

時相論理と言語階層の対応関係について

On Relationship between Temporal Logics and Language Hierarchy

濱口 清治 平石 裕実 矢島 脩三

Kiyoharu HAMAGUCHI, Hiromi HIRAISHI, Shuzo YAJIMA

京都大学工学部情報工学教室

Department of Information Science, Faculty of Engineering, Kyoto University

1 はじめに

時相論理は、古典的な論理体系に時間の概念を表すための様相演算子を加えたものであり、命題論理をもとにした体系と述語論理をもとにした体系がある。とくに、「常に」「～がなりたつまでずっと」などの概念を表す演算子をもつ命題論理の種々の体系は、有限状態機械の動作と対応づけることができ、また充足可能性判定問題や系列集合に対する論理式の真偽値判定問題が決定可能という特徴がある。このため、有限状態機械の動作の形式的記述、設計検証にもちいられてきた [1, 2]。

また、この過程で古典的な時相論理表現能力を改善するため、さまざまな拡張が試みられてきている。しかし、これらの拡張は有限状態の機械を記述できるよう拡張したもの、すなわち正則集合と同等な表現能力をもつように改良したものである [3]。

正則言語よりも広い言語のクラスを記述可能な命題時相論理では、充足可能性判定などの問題が、決定不能になるという問題があるが、記述言語という観点からみたとき、命題時相論理はどのような拡張により、どの程度の記述能力をもつことができるかという点は興味深い。

本稿では、より広い言語のクラスを表現する命題論理として、ITL [4] を取り上げ、そのサブクラスとして、正則言語、文脈自由言語と等価な表現能力を持つクラスを不動点演算子を用いて特徴づける。

2 区間時相論理 (Interval Temporal Logic, ITL)

ITL は Moszkowski によって提案された時相論理である [4]。通常の命題時相論理では、原子命題の真偽が割り当てられた状態の系列に対し、論理式の真偽値が定められるが、ITL では区間 (interval) に対して、原子命題の真偽値が割り当てられており、これをもとに論理式の真偽値もさだめられる。

論理式に対応する系列の集合を考えた場合、系列上の各区間で原子命題の真偽値が定まるため、言語集合との対応をとりにくいという問題がある。そこで、本稿では、以下の QITL を用いることにする。

定義 1 QITL(Quantified Interval Temporal Logic)

構文： 原子命題 $p \in AP$, 命題変数 $v \in PV$, $\neg w$, $w_1 \vee w_2$, $\bigcirc w$, $w_1 ; w_2$, $\exists v.w$ は全て QITL の論理式である (ただし、 w, w_1, w_2 は QITL の論理式)。

論理式に命題変数の自由な出現がないとき、この論理式を QITL の文という。

モデル： 原子命題と命題変数の真偽値は区間に対して定まる。

Σ を状態の集合、 M を各区間 $s_0 s_1 \cdots s_n \in \Sigma^+$ に原子命題と命題変数の真偽値 $M_{s_0 s_1 \cdots s_n}[u]$ $u \in AP \cup PV$ を定める写像としたとき、 (Σ, M) をモデルとよぶ。

区間 $s_0 s_1 \cdots s_n \in \Sigma^+$ の長さは、 n であると定義する。本稿での定義では、長さ 0 の区間 $s \in \Sigma$ は考慮しない。すなわち、かならず、1 以上の長さの区間 $s_0 s_1 \cdots s_n$ ($n \geq 1$) のみを対象にしている。

