更新可能時間オートマトンの新たな拡張について

上里 友弥　南出 靖彦

本発表では，時間オートマトン（Timed automata）の拡張として知られる，更新可能時間オートマトン（Updatable timed automata）について，その拡張となる新たな体系を導入し到達可能性問題の決定可能性を証明する．

従来の更新可能時間オートマトンでは，更新の形を限定されたものとした場合に限り，到達可能性問題が決定可能となることが示されている．仮に一般的な形の更新を許した場合には，チューリング完全となり問題は決定不能となる．我々は，新たな体系として，より一般的な更新形式を持つ代わりに，従来の対角線制約（diagonal constraints）を弱めたものを提案する．この体系に対して新たな証明手法を用いることで，到達可能性問題の決定可能性を示す．同時に我々の体系は，言語のクラスを拡大すること，すなわち，従来の時間オートマトンでは受理できなかった言語が扱えることも述べる．

We propose a new extension of Updatable Timed Automata (UTA), which extend the formalization of standard timed automata, and prove the decidability of the reachability problem of our new extension.

The decidability of the reachability problem for the original formalization of UTA was shown for very restricted sub-classes. This restriction is unavoidable because general UTA can simulate Minsky's two-counter machines and are Turing-complete. By contrast, our formalization adopts another set of formulae and this makes our system more expressive with respect to the decidable classes of UTA. Especially, we use a novel technique to prove the decidability of the reachability problem of our updatable timed automata.

1　はじめに

時間オートマトンは，従来の有限オートマトンに，時間の概念を付け加えたものである．

システムのモデル化の観点からは，有限オートマトンとは，ある状態 p で，イベント a をみたときに，次にどの状態 q に移るか，を定める規則 $p \xrightarrow{a} q$ からなる計算モデルだといえる．一方で時間オートマトンは，直前のイベントと次のイベントの間の経過時間を考える．また，そのような時間概念を扱うために，従来の状態に加えて有限個の「クロック」を用いる．以下は時間オートマトンの計算の一例である：

$\langle p, \{x \mapsto 0\}\rangle \xrightarrow{1.2} \langle p, \{x \mapsto 1.2\}\rangle \xrightarrow{a} \langle q, \{x \mapsto 1.2\}\rangle$.

これは，状態 p とクロック x（その値を 0 とする）の組 $\langle p, \{x \mapsto 0\}\rangle$ から計算が開始され，1.2 秒経過したところで，イベント a をみたことを意味する．

時間オートマトンの体系的な研究は，Alur と Dill による論文[4] に端を発する．特にこの論文では，モデル検査で基礎となる判定問題「到達可能性問題」の決定可能性（PSPACE 完全性）が証明された．その後も，時間オートマトンは実時間システムのための重要な計算モデルであるという認識のもと，非常に様々な拡張が提案されてきている．これについては，例えば[11][18] などのサーベイが参考になる．

本論では，そのような拡張のうち，Bouyer らによって提案された「更新可能」時間オートマトン[7] と呼ばれる体系を，一層拡張する．時間オートマトンでは，クロック x の値を変更する操作として，その値を 0 にリセットする $x := 0$ のみが許されていた．一方，更新可能時間オートマトンは，リセット以外の操作—これを更新と呼ぶ—も許す拡張である．Bouyer

---

New Extension of Updatable Timed Automata.

Yuya Uezato, 筑波大学システム情報工学研究科, University of Tsukuba.

Yasuhiko Minamide, 東京工業大学大学院情報理工学研究科, Department of Mathematical and Computing Sciences, Tokyo Institute of Technology.

らによって研究された体系で到達可能性問題が決定可能なものは，結果的には，従来の時間オートマトンと表現力が変わらないことが示されている [7] [6].

本論で提案する更新可能時間オートマトンでは，以下の二点が重要である：

1. 2 章に現れる L_{ab} のような，従来の時間オートマトンでは受理できない言語を受理できる．
2. 到達可能性問題が決定可能である．

従って，我々の体系は，時間オートマトンの言語クラスを超えた新しい更新可能時間オートマトンであるといえる．

到達可能性問題の決定可能性証明は，*Well-Quasi-Ordering* という非常に良い性質を持つ順序構造が，我々の体系に現れるということから導かれる．本論で与える証明は，近年精力的に研究が行われている *Well Structured Transition System (WSTS)* と呼ばれる体系の基礎となる証明手法と，本質的に一致することにも触れる．特に WSTS の代表的な例である *Lossy Counter Machine* からの還元を用いることで，我々の到達可能性問題が Non Primitive Recursive と呼ばれる，計算に莫大な時間を要するクラスに属することを述べる．

1.1 記法

非負有理数全体を $\mathbb{Q}_{\geq 0} \triangleq \{r \in \mathbb{Q} : r \geq 0\}$ で書くことにする．非負有理数 $r \in \mathbb{Q}_{\geq 0}$ について，$fr(r)$ を，r の小数部を表すために用いる．例えば，$fr(1.5) = 0.5$ である．

また，文字 ν を非負有理数上の割当（クロックの持つ値の表現）$\nu : \mathcal{X} \to \mathbb{Q}_{\geq 0}$ のために用いる．割当を記述するために，記法 $\nu = \{x \mapsto 0.5, y \mapsto 1.3\}$ を用いる．割当 ν 上のクロック x の値を，非負有理数 r で上書きして，新しい割当を作ることを $\nu[x := r]$ で書く．非負有理数 δ と割当 ν について，全てのクロック値を δ だけ増やして得られる割当を $\nu + \delta$ で書く．正確には $(\nu + \delta)(x) \triangleq \nu(x) + \delta$ で定義される．クロックの集合 $\mathcal{X}$ について全てに 0 を割り当てた特殊な割当を $\mathbf{0}_{\mathcal{X}}$ で書く：$\forall x \in \mathcal{X}.\ \mathbf{0}_{\mathcal{X}}(x) = 0.0$.

2 更新可能時間オートマトン

更新可能時間オートマトン（Updatable Timed Automaton，以下で UTA と略す）は，6 つ組 $\mathcal{A} = (Q, q_{\text{init}}, F, \mathcal{X}, \Sigma, \Delta)$ であり，構成要素は以下の通りである：

- Q は状態の有限集合で，$q_{\text{init}} \in Q$ と $F \subseteq Q$ はそれぞれ，初期状態及び最終状態の集合を表す．
- $\mathcal{X}$ はクロックの有限集合を表す．
- Σ は有限の入力アルファベットで，Δ は下で詳しく見るが，遷移規則を定める有限の集合である．

UTA $\mathcal{A}$ の計算状況は $\langle q, \nu \rangle$ で表され，$q \in Q$ は現在の状態，$\nu : \mathcal{X} \to \mathbb{Q}_{\geq 0}$ は各クロック $x \in \mathcal{X}$ の現在の値 $\nu(x)$ を保持する割当関数である．

各遷移規則は $p \xrightarrow{a, \phi, \mathbf{up}} q \in \Delta$ の形をとり，$p, q \in Q$ は状態で，入力文字 $a \in \Sigma$ を読んだのちに，

1. 制約式 ϕ を現在の計算状況が満たしているかどうか，を検査し；
2. 満たしている場合には，更新式（の集合）**up** に基づき計算状況を更新する

ことで遷移を行う．遷移の正確な定義のために，まずは制約式と更新式を定義する．

2.1 制約式

クロックの集合 $\mathcal{X}$ 上の制約式全体 $\mathbb{C}(\mathcal{X})$ を，以下の文法で与える：

$$\phi ::= x \bowtie k \mid x \bowtie y \mid fr(x) = 0 \mid fr(x) \bowtie fr(y) \mid \neg\phi \mid \phi \wedge \phi.$$

ただし，$\bowtie \in \{<, =, >\}$ は二項関係を表す記号で，$k \in \mathbb{N}$ と $x, y \in \mathcal{X}$ を満たすことにする．

$x \bowtie y$ は，時間オートマトンで対角線制約（diagonal constraints）と呼ばれるものの特殊な形で，2 つのクロックの値を比較するために用いる．一方で $fr(x) = 0$ や $fr(x) = fr(y)$ は，クロックの小数部を調べるために用いる．具体的な定義は以下で与え，これらの制約式と既存の体系の比較は，この後に 2.5 節で与える．

