists A(\exists\neg A)$ iff Kripke 構造 S 上の状態 s から始まる無限系列 $\sigma = s_0 s_1 s_2 \cdots$ と ldo-fa A の状態の無限系列 $q_0 q_1 q_2 \cdots$ があって次の条件を満足している $(i = 0, 1, \cdots)$。RRTL 式 f_i が q_i から q_{i+1} への遷移にラベルされていて、$S, s_i \models f_i$ かつ $\forall i. q_i \in F$ $(\exists i. q_i \notin F)$.

ブール演算子 $\wedge$, $\equiv$ および $\Rightarrow$ を通常の意味で用いるものとし、さらに $\forall A \stackrel{\text{def}}{\equiv} \neg\exists\neg A$、$\forall\neg A \stackrel{\text{def}}{\equiv} \neg\exists A$ と定義する。$\forall(\exists)$ は、Kripke 構造上のすべての (ある) 経路を意味する経路限定子である。

$B(f,g)$ および $C(f)$ をそれぞれ図1 (1) と (2) に示すオートマトン演算子とする。

このとき、CTL の時相演算子 U(until) および $\bigcirc$(next) は

$$\forall(fUg) \stackrel{\text{def}}{\equiv} \neg\exists B(f,g) \quad \forall\bigcirc(f) \stackrel{\text{def}}{\equiv} \forall C(f)$$

と定義することができる (直観的には、fUg は g が成り立つまではどの状態でも f が成立することを、$\bigcirc f$ は次の状態で f が成立することを意味する)。$\exists$ についても同様の定義が可能であり、また $\Box$(always)、$\Diamond$(eventually) も U を用いて定義することができる。このことは RRTL が CTL を包含する体系であることを示している。

また、RRTL では繰り返しの概念を含む性質も記述することができ、図1の (3) の D のようなオートマトン演算子を用いると $\forall D$ によって、Kripke 構造上の全経路で「p がある時刻で 1 で次の時刻で q が 0」が繰り返すという性質を表現できる。(1 を真、0 を偽とみなしている)

アルゴリズム 1 モデルチェックアルゴリズム (RRTL)

A Regular Temporal Logic for Model Checking with Linear Time Complexity and Its Extension with a Substitution Mechanism

Kiyoharu HAMAGUCHI, Hiromi HIRAISHI, Shuzo YAJIMA

Kyoto University

```
1 .procedure BoolCheck(s,f)
2 .   if f ∈ AP return ((I(s) ∋ f))
        /* I(s) ∋ f ならば、'T' を返す */
3 .   if f = f1 ∨ f2
        return (BoolCheck(s, f1) ∨ BoolCheck(s, f2))
4 .   if f = ¬g return (¬ BoolCheck(s, g))
5 .   if f = ∃A return (Check_a(s, p))
6 .   if f = ∃¬A return (Check_b(s, p))
7 .end of procedure

1 .procedure Check_a(s,p) (Check_b(s,p))
2 .   x = Label(s,p)
3 .   if x = 'T' then return 'T'
4 .   if x = 'F' then return 'F'
5 .   if x = 'C' then return 'T' ('F')
6 .   Addlabel(s,p,'C');
7 .   p'=Transit(s,p);
8 .   if p' points to a rejecting state then y = 'F' ('T');
9 .   else
10 .     for all s' such that (s,s')∈ R {
11 .       if Check(s',p') = 'T' then y = 'T'; }
12 .  if y = 'T' then Addlabel(s,p,'T'); return 'T';
13 .  else Addlabel(s,p,'F'); return 'F';
14 end of procedure
```

図 2: RRTL のモデルチェックアルゴリズム

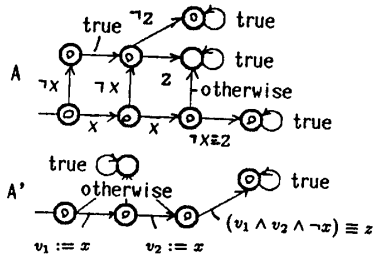

図 3: オートマトン演算子 A と A'

- 入力: Kripke 構造 S および RRTL 式 f
- 出力: S の全初期状態 s に対し $S, s \models f$ ならば、yes。そうでなければ、no。
- 方法: S のすべての初期状態に対し、図 2に示した BoolCheck(s, f) を呼び出す。$S \models f$ iff "BoolCheck が 'T' のみを返す"。

s は Kripke 構造上の状態、p はオートマトン式 A の状態を指すポインタである。図 2では、() 内が Check_b に対応している。

Transit(s,p): p の指す状態から、$S, s \models f_e$ であるような f_e がラベルされている枝を遷移した先の状態を指すポインタを返す関数。

Addlabel(s, p, X) : 状態 s に (p, X) をラベルする手続き。ただし、X = 'T', 'F' or 'C'。'C' はチェック中であることを示す。