原子命題 $p \in AP$ に対する真偽値の割り当てに関しては、次の条件が仮定される。

$$M_{s_0 s_1 \dots s_n}[p] = M_{s_0 s_1}[p]$$

これは、原子命題の区間に対する真偽値は、区間の最初の長さ 1 の部分で決定することを意味する。

意味：文に対してのみ定められる。

- $M_{s_0 \dots s_n}[\neg w] = \text{true} \Leftrightarrow M_{s_0 \dots s_n}[w] = \text{false}$
- $M_{s_0 \dots s_n}[w_1 \vee w_2] = \text{true} \Leftrightarrow M_{s_0 \dots s_n}[w_1] = \text{true}$ または $M_{s_0 \dots s_n}[w_2] = \text{true}$
- $M_{s_0 \dots s_n}[\bigcirc w] = \text{true} \Leftrightarrow n \geq 2$ かつ $M_{s_1 \dots s_n}[w] = \text{true}$
- $M_{s_0 \dots s_n}[w_1; w_2] = \text{true} \Leftrightarrow 1 \leq i \leq n-1$ なる i が存在して、 $M_{s_1 \dots s_i}[w_1]$ かつ $M_{s_{i+1} \dots s_n}[w_2]$.
- $M_{s_0 \dots s_n}[\exists v. w] = \text{true} \Leftrightarrow v$ に対する真偽値の割り当てのみが M と異なるモデル M' が存在して、 $M'_{s_0 \dots s_n}[w] = \text{true}$

□

$M_{s_0 \dots s_n}[w]$ を $s_0 \dots s_n \models w$ と書く場合もある。

定義 2 論理式 w が定義する言語 $L(w)$

$\Delta = 2^{AP}$ としたとき、 $L(w)$ は $\Delta^\dagger$ 上の言語で、次の条件を満足するものと定義する。

$$L(w) = \{\delta = d_1 d_2 \dots d_n \mid s_0 \dots s_n \models w \text{ かつ } d_i = \{p \mid s_{i-1} s_i\}\}$$

□

略記法として、 $\wedge, \equiv, \Rightarrow$ も通常の意味で用いる。また、次の記法を用いる。

- $\Diamond_a w = (V_T; w; V_T) \vee (w; V_T) \vee (V_T; w) \vee w$
- $\Box_a w = \neg \Diamond_a (\neg w)$
- $\Diamond_t w = (V_T; w) \vee w$
- $\Box_t w = \neg \Diamond_t (\neg w)$

- $skip = \neg \bigcirc true$ (長さ 1 の区間を表す)

- $\underline{w} = skip \wedge w$

3 不動点演算子

本節では、次節以降で用いる不動点の性質を述べる。

$\sqsubseteq$ は $2^{\Delta^+} \times 2^{\Delta^+}$ 上の半順序関係とし、 $x \sqsubseteq y \Leftrightarrow x \subseteq y$ と定める。このとき 2^{Δ^+} はこの半順序関係のもとで完備束になる。 $\top$ と $\perp$ はそれぞれ 2^{Δ^+} でのトップとボトムである。(i.e. $\top = \Delta^+$ また $\perp = \emptyset$).

$D = 2^{\Sigma^+}$ とする。任意の $x, y \in D$ に対して、 $x \sqcup y$ と $x \sqcap y$ は $\{x, y\}$ のそれぞれ上限と下限である。集合 $X \subseteq D$ に対して $\sqcup X$ と $\sqcap X$ はそれぞれ X の上限と下限を表す。

定義 3 $\varphi : D \rightarrow D$ が *monotonic* $\Leftrightarrow \forall x, \forall y \in D, x \sqsubseteq y$ ならば、 $\varphi(x) \sqsubseteq \varphi(y)$. □

定義 4 $\varphi : D \rightarrow D$ が $\sqcap$ -連続 $\Leftrightarrow \sqcap\{\varphi(x_i) | i \geq 0\} = \varphi(\sqcap\{x_i | i \geq 0\})$ が D 上の任意の単調減少列 $x_0 \sqsupseteq x_1 \sqsupseteq x_2 \dots x_i \sqsupseteq x_{i+1} \dots$ に対して成立する。