割当 ν と制約式 ϕ について，以下の場合分けのもと，ν が ϕ を成立させることを意味して $\nu \models \phi$ と書く：

- $\nu(x) \bowtie k$ のとき，$\nu \models x \bowtie k$ と書く．
- $\nu(x) \bowtie \nu(y)$ のとき，$\nu \models x \bowtie y$ と書く．
- $fr(\nu(x)) = 0$ のとき，$\nu \models fr(x) = 0$ と書く．同様に，$fr(\nu(x)) \bowtie fr(\nu(y))$ のとき，$\nu \models fr(x) \bowtie fr(y)$ と書く．
- $\nu \models \phi$ で「ない」とき，$\nu \models \neg\phi$ と書く．
- $\nu \models \phi_1$ かつ $\nu \models \phi_2$ のとき，$\nu \models \phi_1 \wedge \phi_2$ と書く．

2.2 更新式と割当の更新

クロック $x \in \mathcal{X}$ に対する更新式 θ は，新しいクロック $\overline{x}$ を含む制約式 $\theta \in \mathbb{C}(\mathcal{X} \cup \{\overline{x}\})$ である．クロック x の更新式 θ は，更新に際してクロック x の値をどのように変更するかを定めている．形式的には，$\mathcal{X}$ 上の割当 ν について，新しい値の候補を以下で定める：

$$\theta(\nu) \triangleq \{r \in \mathbb{Q}_{\geq 0} : \nu \cup \{\overline{x} \mapsto r\} \models \theta\}.$$

$\mathcal{X}$ 上の割当 ν について，各クロック $x_i \in \mathcal{X}$ についての更新式 $\theta_i \in \mathbb{C}(\mathcal{X} \cup \{\overline{x}_i\})$ からなる集合

$$\mathbf{up} = \{x_1 \mapsto \theta_1, x_2 \mapsto \theta_2, \ldots, x_n \mapsto \theta_n\}$$

を考える．このとき，更新式の集合 $\mathbf{up}$ を用いた割当 ν の更新を以下で定める：

$$\mathbf{up}(\nu) \triangleq \{\nu' : \forall i \in [1..n].\ \nu'(x_i) \in \theta_i(\nu)\}.$$

この定義は以下のように読むことができる：

- クロック x_i 上の更新式 θ_i は，現在の割当に基づく，新しい x_i の値の定め方を与える：$\nu'(x_i) \in \theta_i(\nu)$.
- 従って，新たな割当 ν' は，同時に全ての更新式を満たす必要がある．

例えば，次の更新式集合

$$\mathbf{up} = \{\big(\overline{x} < x \wedge fr(\overline{x}) = fr(x)\big),\quad \big(\overline{y} = y\big)\}$$

は，

- クロック x について整数部分を 1 以上減らし，
- クロック y は変更しない

ということを意味する．従って，以下のような更新が可能である：

$$\mathbf{up}(\{x \mapsto 2.4, y \mapsto 1.0\}) = \big\{\{x \mapsto 1.4,\ y \mapsto 1.0\},\ \{x \mapsto 0.4,\ y \mapsto 1.0\}\big\}.$$

2.3 遷移規則と計算列

UTA $\mathcal{A}$ の遷移規則は $p \xrightarrow{a,\phi,\mathbf{up}} q$ の形をとり，特に

- $p, q \in Q$ は状態で，$a \in \Sigma$ は入力文字を表し；
- $\phi \in \mathbb{C}(\mathcal{X})$ は遷移を許すかどうかを調べるための制約であり；
- $\mathbf{up}$ は更新式の集合で，各クロック $x \in \mathcal{X}$ について，丁度一つの制約式 $\mathbf{up}(x)$ を持つ．

UTA における遷移は，以下の形式で表される：

$$\langle p, \nu\rangle \xrightarrow{(\delta,a)} \langle q, \nu'\rangle.$$

これは，現時刻から $\delta \in \mathbb{Q}_{\geq 0}$ 時間経過した後に，入力記号 a を読んだ結果として，$\langle q, \nu'\rangle$ が得られることを意味する．従って，遷移 $\langle p, \nu\rangle \xrightarrow{(\delta,a)} \langle q, \nu'\rangle$ は，以下が満たされる時に成立する：

- 遷移規則 $p \xrightarrow{a,\phi,\mathbf{up}} q \in \Delta$ が存在し，
- 時間 $\delta \in \mathbb{Q}_{\geq 0}$ が経過後に，制約が満たされ $(\nu + \delta) \models \phi$,
- ν' が更新によって得られる $\nu' \in \mathbf{up}(\nu + \delta)$.

制約が満たされない場合だけでなく，更新が可能でない場合 $\mathbf{up}(\nu + \delta) = \emptyset$ にも，遷移が生じないことに注意されたい．

計算列 π は 1 つ以上の遷移の合成からなる：

$$\langle q_0, \nu_0\rangle \xrightarrow{(\delta_0,a_0)} \langle q_1, \nu_1\rangle \xrightarrow{(\delta_1,a_1)} \cdots \xrightarrow{(\delta_n,a_n)} \langle q_n, \nu_n\rangle.$$

計算列 π から，時間付き語 $\mathrm{tw}(\pi) \in (\Sigma \times \mathbb{Q}_{\geq 0})^*$ が定義される：

$$\mathrm{tw}(\pi) \triangleq (\delta_0, a_0)(\delta_1, a_1)\cdots(\delta_n, a_n).$$

特に，初期計算状況 $\langle q_{\mathrm{init}}, \mathbf{0}_{\mathcal{X}}\rangle$ から始まり，何らかの終了状態 $q_{\mathrm{final}} \in F$ で終わる計算列

$$\langle q_{\mathrm{init}}, \mathbf{0}_{\mathcal{X}}\rangle \xrightarrow{(\delta_0,a_0)} \cdots \xrightarrow{(\delta_n,a_n)} (q_{\mathrm{final}}, \nu)$$

を受理計算列と呼ぶ（終端点での割当 ν は何でも良い）．UTA $\mathcal{A}$ の言語を，受理計算列の定める時間付き語の全体からなる集合として定義する：

$$L(\mathcal{A}) \triangleq \{\mathrm{tw}(\pi) : \pi \text{ は } \mathcal{A} \text{ の受理計算列 }\}.$$

（注意．一般的な時間オートマトンの定義では，時間付き語を，イベントの発生時刻とのペアで定義する：

$$\mathrm{tw}(\pi) = (a_0, \delta_0)(a_1, \delta_0 + \delta_1)\cdots(a_n, \Sigma_{i=0}^{n}\delta_i).$$

ただし，これと本論文の定義では，本質的な差はない．本論文では，この後で言語例を考える時に，その説明を簡単にしたかったために上の定義を採用した．）

到達可能性問題. 時間オートマトンでの到達可能性問題の定義にならって，本論文での UTA $\mathcal{A}$ での到達可能性問題を定める．

$\mathcal{A}$ の 2 つの状態 q_{start} と q_{goal} について，q_{start} から q_{goal} への到達可能性問題とは，

$$\exists\nu.\ \langle q_{\mathrm{start}}, \mathbf{0}_{\mathcal{X}}\rangle \xrightarrow{(\delta_0,a_0)} \cdots \xrightarrow{(\delta_n,a_n)} \langle q_{\mathrm{goal}}, \nu\rangle$$

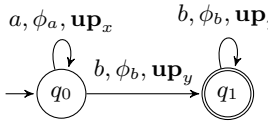
が成立するかどうかを判定する問題である．3 章以降で，到達可能性問題の決定可能性を示す．

2.4 UTA の言語の例

次の言語 L_{ab} を受理する UTA を考える：

$$L_{ab} \triangleq \{\overbrace{(1,a)(1,a)\dots(1,a)}^{n}\overbrace{(0,b)\dots(0,b)}^{m} : 0 < m \leq n\}.$$

この言語は丁度 1 秒毎に a が現れ，一方で b は時間経過なしに a の個数以下現れるような，時間付き語全体からなるものである．L_{ab} は 2 つのクロック x, y を持つ UTA で受理される：



ただし，遷移規則中の制約式や更新式は以下で定義されているとする：

$$\phi_a \equiv x = 1, \quad \phi_b \equiv x = 0,$$
$$\mathbf{up}_x \equiv \{\big(\overline{x} = 0\big),\ \big(\overline{y} = y\big)\},$$
$$\mathbf{up}_y \equiv \{\big(\overline{x} = x\big),\ \big(\overline{y} < y \wedge \mathit{fr}(\overline{y}) = \mathit{fr}(y)\big)\}.$$

