情報処理学会論文誌　プログラミング　Vol.5　No.2　43–66 (Mar. 2012)

確率時間CEGARの開発とその実証実験

清水 隆也[1]　森下 篤[1]　山根 智[1,a)]

受付日 2011年9月26日, 採録日 2011年12月20日

概要：本論文では，確率時間オートマトンの到達可能性解析に述語抽象化と反例を用いた精錬の枠組み（CEGAR）を適用する手法を提案する．CEGARを用いることにより，組込システムのようなリアルタイム動作，確率動作を持つシステムに対する，効率的な自動検証が可能となる．

キーワード：確率リアルタイムシステム，確率時間オートマトン，モデル検査，抽象化精錬，反例解析

Development and Experiments of Probabilistic Timed CEGAR

Takaya Shimizu[1]　Atsushi Morimoto[1]　Satoshi Yamane[1,a)]

Received: September 26, 2011, Accepted: December 20, 2011

Abstract: In this paper, we present an efficient verification method for probabilistic timed automaton. This method based on predicate abstraction and refinement realizes effective automated verifications for real-time and probabilistic embedded systems.

Keywords: probabilistic real-time system, probabilistic timed automaton, model checking, abstract and refinement, counterexample analysis

1. 導入

1.1 背景

Clarkeらによって，リアクティブシステムの反例による抽象化精錬の枠組み（CEGAR）[7]のモデル検査[5]が提案された．モデル検査とはシステムが特定の性質を満たしているか調べることであり，システムの動作を網羅的に検証する必要がある．そのためモデル検査ではシステムの状態数が大きくなってしまう状態爆発の解決が課題になる．この状態爆発に対し，述語抽象化[8]を導入し，状態数を抑えながら検証可能なCEGARに注目が集まっている．

モデル検査で対象とする性質として，“システムが動作中に危険な状態に到達しない”という安全性が最も典型的であり，これは到達可能性解析によって検証可能である．本研究は，確率時間オートマトンに対してCEGARを適用し，到達可能性解析による安全性の検証を行う．

1　金沢大学大学院自然科学研究科
Graduate School of Natural Science and Technology, Kanazawa University, Kanazawa, Ishikawa 920–1192, Japan
a)　syamane@is.t.kanazawa-u.ac.jp

1.2 確率時間CEGAR

一般に，抽象化を行うとシステムは本来の性質を損なう．CEGARでは，抽象モデル上での反例の候補の導出と，反例を用いた抽象モデルの精錬を，結論が得られるまで繰り返すことで正当性を保証する．CEGAR[7]による検証の枠組みは以下の手順で行われる．

(1) 抽象モデルを構築する．
(2) 抽象モデルから反例の候補を導出する．もし，反例の候補が存在しなければ“検証したい性質を満たす”と出力する．
(3) 導出した反例の候補が具体モデルで動作可能な反例であるか解析する．もし，具体モデルで動作可能であれば“検証したい性質を満たさない”と出力する．
(4) 反例が存在しなければ，偽反例となった反例の候補を取り除くように抽象モデルを精錬する．
(5) (2)に戻る．

この手順のため，偽反例から得た情報により，検証に必要な部分のみを詳細化することで状態数を抑えることができる．CEGARによる枠組みを用いて検証を行うには以下の手法を確立しなければならない．

- 検証したい性質を抽象モデルが持っていれば，具体モデルも持っている健全性を保つ抽象化手法
- 抽象モデルから反例の候補を導出でき，その反例の候補が実動作可能な反例であるか解析する反例解析手法
- 反例解析の結果，反例の候補に対応する反例が存在しないとき，同じ反例の候補を生じさせないように抽象モデルを精錬する手法

本論文ではこれらを確立し，確率時間オートマトンを対象としてCEGARを導入することで，検証対象に応じて抑えられた状態空間の構築が可能であることを示す．上記手法の開発において，確率分岐による複数パスの同時の実行可能性を同時実行反例解析によって判定し，その偽反例による抽象モデルの精錬手法を実現する．また，本手法の実装実験を行い，既存手法 [13] と比較することで，より小さい状態空間での検証が可能であることを示す．

1.3　関連研究

これまでCEGARを適用した検証手法として，リアルタイムシステムを対象とした時間CEGAR [14] や，ハイブリッドシステムを対象とした研究 [1], [6]，確率システムを対象とした確率CEGAR [11]，等が研究されてきた．

本研究で検証対象とするのは，確率システムおよびリアルタイムシステムの両方の性質をあわせ持つ，確率リアルタイムシステムである．確率リアルタイムシステムに対する検証を行ううえでの課題として，同時実行可能性の解析手法が必要である．CEGARを適用するにあたり，同時実行可能性の解析手法はこれまで提案されておらず，本論文ではその手法を示す．

一方，確率リアルタイムシステムに対する検証の既存手法として，記号モデル検査 [13] がある．また，確率時間オートマトンのモデル検査の計算量に関する研究 [16] もある．CEGARは，状態数を抑えながら検証可能な手法であり，導入することで効率的な検証を期待できる．本研究では確率時間CEGARを計算機上に実装して，上記手法との性能比較を行う．

1.4　本論文の構成

まず2章において確率時間オートマトンを導入し，その意味である時間確率システムについて説明する．3章では，CEGARの導入に必要な時間確率システムに対する述語抽象化について説明し，4章で確率時間CEGARによる検証の流れを説明する．確率時間CEGARの中でも重要な処理である反例解析について5章で，精錬について6章でそれぞれ説明する．本手法の実装および，その比較実験の結果を7章で述べ，8章でまとめる．