$\varphi : D \rightarrow D$ が $\sqcup$ -連続 $\Leftrightarrow \sqcup\{\varphi(x_i) | i \geq 0\} = \varphi(\sqcup\{x_i | i \geq 0\})$ が D 上の任意の単調増加列 $x_0 \sqsubseteq x_1 \sqsubseteq x_2 \dots x_i \sqsubseteq x_{i+1} \dots$ に対して成立する。 □

定理 1 (不動点定理) φ が $\sqcap$ -連続ならば、一意に定まる φ の最大不動点 $(\nu x. \varphi(x))$ が存在し、次の等式を満足する。

$$\nu x. \varphi(x) = \sqcap_{i \geq 0} \varphi(x_i),$$

ここで、 $x_0 = \top$ また $x_{i+1} = \varphi(x_i)$ ($i \geq 0$)。

φ が $\sqcup$ -連続ならば、一意に定まる φ の最小不動点 $\mu x. \varphi(x)$ が存在し、次の等式を満足する。

$$\mu x. \varphi(x) = \sqcup_{i \geq 0} \varphi(x_i),$$

ここで、 $x_0 = \perp$ かつ $x_{i+1} = \varphi(x_i)$ ($i \geq 0$)。 □

4 正則言語と不動点

本節の議論は、文献 [5] を参考にしている。

2 節で定義したように、ある ITL 式は $\Delta = 2^{AP}$ をアルファベットとする系列の集合をあらわしている。さらに、ITL の命題変数 v は系列に対して真偽が定まることから、 2^{Δ^+} 上の変数とみなすことができる。

ITL の部分クラスに最小不動点演算子を導入して、次の RITL (Regular Interval Temporal Logic) を構成する。

定義 5 RITL

構文：原子命題、 $\bigcirc w$, $\neg w$, $w_1 \vee w_2$ および、 $\mu v. \Psi(v)$ が RITL の文である。ここで、 $\Psi(v)$ は命題変数 v を含む RITL 式であり、 v は偶数回の否定のもとで出現する。

意味： $\mu v. \Psi(v)$ についてのみに示す。

- $M_{s_0 \dots s_n}[\mu v. \Psi(v)] = \text{true} \Leftrightarrow s_0 s_1 \dots s_n$ が Ψ の最小不動点に含まれる

□

不動点演算子に関して次の定理を示すことができる。

定理 2 $M_{s_0 \dots s_n}[\mu v. \Psi(v)] = \text{true} \Leftrightarrow \exists v. [v \wedge \Box_t(v \Rightarrow \Psi(v))]$

□

したがって、RITL は QITL に含まれる。

定理 3 任意の正則集合 R に対して、 $L(w) = R$ なる RITL 式 w が存在する。

□

上述の定理に関して例を示す。正則文法として、

$$\begin{aligned} V_1 &\rightarrow a|bV_1|cV_2 \\ V_2 &\rightarrow d|eV_1|fV_2 \end{aligned}$$

を考える。ここで、 a, b, c, d, e, f は終端記号、 V_1 と V_2 は非終端記号である。

$\Delta = 2^{AP}$ の要素を a, b, c, d, e, f に割り当て、原子命題を使って、 $p_a = \neg p_1 \wedge \neg p_2 \wedge \neg p_3$

($p_i \in AP$) などと表現する。 V_1, V_2 に対して、命題変数 v_1, v_2 を対応させると、

$$\mu v_1.(\underline{p_a} \vee (\underline{p_b} \wedge \bigcirc v_1) \vee (\underline{p_c} \wedge \bigcirc(\mu v_2.(\underline{p_d} \vee (\underline{p_e} \wedge \bigcirc v_1) \vee (\underline{p_f} \wedge \bigcirc v_2))))))$$

により、 V_1 が生成する言語が表現できる。

この定理の逆も成立する。

定理 4 任意の RITL 式 w に対して、 $L(w)$ は正則集合である。

(略証) w には、 $\bigcirc, \vee, \neg$, および $\mu v. \Psi(v)$ が用いられうる。 w, w_1, w_2 が正則集合ならば、 $\bigcirc w, w_1 \vee, \neg w$ の表現する言語は、正則集合である。