例として，2 つの時間付き語 $(1,a)(1,a)(0,b)$ と $(1,a)(1,a)(0,b)(0,b)$ についての計算を考える．初期計算状況 $\langle q_0, \mathbf{0}_{\{x,y\}}\rangle$ で考えるべき遷移規則は，

$$q_0 \xrightarrow{a,\phi_a,\mathbf{up}_x} q_0$$

であり，この遷移規則は，丁度 1 秒経過した時に限り遷移を許し，その際に経過時間の計測を行うためのクロック x を 0 にリセットする．従って，

$$\langle q_0, \mathbf{0}_{\{x,y\}}\rangle \xrightarrow{(1,a)} \langle q_0, \{x \mapsto 0, y \mapsto 1\}\rangle$$

が成立し，

$$\langle q_0, \mathbf{0}_{\{x,y\}}\rangle \xrightarrow{(1,a)\,(1,a)} \langle q_0, \{x \mapsto 0, y \mapsto 2\}\rangle$$

までが成立する．

次の入力 $(0,b)$ で使用する遷移規則 $q_0 \xrightarrow{b,\phi_b,\mathbf{up}_y} q_1$ は，時間が一切経過していない場合に限り遷移を許し，その際に個数を記憶しているクロック y を「1 以上の整数値分」減らす操作を行う．よって

$$\langle q_1, \{x \mapsto 0, y \mapsto 2\}\rangle \xrightarrow{(0,b)} \langle q_1, \{x \mapsto 0, y \mapsto 0\}\rangle$$

が成立し，この時，結果として $(1,a)(1,a)(0,b)$ が受理される．また，

$$\langle q_1, \{x \mapsto 0, y \mapsto 2\}\rangle \xrightarrow{(0,b)} \langle q_1, \{x \mapsto 0, y \mapsto 1\}\rangle \xrightarrow{(0,b)} \langle q_1, \{x \mapsto 0, y \mapsto 0\}\rangle$$

も成立し，この時は $(1,a)(1,a)(0,b)(0,b)$ が受理される．

一方で，入力 $(1,a)(1,a)(0,b)(0,b)(0,b)$ を考える．$(1,a)(1,a)(0,b)(0,b)$ までを読んだとすると，計算状況は必ず $\langle q_2, \{x \mapsto 0, y \mapsto 0\}\rangle$ となる：

$$\langle q_0, \mathbf{0}_{\{x,y\}}\rangle \xrightarrow{(1,a)(1,a)(0,b)(0,b)} \langle q_1, \{x \mapsto 0, y \mapsto 0\}\rangle.$$

この時，例えば

$$\langle q_1, \{x \mapsto 0, y \mapsto 0\}\rangle \xrightarrow{(0,b)} \langle q_1, \{x \mapsto 0, y \mapsto -1\}\rangle$$

とできると良いが，クロック値の領域は「非負有理数」に限られているため，更新に失敗する：

$$\mathbf{up}_y(\{x \mapsto 0, y \mapsto 0\}) = \emptyset.$$

よって $(1,a)(1,a)(0,b)(0,b)(0,b)$ は受理されない．

上の言語 L_{ab} は，時間経過をせずに複数の b を処理するという意味で，実時間システムのモデル化という観点からは，少し不自然な例である．ここで，もう少し現実的な次の言語 L'_{ab} についても言及しておく：

$$L'_{ab} \triangleq \{\overbrace{(1,a)\dots(1,a)}^{n}\overbrace{(r_1,b)(r_2,b)\dots(r_m,b)}^{m} : 0 < m \leq n,\ (\Sigma_{i=1}^{m} r_i) = 1\}.$$

この言語は，1 秒毎に文字 a が現れ（その個数を n とする），続く「1 秒間」に b が n 個以下連続する文字列全体を表している．先と同様に，時間経過の計測のためのクロック x と，a の個数を記録するためのクロック y を用いて，同じアイディアによって UTA で受理することができる．

また，L_{ab} の別の方向の拡張として，使用するクロックの個数を増やすことで，次の言語も受理できる：

$$L_{abc} \triangleq \{\overbrace{(1,a)\dots(1,a)}^{n}\overbrace{(0,b)\dots(0,b)}^{m}\overbrace{(0,c)\dots(0,c)}^{l} : 0 < m \leq n,\ \ 0 < l \leq n\}.$$

L_{ab} は，a の個数 n について，これよりも少ない b の連続からなる時間付き語全体であった．L_{abc} は，更に別の文字 c について，これを拡張したものであり，3 つのクロックを持つ UTA で受理できる．

2.5 従来の体系との比較

我々の更新可能時間オートマトンの位置づけを示すために，Alur と Dill による時間オートマトン [4] と，Bouyer らによる更新可能時間オートマトン [7] との比較を行う．以下の表 1 が，簡単な要約である．

Alur と Dill によって [4] で導入された時間オートマトン（Timed automata）は, 更新可能時間オートマトンの制限として説明できる．正確には，更新式

表 1　時間オートマトンの比較

体系	制約式	更新式	到達可能性
TA [4]	$x \bowtie k$	$\overline{x} = x$, $\overline{x} = 0$	PSPACE 完全
Diag. [5] [4]	$x \bowtie k$, $x - y \bowtie k$	=TA	PSPACE 完全
Bouyer [7]	$x \bowtie k$, $x - y \bowtie k$	TA + $\overline{x} \leq k$	PSPACE 完全
	$x \bowtie k$, $x - y \bowtie k$	TA + $\overline{x} > k$	決定不能
	$x \bowtie k$	TA + $\overline{x} - y \bowtie k$	決定不能
本論文	$x \bowtie k, x \bowtie y$, $fr(x) = 0$, $fr(x) \bowtie fr(y)$	同左	Non.Prim.Rec

を，値を変更しない ($\overline{x} = x$) か，値 0 へのリセット ($\overline{x} = 0$) のみに制限し，制約式は，即値との比較（例えば $1 < x$ や $2 = y$）に制限することで得られる体系である．この体系では，到達可能性問題が，時間オートマトンの大きさに関して PSPACE 完全であることが証明されている [4].

先の言語 L_{ab} を受理する UTA の構成においては，更新操作が要となっていたため，従来の時間オートマトンで受理することは困難に思える．実際に，L_{ab} は従来の時間オートマトンでは受理できない．

補題 1. L_{ab}, L'_{ab}, 及び L_{abc} は時間オートマトンでは受理できない．

Proof. Alur と Dill による次の結果 [4] を用いる：

時間オートマトン $\mathcal{A}$ の言語 $L(\mathcal{A})$ に関して，
$Untime(L(\mathcal{A}))$ は正規言語である．

Untime は，時間付き語から，タイムスタンプを忘却するような関数であり，例えば，上の 3 つの言語について以下が成立する：

$$Untime(L_{ab}) = Untime(L'_{ab}) = \{\overbrace{aa\ldots a}^{n}\ \overbrace{bb\ldots b}^{m} : 0 < m \leq n\},$$

$$Untime(L_{abc}) = \{\overbrace{a\ldots a}^{n}\overbrace{b\ldots b}^{m}\overbrace{c\ldots c}^{l} : 0 < m \leq n,\ \ 0 < l \leq n\}.$$

従って，これら 2 つの言語が正規言語でないことを示せば，上の結果の対偶から，時間オートマトンでは受理できないことが示される．ところが，この 2 つの言語が正規言語でないことは，正規言語に関する反復補題から直ちに得られる．　□

次に，時間オートマトンに対角線制約（Diagonal constraints）と呼ばれる制約式を許した拡張を考える．Diagonal constraints とは $x - y \bowtie k$（ここで $k \in \mathbb{N}$）という形をした式であり，これを用いることで，2 つのクロックの値の差について調べることができる．通常の時間オートマトンについては，diagonal constraints を許す拡張でも，その表現力に変化がないことが良く知られている [5] [4].