Label(s, p): s に (p, X) がラベルされていれば X を、そうでなければ NIL を返す関数。

ここで、オートマトン演算子の大きさをその状態数と遷移の枝の数の和とし、また RRTL 式 f の長さ $|f|$ は f 中のシンボル数とする。ただし、各オートマトン演算子記号の長さはその大きさで評価する。

命題 1 アルゴリズム 1 の時間計算量は $O(|S| \cdot |f|)$。

(証明の概略) 次の 2 つの性質から証明される。

- 任意のポインタと状態の対 (p, s) に対し, Check(p, s) は S の各 (s, s') について一度しか呼び出されない。
- 異なるポインタの数はせいぜい $|f|$ である。 □

3 変数代入機構を持つ時相論理

CTL や RRTL にはモデルチェックが容易という特徴があるが、その反面、記述が繁雑になることがある。たとえば、信号線 x 上に 110 の系列を検出した時そしてその時にのみ信号線 z の値が 1 になるという系列検出器に対する条件は、RRTL では、図 3 (1) のオートマトン演算子 A を用いて $\forall\Box(\forall A)$ と記述することができる。ここで、ブール変数への値の代入とそれらの値の参照を許すと、図 3 (2) のオートマトン演算子 A' により、$\forall\Box(\forall A')$ と記述でき、オートマトン演算子の表現が簡潔になる。ここで v_1 と v_2 はブール変数であり、$v_1 := x$ は x の値 (1 または 0) を v_1 に代入することを意味している。

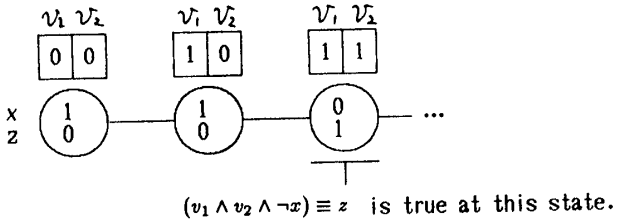

図 4: $\exists A'$ を真にする系列 (Kripke 構造) の例

定義 4 ERRTL(Extended RRTL) のシンタクス

V をブール変数の集合、ERRTL の AP は V を含むものとする。ERRTL 式は RRTL 式と同様に定義される。オートマトン演算子については以下の点で異なる。

- 遷移にラベルづけされる式は ERRTL 式である。
- 1 つ以上の代入 $v_i := f$ (f は ERRTL 式) を遷移にラベルすることができる。ただし、1 遷移に対し v_i への代入は一度のみとする。 □

セマンティクスの形式的な定義は別稿に譲ることとし、ここでは、図 3 (2) の A' に対して、$\exists A'$ を真にする系列 (Kripke 構造) を図 4に示す。各状態にラベルされている値の組は、v_1 と v_2 の値を格納している。各ブール変数の初期値はあらかじめ与えなくてはならない。この例では、$v_1 = v_2 = 0$ が初期値である。

ある状態 s における代入は次の状態 s' でのブール変数の値を決定し、s でのブール変数の値は変更しない。

アルゴリズム 2 モデルチェックアルゴリズム (ERRTL)

- 入力: Kripke 構造 S, ERRTL 式 f およびブール変数の初期値 $u \in 2^V$
- 出力: S のすべての初期状態に対して $S, s, u \models f$ ならば、yes、そうでないならば、no。
- 方法: アルゴリズム 1 を次のように変更する。
  1. ポインタ p とブール変数の値の組 u の対 (p, u) を S の各状態にラベルする。アルゴリズム 1 中の手続き Label は (p, u) がすでに s にラベルされている時、X の値を返す。
  2. u は手続き Transit が、状態 s(S 上) と p が指す状態 (オートマトン演算子上) からの遷移に応じて書き換える。 □

命題 2 アルゴリズム 2 の時間計算量は $O(|S| \cdot |f| \cdot 2^{|V|})$

これは、ラベル (p, u) の数が最大 $|f| \times 2^{|V|}$ しかないことより証明される。 □

4 おわりに

CTL より高い表現能力を持ち、モデルチェックが容易な時相論理 RRTL を示した。また、ブール変数への代入機構を導入した ERRTL を提案した。

謝辞 種々御議論頂いた矢島研究室の皆様に深謝します。

参考文献

[1] E. M. Clarke, E. A. Emerson, and A. P. Sistla. Automatic Verification of Finite State Concurrent Systems Using Temporal Logic Specifications: A Practical Approach. In *10th ACM Symposium on Principles of Programming Languages*, pages 117–126, January 1983.

[2] H. Hiraishi. Design Verification of Sequential Machines Based on a Model Checking Algorithm of ϵ-free Regular Temporal Logic. In *Computer Hardware Description Languages and their applications*, pages 249–263, June 1989.