また、 $\Psi(v)$ については、 $\vee$ と $\bigcirc$ の連続性と v に対する否定の出現の仕方から連続性が示せる。したがって、不動点定理より、 $\mu v. \Psi(v) = \Psi(\emptyset) \cup \Psi(\Psi(\emptyset)) \cup \dots$ であるから、 $\mu v. \Psi(v)$ も正則集合になることを示せる。 $\square$

5 文脈自由言語と不動点

前節の RITL に含まれていない 接続演算 ; を不動点演算子中に書くことを許したものが次の CFITL である。

定義 6 CFITL(Context Free Interval Temporal Logic)

構文: 原子命題、命題変数、 $\bigcirc w, w_1 \vee w_2, w_1; w_2$ および、 $\mu v. \Psi(v)$ が CFITL 式である。

ここで、 $\Psi(v)$ は命題変数 v を含む CFITL 式である。(否定は含まない。)

意味: $\mu v. \Psi(v)$ についてののみ示す。

- $M_{s_0 \dots s_n}[\mu v. \Psi(v)] = \text{true} \Leftrightarrow s_0 s_1 \dots s_n$ が Ψ の最小不動点に含まれる

$\square$

定理 5 $M_{s_0 \dots s_n}[\mu v. \Psi(v)] = \text{true} \Leftrightarrow \exists v. [v \wedge \bigcirc_a (v \Rightarrow \Psi(v))]$

$\square$

定理 6 任意の文脈自由言語 R に対して、 $L(w) = R$ なる CFITL 式 w が存在する。 $\square$

例を示す。文脈自由文法として、

$$\begin{aligned} V_1 &\rightarrow aV_2b|ab \\ V_2 &\rightarrow cV_1d|cd \end{aligned}$$

を考える。ここで、 a, b, c, d は終端記号、 V_1 と V_2 は非終端記号である。

このとき、 V_1 の生成する言語を表現する CFITL 式は、

$$\mu v_1(\underline{p_a}; \mu v_2.(\underline{p_c}; v_1; \underline{p_d} \vee \underline{p_c}; \underline{p_d}); \underline{p_b} \vee \underline{p_a}; \underline{p_b})$$

となる。

定理 7 任意の CFITL 式 w に対して $L(w)$ は文脈自由言語である。

(略証) CFITL が否定を含まず、また、文脈自由言語が接続、和、代入に関して閉じていることと、; 演算が連続であることを用いて、定理 4 と同様にして示せる。 $\square$

また、文脈自由文法は共通集合をとる演算に関して閉じていないため、 $\wedge$ を許している QITL は CFITL を真に含んでいることがわかる。

6 おわりに

本稿では、QITL のサブクラスとして、正則言語および文脈自由言語と表現能力が等価な体系を示した。これらの関係は、 $\text{RITL} \subset \text{CFITL} \subset \text{QITL}$ となる。RTIL と CFITL は、許されている時相演算子の種類 ($\bigcirc$ か ;) のぞくと、ほぼ同様の形式で不動点演算子を用いて定式化できる。

文脈依存言語については、文法規則の左辺に 2 つ以上の非終端記号が出現するため、同様の定式化はできないと思われる。また、従来の時相論理で時間に関する概念を表現するために導入された演算では、文脈依存言語を表現することは困難ではないかと考えられる。

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.
 - [3] P. Wolper. Temporal Logic Can Be More Expressive. In *Proceedings of 22nd Annual Symposium on Foundations of Computer Science*, pages 340–348, 1981.
 - [4] B. Moszkowski. Reasoning about Digital Circuits. Technical Report STAN-CS-83-970, Stanford University, 1983. Ph.D thesis.
 - [5] M. Y. Vardi. A Temporal Fixpoint Calculus. In *Proceedings of 5th ACM Symposium on Principle on Prangramming Languages*, January 1988.