最後に，Bouyer らの更新可能時間オートマトンをみる．Bouyer らは，更新可能時間オートマトンの「制約に使用可能な式」と「更新に使用可能な式」の 2 つをパラメータとし，(上の表のように）様々な組合せによって得られる体系の到達可能性問題の決定可能性及び，表現力を調べた [7]．更新可能時間オートマトンにおいては，一般の diagonal constraints（例えば $x - y = 1$）と，ある定数より大きな値への更新 $\overline{x} > k$ の組合せで，2 計数機械を十分に模倣できることを示した．そのようなチューリング完全な拡張を与える一方で，Bouyer らは制約式と更新式の組合せを強く制限することで，到達可能性問題が決定可能なクラスも与えている．しかし，そのようなクラスの表現力は，それ以前に知られていた「ϵ 遷移を許す」時間オートマトン [5] に含まれることも同時に示された．

実は上でみた言語 L_{ab} などは，補題 1 の証明と同様にして，ϵ 遷移を許す時間オートマトンでも受理できないことが示される．

補題 2. L_{ab}, L'_{ab}, 及び L_{abc} は ϵ 遷移を許した時間オートマトンでは受理できない．従って，Bouyer らの決定可能な部分体系でも受理されない．

加えて，我々の更新可能時間オートマトンで ϵ 遷移を許したとしても，本論で与える到達可能性解析アルゴリズムを適用できる．すなわち，我々の更新可能時間オートマトンは，その到達可能性問題が決定可能でありながら，表現力について Bouyer らの時間オートマトンの体系を超えたものであることが分かる．

我々の体系で L_{ab} といった言語を受理できる最大

の要因，もしくは Bouyer らの体系との決定的な違いは，Bouyer らの研究では扱われなかった，以下の 2 つの式に注目したことである：

- $k = 0$ の場合の $x - y \bowtie k$ の式，すなわち $x \bowtie y$ のみを許す．
- クロック値の小数部分を検査するための式 $fr(x) = 0$ や $fr(x) \bowtie fr(y)$ を許す．

実は，上の表で決定不能と書かれた部分では，本質的に $k > 0$ の場合の対角線式を用いて，2 計数機械の模倣を行っている．Bouyer らの論文では，$k = 0$ の場合に限った時についてどうなるかは，言及されていなかった．

また，$x \bowtie y$ の形式と，小数部分の検査のための式の，2 つを同時に採用することが重要である．本論ではその証明を与えることはしないが，どちらか 1 つだけの採用では，従来の時間オートマトンと表現力が変わらないことが示される．

表には現れていないが，additive constraints と呼ばれる，対角線制約とは異なる形式の制約と，更新の組合せについて論じた研究もある [13]．この研究では言語のクラスについての議論を行ってはいないが，Alur と Dill が示したものと同様に，*Untime* 関数の像が正規言語となるため，L_{ab} などを受理することはできない．

2.6 更新可能時間遷移系

本節では，UTA の到達可能性問題を解くために，より単純な計算モデルを考える．

更新可能時間遷移系（Updatable Timed Transition System，UTS）$\mathcal{S}$ は 3 つ組 $(Q, \mathcal{X}, \Delta)$ である．ここで Q と $\mathcal{X}$ は，それぞれ UTA と同じように，状態の有限集合とクロックの有限集合である．到達可能性問題を解くにあたり，入力アルファベットは必要でないため，UTS にはアルファベットはない．

UTS の計算状況 $\langle q, \nu \rangle$ は，状態 $q \in Q$ と割当 $\nu : \mathcal{X} \to \mathbb{Q}_{\geq 0}$ の組である．遷移規則 δ は，$(p, q, \textsc{Upd}(x)), (p, q, \textsc{Time}), (p, q, \textsc{Test}(\varphi)) \in \Delta$ の 3 種類がある．計算状況の集合 C について，記法 $\delta(C)$ により遷移を定義する：

- $\delta = (p, q, \textsc{Upd}(x))$ の場合：
  $\delta(C) \triangleq \{\langle q, \nu[x := r] \rangle : \langle p, \nu \rangle \in C, r \in \mathbb{Q}_{\geq 0}\}$.
- $\delta = (p, q, \textsc{Time})$ の場合：
  $\delta(C) \triangleq \{\langle q, \nu + \delta \rangle : \langle p, \nu \rangle \in C, \delta \in \mathbb{Q}_{\geq 0}\}$.
- $\delta = (p, q, \textsc{Test}(\varphi))$ の場合：
  $\delta(C) \triangleq \{\langle q, \nu \rangle : \langle p, \nu \rangle \in C, \nu \models \varphi\}$.

何らかの遷移規則 $\delta \in \Delta$ のもとで $\langle q, \nu' \rangle \in \delta(\{\langle p, \nu \rangle\})$ が成立するときに $\langle p, \nu \rangle \Rightarrow \langle q, \nu' \rangle$ と書く．複数のステップで遷移可能な場合を $\langle p, \nu \rangle \Rightarrow^* \langle q, \nu' \rangle$ と書く．

UTA の到達可能性問題を解くためには，UTS を考えれば十分であることが，次の補題から保証される．

補題 3. 与えられた UTA $\mathcal{A} = (Q, q_{\text{init}}, F, \mathcal{X}, \Sigma, \Delta)$ と 2 状態 q_{start} と q_{goal} について，以下を等価にする，UTS $\mathcal{S} = (Q', \mathcal{X}, \Delta')$（と 2 状態 $q'_{\text{start}}, q'_{\text{goal}}$）を構成することができる：

- $\mathcal{A}$ で，$\langle q_{\text{start}}, \mathbf{0}_{\mathcal{X}} \rangle$ から状態 q_{goal} へ到達できる．
- $\mathcal{S}$ で，$\langle q'_{\text{start}}, \mathbf{0}_{\mathcal{X}} \rangle$ から状態 q'_{goal} へ到達できる．

従って，UTS $\mathcal{S}$ の到達可能性問題が決定可能となることを示せば良いことが分かる．しかし，この証明は本質的でないところで複雑となってしまうので，証明のアイディアを述べるためにも，UTS を一層単純化した，更新可能離散時間遷移系（Updatable Discrete Timed Transition System，UDTS）を考え，この到達可能性判定アルゴリズムを先に与える．UDTS は，lossy counter machine と呼ばれる計算モデルを模倣することができ，それ自身で興味深い体系となっている．

3 更新可能離散時間遷移系

更新可能離散時間遷移系（Updatable Discrete Timed Transition System，UDTS）$\mathcal{D}$ は，UTS と同じく 3 つ組 $(Q, \mathcal{X}, \Delta)$ で，Q は有限の状態集合，$\mathcal{X}$ はクロックの有限集合，Δ は遷移規則の有限集合である．

UDTS の計算状況 $\langle q, \mu \rangle$ は，状態 $q \in Q$ と，自然数上の割当 $\mu : \mathcal{X} \to \mathbb{N}$ である．自然数上の割当に限定するところが，UTS とは異なる．遷移規則 δ は，$(p, q, \textsc{Upd}(x)), (p, q, \textsc{Time}), (p, q, \textsc{Test}(\varphi)) \in \Delta$ の 3 種類がある．計算状況の集合 C について，記法 $\delta(C)$ により遷移を定義する：

- $\delta = (p, q, \textsc{Upd}(x))$ の場合：
$$\delta(C) \triangleq \{\langle q, \mu[x := n]\rangle : \langle p, \mu\rangle \in C, n \in \mathbb{N}\}.$$
- $\delta = (p, q, \textsc{Time})$ の場合：
$$\delta(C) \triangleq \{\langle q, \mu + 1\rangle : \langle p, \mu\rangle \in C\}.$$
- $\delta = (p, q, \textsc{Test}(\varphi))$ の場合：
$$\delta(C) \triangleq \{\langle q, \mu\rangle : \langle p, \mu\rangle \in C, \mu \models \varphi\}.$$

$\textsc{Test}(\varphi)$ では，その時点での割当を検査するが，UDTS では自然数上の割当のみを考えているので，φ は以下の文法で与えられると考えても良い：
$$\varphi ::= x \bowtie k \mid x \bowtie y \mid \neg\phi \mid \phi \wedge \phi.$$
$\mathcal{D}$ は構成要素が有限の計算モデルであるから，制約式に現れる最大の自然数 M が定まる：
$$\mathsf{M} \triangleq \max\{k : \text{“}x \bowtie k\text{” が } \Delta \text{ 中に現れる }\}.$$
この M は次節以降で用いられるが，直感的に言えば，Δ 中の制約式では，クロック値が M を超えた場合に具体的な値を調べられないことを意味する．

本章では，次の定理を示す．

定理. UDTS $\mathcal{D}$ と 2 つの位置 $q_{\text{init}}, q_{\text{final}}$ について，到達可能性問題 $\langle q_{\text{init}}, \mathbf{0}_{\mathcal{X}}\rangle \Rightarrow^* \langle q_{\text{final}}, \mu\rangle$ は決定可能である（終端点の μ は何でも良いとする）．

我々の到達可能性判定アルゴリズムは，状態 q_{final} からの後ろ向き到達可能性解析を行うもので，概略的には次のようにまとめられる：

- 状態 q_{final} にさえ到達することができれば良いので，集合 $R_0 = \{\langle q_{\text{final}}, \mu\rangle : \mu \in \mathcal{X} \to \mathbb{N}\}$ に $\langle q_{\text{init}}, \mathbf{0}_{\mathcal{X}}\rangle$ が入っているかどうかを調べる．
- もし入っていないのであれば，1-step の計算で集合 R_0 に入る計算状況全体の集合 $R_1 = pre(R_0) = \{c : c \Rightarrow c', c' \in R_0\}$ を計算し，$\langle q_{\text{init}}, \mathbf{0}_{\mathcal{X}}\rangle \in R_1$ かどうかを調べる．
- またしても入っていないのであれば，2-step の計算で集合 R_0 に入る計算状況全体の集合 $R_2 = pre(R_1)$ を計算し，$\langle q_{\text{init}}, \mathbf{0}_{\mathcal{X}}\rangle \in R_2$ かどうかを調べる．

以上の操作の繰り返しが判定アルゴリズムの骨子である．このアルゴリズムには，$\langle q_{\text{init}}, \mathbf{0}_{\mathcal{X}}\rangle$ から q_{final} へ到達できない場合に，手続きが停止しないという問題点がある．しかし，更新可能離散時間遷移系（及び更新可能時間遷移系）では，後ろ向き到達可能性解析について，停止性を保証する良い性質が現れる．次節以降でこのことをみていく．

3.1 計算状況集合の記号表現

後ろ向き到達可能性解析では，各 step において，計算状況集合 R_0 へ到達しうる計算状況の集合を拡大する操作を行っている．そのため，計算状況の「集合」を記号的に表現し，個々の要素ではなく記号表現を操作したい．そこで，本節では，以下の性質を満たす表現形式を探す．

望まれる性質. E を計算状況の表現，$[\![E]\!]$ を E の表す計算状況全体の集合，$pre([\![E]\!])$ を 1-step の遷移で $[\![E]\!]$ に入る計算状況全体とする．この時，E から次を満たす有限個の（空かもしれない）表現の集合 $\{E_1, E_2, \ldots, E_n\}$ が計算可能である：
$$pre([\![E]\!]) = \bigcup_i [\![E_i]\!].$$

まず，制約式で区別不能な計算状況をひとまとめにする，ということが考えられる．すなわち，ある表現 E の要素 $\langle q, \mu_1\rangle, \langle q, \mu_2\rangle \in [\![E]\!]$ について，UDTS $\mathcal{D}$ の全ての制約式 φ に関して
$$\mu_1 \models \varphi \iff \mu_2 \models \varphi$$
が成立するべきである．$\mathcal{D}$ において，既に制約式中の最大の定数 M を定義した．その際にも述べたが，$\mathcal{D}$ では M を超えた値を $x \bowtie k$ の形の式で調べることはできない．従って，M を超えたクロック値については，正確な値を保持しておく必要はない．その一方で，式 $x \bowtie y$ によって，2 つのクロックが等しいか，もしくは大小の関係などは調べることはできる．そこで，M を超えた部分では順序関係だけを覚える，ということを念頭に次の *preshape* と呼ぶ表現を考える：
$$P : \wr 0, x_1, x_2 \wr 1 \wr (x_3, x_4 \wr (x_5 \wr.$$
以降では（$\mathcal{D}$ から計算した）M が 1 であるとし，2 以上の自然数値を含まない preshape だけを考える．

Preshape では，あるバッグ $\wr \cdots \wr$ で，ひとまとめにされるクロック（と M 以下の定数）が全て等しいとし，同時にバッグのならびが，クロックの大小関係を表すものとする．したがって，この preshape P

は，以下の割当の集合を表す：

$$[\![P]\!] = \{\mu : \{x_1, \ldots x_5\} \to \mathbb{N} : \\ \mu \models 0 = x_1 = x_2,\ \ \mu \models x_3 = x_4 \\ \mu \models 1 < x_3,\ \ \mu \models x_3 < x_5\}.$$

Preshape は割当集合のための記号表現だが，状態 q と組合せて，$[\![\langle q, P\rangle]\!]$ で「計算状況」の集合を与える：

$$[\![\langle q, P\rangle]\!] \triangleq \{\langle q, \mu\rangle : \mu \in [\![P]\!]\}.$$

Preshape の利点は，次の命題にある．

命題 1. P を preshape とする．このとき，Δ に出現する任意の制約式 φ について，以下が成立する：

$$\forall \mu_1, \mu_2 \in [\![P]\!].\ \mu_1 \models \varphi \iff \mu_2 \models \varphi.$$

同時に，規則 $\delta = (p, q, \textsc{Test}(\varphi)) \in \Delta$ について，以下を満たす（空かもしれない）有限集合 $\{P_1, \ldots, P_n\}$ が計算可能：

$$\delta^{-1}([\![\langle q, P\rangle]\!]) = \bigcup_i [\![\langle p, P_i\rangle]\!].$$

命題 2. Preshape P と，規則 $\delta = (p, q, \textsc{Time}) \in \Delta$ について，以下を満たす（空かもしれない）有限集合 $\{P_1, \ldots, P_n\}$ が計算可能：

$$\delta^{-1}([\![\langle q, P\rangle]\!]) = \bigcup_i [\![\langle p, P_i\rangle]\!].$$

すなわち，preshape は $\textsc{Test}(\varphi)$ や $\textsc{Time}$ 遷移規則についての *pre* の計算を正確に表現できることを意味する．一方で，preshape では，M を超えた部分のクロック間の「距離」に相当する情報を失っているため，$\textsc{Upd}(x)$ についての *pre* に相当する計算が正確に表現できない．これは以下で確かめられる．

1. 規則 $(p, q, \textsc{Upd}(x))$ によって $[\![\langle q, \wr 0 \wr\!\wr 1 \wr\!\wr x \wr\!\wr y \wr\rangle]\!]$ となる計算状況全体を考える．
   $$[\![\wr 0 \wr\!\wr 1 \wr\!\wr x \wr\!\wr y \wr]\!] = \{\mu : \mu \models 1 < x < y\}$$
   であるから，そのような計算状況全体は
   $$\{\langle p, \mu\rangle : \mu \models 2 < y\}$$
   であることが分かる．
2. すなわち，y が「3 以上」の値を取る全てを表現する必要がある．しかし，preshape ではバッグ間の距離を表現することができないため，これを正確に表現することができない．
3. 例えば，x と y が同じ値を持つ計算状況の集合は，$\wr 0 \wr\!\wr 1 \wr\!\wr x, y \wr$ でしか表現できないが，
   $$[\![\wr 0 \wr\!\wr 1 \wr\!\wr x, y \wr]\!] = \{\mu : \mu \models x = y \wedge 1 < x\}$$
   となってしまい，$\{x \mapsto 2, y \mapsto 2\}$ などの，望まない割当が紛れる．

この考察に基づき，preshape を，M を超えた部分についてバッグ間の距離情報を持つように細分化した，*shape* と呼ぶものを考える．各 shape S は，以下の形式で書かれる：

$$S : \wr 0, x_0 \wr\!\wr 1, x_1 \wr <_a \wr x_2, x_3 \wr <_b \wr x_4 \wr.$$

M を超えた部分について，直前のバッグとどれだけ離れているかを，index $a, b \in \mathbb{N}$ によって表現する．この時 S は，以下の割当の集合を表現する：

$$[\![S]\!] = \{\mu : \mu \models 0 = x_0 \wedge 1 = x_1 \wedge x_2 = x_3, \\ \mu \models 1 + a < x_2,\ \mu \models x_2 + b < x_4\}.$$

任意の preshape は，全ての index を 0 とする shape で表すことができる．

Preshape の代わりに shape を用いると，先の状況を回避できることを確認する：

1. 再び $\textsc{Upd}(x)$ によって $\wr 0 \wr\!\wr 1 \wr <_0 \wr x \wr <_0 \wr y \wr$ となる計算状況全体を考える．
   $[\![\wr 0 \wr\!\wr 1 \wr <_0 \wr x \wr <_0 \wr y \wr]\!] = \{\mu : \mu \models 1 < x < y\}$ であるから，以下の計算状況の集合
   $$\{\mu \in \{x, y\} \to \mathbb{N} : \mu \models 2 < y\}$$
   を，複数の shape で表現できればよい．
2. shape を用いると，
   $$\{\mu : \mu \models 2 < y\} = [\![S_1]\!] \cup [\![S_2]\!] \cup \cdots \cup [\![S_5]\!]$$
   で表現できる．ただし，S_1 から S_5 は以下で定義する：
   $$S_1 = \wr 0, x \wr\!\wr 1 \wr <_1 \wr y \wr, \quad S_2 = \wr 0 \wr\!\wr 1, x \wr <_1 \wr y \wr,$$
   $$S_3 = \wr 0 \wr\!\wr 1 \wr <_0 \wr x \wr <_0 \wr y \wr,$$
   $$S_4 = \wr 0 \wr\!\wr 1 \wr <_1 \wr x, y \wr,$$
   $$S_5 = \wr 0 \wr\!\wr 1 \wr <_1 \wr y \wr <_0 \wr x \wr.$$

Shape は preshape を細分化したものであるため，既に preshape で成立していた $\textsc{Test}(\varphi)$ や $\textsc{Time}$ に関する性質は保存され，加えて細分化によって，$\textsc{Upd}(x)$ についての一歩前の計算状況全体も正確に表現できるようになる．その結果として，我々の望む性質が次の補題で実現される．

補題 4. 状態と shape の任意の組 $\langle q, S\rangle$ について，以下を満たす（空かもしれない）有限の shape の集合 $\{\langle p_1, S_1\rangle, \ldots, \langle p_n, S_n\rangle\}$ を計算できる：

$$pre([\![\langle q, S\rangle]\!]) = \bigcup_i [\![\langle p_i, S_i\rangle]\!].$$

特に $\textsc{Pre}(\langle q, S\rangle) = \{\langle p_1, S_1\rangle, \ldots, \langle p_n, S_n\rangle\}$ と書くことにする．

Preshape に代わって shape を採用することで上の補題を得るが，その一方で別の問題として，shape 全体の集合が，index のために加算無限程度のサイズとなってしまう．しかし，shape 上に *well-quasi-ordering* を定めることができ，これによって後ろ向き到達可能性解析アルゴリズムが得られることを次節でみる．

3.2 Shape 上の Well-quasi-ordering

2 つの shape S_1 と S_2 について，index を忘れると同じ preshape になる場合に，$S_1 \sim S_2$ と書く．すなわち以下のような S_1 と S_2 については，$S_1 \sim S_2$ が成立する：

$$S_1 = \wr 0, \cdots \wr 1, \cdots \wr <_{a_1} \wr \cdots \wr <_{a_2} \wr \cdots \wr,$$
$$S_2 = \wr 0, \cdots \wr 1, \cdots \wr <_{b_1} \wr \cdots \wr <_{b_2} \wr \cdots \wr.$$

$S_1 \sim S_2$ について $\langle a_1, a_2, \ldots, a_n \rangle$ と $\langle b_1, b_2, \ldots, b_n \rangle$ をそれぞれ S_1 と S_2 の index とする．このとき，すべての $i \in [1..n]$ について $a_i \leq b_i$ が成立しているならば，もしくは同じことであるが $\mathbb{N}^n$ 上の直積順序で $\langle a_1, a_2, \ldots, a_n \rangle \leq \langle b_1, b_2, \ldots, b_n \rangle$ ならば，$S_1 \preccurlyeq S_2$ と書く．

このとき，$\preccurlyeq$ が shape 上の半順序（partial order）であることは明らかである．また，バッグ間の距離 $\wr x, \cdots \wr <_d \wr y, \cdots \wr$ は，$x + d < y$ を要求することを思い出すと，shape 上の順序 $S_1 \preccurlyeq S_2$ では，S_1 でのバッグ間の距離 a_i が，S_2 での距離 b_i 以下 $a_i \leq b_i$ であるから，割当の集合としては S_1 の方が大きいことを意味する．

命題 3. 2 つの shape S_1 と S_2 について，$S_1 \preccurlyeq S_2$ ならば，$[\![S_2]\!] \subseteq [\![S_1]\!]$ が成立する．

特に，shape 上の半順序 $\preccurlyeq$ は，*well-quasi-ordering* [12] [2] [10] と呼ばれるものになっている．これは，次節で与える，後ろ向き解析アルゴリズムの停止性を示す上で鍵となる性質であり，正確には次で定義される．半順序集合 $(X, \preceq)$ で $\preceq$ が well-quasi-ordering（w.q.o.）になるのは，次を満たすとき：

- X 上の任意の無限列 $(x_i)_i = x_1, x_2, x_3, \ldots$ について，index の組 (m, n) で $m < n$ かつ $x_m \preceq x_n$ を満たすものが存在する．

Well-quasi-ordering の代表的な例は，自然数上の標準的な順序 $(\mathbb{N}, \leq)$ と，次元 d の自然数ベクトル上の直積順序 $(\mathbb{N}^d, \leq)$ がある．我々の shape 上の順序 $\preccurlyeq$ は，自然数ベクトル上の直積順序を元にして導入したものであり，これから次の補題が示される．

補題 5. Shape 上の $\preccurlyeq$ は well-quasi-ordering である．

また，2 つの w.q.o. から作った直積順序が w.q.o. を満たすという Dickson の補題 [12] により，状態と shape 上の次の直積順序が w.q.o になることも分かる．

補題 6. 状態と shape の組 $\langle q, S \rangle$ と $\langle q', S' \rangle$ について，以下の順序 $\preccurlyeq$ は w.q.o. になる：

$$\langle q, S \rangle \preccurlyeq \langle q', S' \rangle \iff q = q' \text{ かつ } S \preccurlyeq S'.$$

この w.q.o を用いて，後ろ向き到達可能性解析アルゴリズムを与えた上で，その停止性を証明する．

3.3 後ろ向き到達可能性アルゴリズム

3 章の冒頭で述べた，後ろ向き到達可能性解析アルゴリズムを，補題 4 を用いて shape 上で形式化していく．

まず，$R_0 = \{\langle q_{\text{final}}, \mu \rangle : \mu \in \mathcal{X} \to \mathbb{N}\}$ について

$$R_0 = [\![\langle q_{\text{final}}, I_1 \rangle]\!] \cup \cdots \cup [\![\langle q_{\text{final}}, I_n \rangle]\!]$$

を特徴付ける有限個の shape $\{I_1, \ldots, I_n\}$ を計算しておく．

次に，現時点で到達すると分かっている計算状況の集合 X から，$X \cup pre(X)$ を求める計算に相当する関数 Expand を定義する．

```
1: function Expand({C_1, C_2, ..., C_n})
2:     R := {C_1, C_2, ..., C_n}
3:     for i = 1 to n do
4:         for each D ∈ Pre(C_i) do
5:             if R は C ≼ D とする C を持たない then
6:                 R := R ∪ {D}
7:     return R
```

この関数は，既に q_{final} に到達することが分かっている計算状況の集合 $\{C_1, C_2, \ldots, C_n\}$ を，補題 4 を用いて，1 ステップ分拡大する．2 つの shape S_1, S_2 が $S_1 \preccurlyeq S_2$ であるときに，$[\![S_2]\!] \subseteq [\![S_1]\!]$ が成立することは既にみた通りであるから，既に到達すると分かっている $C \in \mathcal{R}$ よりも，$\preccurlyeq$ として大きな D を新たに含めることは「無駄」であるために，これを省いている

ことに注意されたい.

このとき，Expand は明らかに単調増加関数であるから，

$$\begin{aligned}
&\textsc{Expand}(\{\langle q_{\text{final}}, I_1\rangle, \ldots, \langle q_{\text{final}}, I_n\rangle\}) &&\subsetneq\\
&\textsc{Expand}^2(\{\langle q_{\text{final}}, I_1\rangle, \ldots, \langle q_{\text{final}}, I_n\rangle\}) &&\subsetneq\\
&\qquad\vdots\\
&\textsc{Expand}^k(\{\langle q_{\text{final}}, I_1\rangle, \ldots, \langle q_{\text{final}}, I_n\rangle\}) &&=\\
&\textsc{Expand}^{k+1}(\{\langle q_{\text{final}}, I_1\rangle, \ldots, \langle q_{\text{final}}, I_n\rangle\})
\end{aligned}$$

のような不動点の存在を保証することができれば，次の補題により，$\langle q_{\text{init}}, \mathbf{0}_{\mathcal{X}}\rangle$ を含む記号表現が含まれているかどうかを調べることで，到達可能性が解ける．

補題 7. 次の 2 つは同値である：

- ある μ で $\langle q_{\text{init}}, \mathbf{0}_{\mathcal{X}}\rangle \Rightarrow^* \langle q_{\text{final}}, \mu\rangle$ が成立する．
- 不動点 $\textsc{Expand}^k(\{\langle q_{\text{final}}, I_1\rangle, \ldots, \langle q_{\text{final}}, I_n\rangle\}) = \{C_1, \ldots, C_m\}$ について，$\langle q_{\text{init}}, \mathbf{0}_{\mathcal{X}}\rangle \in [\![C_i]\!]$ とする C_i が存在する．

実は，Expand のような「無駄のない」拡大について，かならず不動点が存在することが，well-quasi-ordering の定義から示される．

補題 8. $(X, \preceq)$ を well-quasi-ordering 付きの集合とする．X の部分集合 Y について，無駄のない拡大を次で定義する：

$$Y \Subset Y \cup \{x\} \iff \forall y \in Y.\neg(y \preceq x).$$

このとき，以下のような

$$Y_0 \Subset Y_1 \Subset Y_2 \Subset \cdots$$

無駄のない拡大を行う無限列は存在しない．

Proof. 背理法で示す．任意の $i \geq 0$ について，$y_{i+1} \in Y_{i+1} \setminus Y_i$ を選ぶことができ，これを用いて無限列 $y_1, y_2, \ldots, y_m, \ldots, y_n, \ldots$ を作る．$\preceq$ は w.q.o なので，この列の中に $m < n$ で $y_m \preceq y_n$ とするものが存在しなくてはならない．しかし，y_n が追加される理由は，$y \preceq y_n$ となる y が存在しないことに他ならない筈なので，矛盾する． □

逆に，無駄のない拡大での無限列が存在しなければ，$(X, \preceq)$ が w.q.o になることも導ける．

ここまでの議論をまとめると，次の結果が得られる．

定理 1. 更新可能離散時間遷移系 $\mathcal{D}$ で，到達可能性問題は決定可能である．

3.4 Well Structured Transition System との関連

Well-quasi-ordering の導入から補題 8 が得られ，到達可能性解析アルゴリズムの停止性が保証された．

この形での停止性保証を行う議論の流れそのものは，既に存在しており，本論で新たに与えたものではない．具体的には，近年精力的に研究が行われている *Well structured transition system (WSTS)* [2] [10] と呼ばれる体系で現れている．詳細については引用した文献を参考にしてもらうことにして，WSTS でも，本論で考えるような初期計算状況から状態への到達可能性問題（これは [2] [10] では coverability problem とも呼ばれている）のための，後ろ向き到達可能性解析アルゴリズムの停止性が，w.q.o によって保証される．そこで考える後ろ向き到達可能性解析アルゴリズムは，本論で与えた，既知の集合に対する無駄のない拡大と本質的に同じことを行う．ただし，我々の更新可能離散時間遷移系そのものは，直接 WSTS になるわけではないことも述べておく．5 章では，より具体的に，WSTS の一例である Lossy counter machine を用いることで，更新可能離散時間遷移系の到達可能性問題の計算量について言及する．

4 更新可能時間遷移系の到達可能性解析

3 章の $\mathbb{N}$ を領域とする更新可能離散時間遷移系のための到達可能性解析アルゴリズムを踏まえ，$\mathbb{Q}_{\geq 0}$ を領域とする更新可能時間遷移系のための到達可能性解析アルゴリズムを与える．

前節で考えた shape は，自然数上の割当のなす集合を記号表現したものであり，特にこの上の well-quasi-ordering が，解析アルゴリズムの停止性のために重要であった．本節では，まず shape を，非負有理数上の割当のなす集合を表現するために設計し直す．この目的のために，時間オートマトンの理論で有用な道具である，region abstraction の手法を用いる．

4.1 Region abstraction

抽象化手法である region abstraction は，時間オートマトンの到達可能性問題の決定可能性を示すために Alur と Dill によって導入された [4].

従来の region abstraction では，与えられた割当 ν について，小数部分の実際の値を捨て，代わりにクロックの小数部分の「ならび」だけを覚える．例えば，$\nu = \{v \mapsto 1.2, w \mapsto 0.5, x \mapsto 0.2, y \mapsto 1.0, z \mapsto 1.6\}$ という割当は，以下のように抽象化される：

$$\mathcal{R}(\nu) = \{(y,2)\}_0\{(x,1),(v,2)\}\{(w,1)\}\{(z,2)\}.$$

$\mathcal{R}(\nu)$ におけるクロックの列びは，ν の小数部分に対応し，整数部分の値だけが残されていることに注意されたい．特に，一番左の集合 $\{\ldots\}_0$ によって，小数部分が丁度 0 のクロックを表すようにする．この記法は，どちらかといえば [1] におけるものを基にしているが，Alur と Dill が考えた region abstraction と本質的な差はない．

制約式の，特に小数部分を調べる原子式は $fr(x) = 0$ ないし $fr(x) \bowtie fr(y)$ であり，この範囲を調べるだけであれば上のような抽象化で十分であることは分かる．実際，次の命題が成立する．

命題 4. 2 つの割当 ν_1, ν_2 が $\mathcal{R}(\nu_1) = \mathcal{R}(\nu_2)$ を満たすとき，任意の制約式 φ について以下が成立する：

$$\nu_1 \models \varphi \iff \nu_2 \models \varphi.$$

我々の目的のために，$\mathcal{R}(\nu)$ を，shape と小数部のならびの組で表現する．これを Rshape と呼ぶことにし，各 Rshape $R = (S, F)$ は次のような形をしている：

$$S = \wr 0 \wr\!\wr x \wr\!\wr w \wr\!\wr 1, y \wr <_a \wr v \wr <_b \wr z \wr,$$
$$F = \{y\}_0\{x,v\}\{w\}\{z\}.$$

（ここでは，$\mathsf{M} = 1$ としていることに注意.）

有理数上の値の抽象化を行うので，自然数上の値の抽象化とは異なり，$\wr 0 \wr\!\wr x \wr\!\wr w \wr\!\wr 1, y \wr$ といったことが起こりうることに注意されたい．

Rshape (S, F) は，クロック値の情報を先と同じように shape S で表現し，更に小数部分を調べる制約式に対応するために，小数部分の情報を別に持っているにすぎない．従って，Rshape $R = (S, F)$ の表す計算状況全体は，$[\![R]\!] = [\![S]\!] \cap [\![F]\!]$ で与えられ，この例については $[\![S]\!]$ と $[\![F]\!]$ は以下で定義される：

$$[\![S]\!] = \{\nu : \nu \models y = 1,\ \ \nu \models 0 < x < w < 1,\ \ \nu \models (1+a) < v,\ \ \nu \models (v+b) < z\},$$
$$[\![F]\!] = \{\nu : \nu \models fr(y) = 0, \nu \models fr(x) = fr(v),\ \ \nu \models fr(y) < fr(x) < fr(w) < fr(z)\}.$$

このとき，Rshape は，後ろ向きの計算について正確であり，補題 4 に対応する性質が成立する．

補題 9. 状態と Rshape の任意の組 $\langle q, R\rangle$ について，以下を満たす（空かもしれない）有限の集合 $\{\langle p_1, R_1\rangle, \ldots, \langle p_n, R_n\rangle\}$ を計算できる：

$$pre([\![\langle q, R\rangle]\!]) = \bigcup_i [\![\langle p_i, R_i\rangle]\!].$$

4.2 RShape 上の Well-Quasi-Ordering

2 つの Rshape $R_1 = (S_1, F_1)$ と $R_2 = (S_2, F_2)$ について，以下を満たす時に，$R_1 \preccurlyeq R_2$ と書く：

- S_1 と S_2 が index を忘れると一致し；
- S_1 と S_2 の index 列をそれぞれ $\langle a_1, a_2, \ldots, a_n\rangle$, $\langle b_1, b_2, \ldots, b_n\rangle$ とすると，$\forall i \in [1..n].\, a_i \leq b_i$ が成立し，
- $F_1 = F_2$ が成立する．

このとき，$R_1 \preccurlyeq R_2$ ならば $[\![R_2]\!] \subseteq [\![R_1]\!]$ が成立し，かつ well-quasi-ordering となる．

従って，前節で考えたものと同じ到達可能性解析アルゴリズムによって，更新可能時間遷移系の到達可能性問題を解くことができる．

定理 2. 更新可能時間遷移系 $\mathcal{S}$ の到達可能性問題は決定可能である．

この定理と，補題 3 によって，主定理が得られる．

定理 3. 更新可能時間オートマトン $\mathcal{A}$ の到達可能性問題は決定可能である．

5 到達可能性問題の計算量について

更新可能時間オートマトンの到達可能性問題が決定可能であることを示したので，ここでは，その計算量についての考察を行う．

補題 3 として，更新可能時間オートマトンの到達可能性問題が，更新可能時間遷移系の対応する問題に還元されることをみた．実は，この逆も成立する．

補題 10. 与えられた UTS $\mathcal{S} = (Q, \mathcal{X}, \Delta)$ と 2 状態 q_{start} と q_{goal} について，以下を等価にする，UTA $\mathcal{A} = (Q', q_{\mathrm{init}}, q_{\mathrm{final}}, \mathcal{X}', \Sigma, \Delta')$ を構成できる：

- $\mathcal{S}$ で，$\langle q_{\mathrm{start}}, \mathbf{0}_{\mathcal{X}}\rangle$ から状態 q_{goal} へ到達できる．
- $\mathcal{A}$ で，$\langle q_{\mathrm{init}}, \mathbf{0}_{\mathcal{X}'}\rangle$ から状態 q_{final} へ到達できる．

従って，ここでは，本論文で与えた 3 つの体系のうちで，最も簡単な「更新可能離散時間遷移系」におけ

る，到達可能性問題の時間計算量について考えることにする．更新可能離散時間遷移系における到達可能性解析アルゴリズムの停止性は，well-quasi-ordering 上での無駄のない拡大が必ず停止することを保証する補題 8 によるものであった．しかし，この補題はいつか停止するということだけを保証し，どの程度のステップを要するかについては保証していないため，これを元にした計算量の解析は困難である．

そこで我々は，*Lossy counter machine (LCM)* と呼ばれる体系の到達可能性問題について既に知られている，計算量の下界に関する結果を用いる．LCM は，計数機械の定義をもとにして得られる計算モデルで，一般には複数のカウンタを考える．よく知られているように，計数機械では 2 個以上のカウンタを持つ場合にはチューリング完全となるが，LCM では「値を loss する」ために，カウンタが幾つあったとしても，到達可能性問題が決定可能になることが知られている．体系の正確な定義や到達可能性問題の定義については，[14] [16] などを参考にしてもらうことにして，ここでは [17] で示された次の定理を用いる．

定理 (LCM の時間計算量の下界 [17])**.** LCM の到達可能性問題は，LCM のサイズを n とした時に，決定性チューリング機械で $F(n)$ 時間必要となる．ただし，ここで F とは,「原始帰納的ではない」ような全域帰納的関数である．

また，本論では LCM の定義を与えないために正確な議論を述べられないが，以下が成立する．

補題 11. 任意の LCM から，そのサイズに関して線形時間で，これを模倣する更新可能離散時間遷移系を構成することができる．すなわち，LCM の到達可能性問題は，線形時間で，更新可能離散時間遷移系の到達可能性問題に帰着可能である．

従って，これらを組み合わせると，時間計算量の下界について，次の定理が得られる．

定理 4. 更新可能離散時間遷移系のサイズを n とした時に，到達可能性問題を解くには，決定性チューリング機械で $F(n)$ 時間必要である．上と同様に，関数 F は，原始帰納的でない全域帰納的関数のクラスに属する．

時間オートマトンの到達可能性問題の計算量は，PSPACE 完全 [4] であることが知られており Bouyer らの更新可能時間オートマトンで決定可能なクラスについても，計算量は PSPACE 完全 [7] [8] であることが分かっている．従って，我々の更新可能時間オートマトンは，元になった体系に比べて，その到達可能性問題が非常に難しくなっていることが分かる．

6 まとめと今後の課題

本論では，Alur と Dill による時間オートマトンに対して，更新の概念を導入した Bouyer らの更新可能時間オートマトンという体系に関する，新たな拡張を提案した．特に我々の更新可能時間オートマトンは，時間オートマトンの言語クラスを拡大するものであることを，具体的な言語 L_{ab} を分析して確認することができた．また，そのような表現力の拡張を行うにも関わらず，その到達可能性問題は決定可能であるということを，well-quasi-ordering という順序を見出すことによって証明した．しかしながらその計算量は，lossy counter machine の到達可能性問題からの還元により，全域帰納関数を用いなければならない程度に巨大であることが示された．

今後の理論的な課題として，以下のことを考えている．

一つに，更新可能離散時間遷移系の到達可能性問題の計算量上界を調べるということがある．実は，lossy counter machine の到達可能性問題は，Class of Ackermannian problems Ack と呼ばれるクラスに属することが判明している [15]．そこで，これと同じ証明手法で，我々の体系も Ack に属するかどうかを分析したい．

一つに，Abdulla らによって導入された Dynamic lossy channel system (DLCS) と呼ばれる体系 [3] との比較を行うことがある．DLCS は，Timed lossy channel system と呼ばれる体系の到達可能性問題の決定可能性証明のために用いられ，DLCS 自身の到達可能性問題の決定可能性も，well-quasi-ordering を用いたものとなっている [3]．この体系と我々の更新可能時間遷移系には，類似した点が見受けられる．例えば，DLCS には，lossy channel system（や lossy counter machine）にはない，チャンネルそのものを

データとして取り扱う機能が備わっている．一方で，我々の体系でも，更新を用いることでクロックの値を失うことなくコピーすることが可能であった．そこで，これらの体系を分析することで，より一般的な体系を与えることに取り組みたい．

一つに，我々の更新可能時間オートマトンを，スタックを伴う形で拡張するということがある．最近になって，時間オートマトンとプッシュダウンオートマトンの考えを組み合わせたもので，その到達可能性問題が決定可能な体系が幾つか提案されている [1] [9]．しかしこれらの体系では，本論でみたような更新操作を考慮していない．そこで，我々の更新可能時間オートマトンをこの方向で拡張した時に，到達可能性問題が依然として決定可能であるかどうかを分析したい．

参 考 文 献

[1] Abdulla, P., Atig, M., and Stenman, J.: Dense-Timed Pushdown Automata, *LICS '12*, IEEE, 2012, pp. 35–44.

[2] Abdulla, P., Čerāns, K., Jonsson, B., and Tsay, Y.: Algorithmic Analysis of Programs with Well Quasi-ordered Domains, *Inf. & Comp.*, Vol. 160, No. 1–2(2000), pp. 109–127.

[3] Abdulla, P. A., Atig, M. F., and Cederberg, J.: Timed Lossy Channel Systems, *FSTTCS'12*, LIPIcs, 2012, pp. 374–386.

[4] Alur, R. and Dill, D.: A theory of timed automata, *TCS*, Vol. 126, No. 2(1994), pp. 183–235.

[5] Bérard, B., Diekert, V., Gastin, P., and Petit, A.: Characterization of the Expressive Power of Silent Transitions in Timed Automata, *Fundam. Inf.*, Vol. 36, No. 2(1998), pp. 145–182.

[6] Bouyer, P., Dufourd, C., Fleury, E., and Petit, A.: Expressiveness of Updatable Timed Automata, *MFCS'00*, LNCS, Vol. 1893, Springer, 2000, pp. 232–242.

[7] Bouyer, P., Dufourd, C., Fleury, E., and Petit, A.: Updatable timed automata, *TCS*, Vol. 321, No. 2–3(2004), pp. 291–345.

[8] Bouyer, P., Dufourd, C., Fleury, E., and Petit, A.: Are Timed Automata Updatable?, *CAV'00*, LNCS, Vol. 1855, Springer, 2000, pp. 464–479.

[9] Clemente, L. and Lasota, S.: Timed pushdown automata revisited, *LICS '15*, IEEE, 2015, pp. 738–749.

[10] Finkel, A. and Schnoebelen, P.: Well-structured transition systems everywhere!, *TCS*, Vol. 256, No. 1–2(2001), pp. 63–92.

[11] Fontana, P. and Cleaveland, R.: A Menagerie of Timed Automata, *ACM Comput. Surv.*, Vol. 46, No. 3(2014), pp. 40:1–40:56.

[12] Higman, G.: Ordering by divisibility in abstract algebras, *Proc. London Math. Soc.*, Vol. 2(1952), pp. 326–336.

[13] Manasa, L., Krishna, S., and Nagaraj, K.: Updatable Timed Automata with Additive and Diagonal Constraints, *Logic and Theory of Algorithms*, LNCS, Vol. 5028, Springer, 2008, pp. 407–416.

[14] Mayr, R.: Undecidable problems in unreliable computations, *TCS*, Vol. 297, No. 1—3(2003), pp. 337–354.

[15] Schmitz, S.: Complexity Hierarchies Beyond Elementary, Manuscript arXiv:1312.5686v2 [cs.CC], 2013.

[16] Schnoebelen, P.: Lossy Counter Machines Decidability Cheat Sheet, *RP'10*, LNCS, Vol. 6227, Springer, 2010, pp. 51–75.

[17] Schnoebelen, P.: Revisiting Ackermann-Hardness for Lossy Counter Machines and Reset Petri Nets, *MFCS'10*, LNCS, Vol. 6281, Springer, 2010, pp. 616–628.

[18] Waez, M. T. B., Dingel, J., and Rudie, K.: A survey of timed automata for the development of real-time systems, *Computer Science Review*, Vol. 9(2013), pp. 1–26.