2.　確率時間オートマトン

この章では，確率時間オートマトンと，その確率時間オートマトンの安全性を検証する確率到達可能性問題を定義する．確率到達可能性問題は，確率時間オートマトンの意味にあたる時間確率システム上で定義される．

2.1　準備

まず，確率時間オートマトンの動作のランダム性を表現する確率分布と，リアルタイム性を表現するクロック変数について定義する．

定義 2.1（確率分布）．可算状態集合 Q 上の離散確率分布を関数 $p : Q \to [0,1]$ と表す．この関数は，$\sum_{q\in Q} p(q) = 1$ を満たす．非可算集合 Q_∞ において，$\mathsf{Dist}(Q_\infty)$ を Q_∞ の有限部分集合上の確率分布の集合とする．

□

定義 2.2（クロック変数とクロック評価）．クロック変数 x は非負の実数値をとる変数であり，すべてのクロックが同じ割合で増加する．すべてのクロック変数の集合を C とする．クロック評価 ν はそれぞれのクロック変数 x に実数を割り付ける関数 $\nu : C \to \mathbb{R}^{\geq 0}$ である．すべてのクロック評価からなる集合を $\mathcal{V}_C$ とする．$\delta \in \mathbb{R}$ とすると，クロック評価 $(\nu+\delta)$ は，すべての $x \in C$ について，$(\nu+\delta)(x) = \nu(x)+\delta$ となるクロック評価である．$X \subseteq C$ において，$\nu[X := 0]$ は X 上のすべてのクロック変数 $x \in X$ を 0 にリセットし，他のクロック変数は変化しないクロック評価を意味する．また，ν_0 をすべてのクロック変数において 0 であるクロック評価とする．

□

クロック評価を集合として扱うゾーンを定義する．これは確率時間オートマトンの動作の制約を表現するだけではなく，時間抽象化の定義や反例解析のアルゴリズムとしても広く用いる．クロック変数を用いたリアルタイムシステムの表現ではよく利用される．

定義 2.3（ゾーン）．ゾーンは以下ような構文で定義される．

$$\zeta ::= x \leq c \mid x < c \mid x > c \mid x \geq c \mid x_1 - x_2 \leq d \mid x_1 - x_2 < d \mid \mathrm{true} \mid \zeta \wedge \zeta$$

ここで，$x \in C$，$c \in \mathbb{N}$，$d \in \mathbb{Z}$ である．すべてのクロック変数 $x \in C$ について，あるクロック評価 $\nu(x)$ によって値を割り付けたとして，ζ で表される式が真となるなら，クロック評価 ν は ζ を満たすとし，$\nu \triangleright \zeta$ と記述する．以降，ゾーン ζ を，ζ を満たすクロック評価の集合として扱う．クロック評価 ν_0 のみを満たすゾーンを $\zeta_0 = \{\nu_0\}$ とする．また，C の元によって構成可能なゾーンの集合を $Zones(C)$ とする．

□

2.2　構文

確率リアルタイムシステムの記述モデルとして確率時間オートマトン [13] を定義する．

定義 2.4（確率時間オートマトン）．確率時間オートマトン G は以下の組 $(L, l_0, C, Inv, prob)$ で定義される．

- ロケーションの有限集合 L
- 初期ロケーション $l_0 \in L$
- クロック変数の有限集合 C
- ロケーションに不変条件を割り付ける関数 $Inv : L \to Zones(C)$
- 確率遷移関係の有限集合 $prob \subseteq L \times Zones(C) \times \mathsf{Dist}(2^C \times L)$

□

G の状態 s はロケーションとクロック評価の対 (l, ν) で表現される．G の動作は，初期ロケーション l_0 とすべてのクロック変数の値が 0 であるクロック評価の対である初期状態 (l_0, ν_0) から開始する．(l_0, ν_0) から状態間を時間遷移か離散遷移を行うことによって動作する．

時間遷移は同一ロケーション内で行い，状態 (l, ν) から $t \in \mathbb{R}^{>0}$ 時間遷移する場合は，確率 1 で状態 $(l, \nu + t)$ になる．ただしロケーションの不変条件により，t は $\nu + t \triangleright Inv(l)$ を満たさなければならない．

離散遷移は確率遷移関係 $(l, \zeta_g, p) \in prob$ を用いてロケーション間で行う遷移である．確率遷移関係は，遷移元ロケーション $l \in L$，遷移条件 $\zeta_g \in \mathrm{Zones}(C)$，確率分布 $p \in \mathsf{Dist}(2^C \times L)$ の組である．確率分布 $p : 2^C \times L \to (0, 1]$ は，リセットするクロック変数の集合 $X \in 2^C$ と，遷移先ロケーション $l' \in L$ から遷移確率 $p(X, l') \in (0, 1]$ を与える写像である．

状態 (l, ν) からの離散遷移を考える．遷移関係が (l, ζ_g, p)，リセットするクロックの集合が X，遷移先ロケーションが l' であったとき，離散遷移先の状態は $(l', \nu[X := 0])$ となる．

以下の 2 つを満たすとき離散遷移可能である．

- 遷移元状態のクロック評価が遷移条件を満たしている．
- 遷移先状態のクロック評価が遷移先ロケーションの不変条件を満たしている．

つまり，$\nu \triangleright \zeta_g$ かつ $\nu[X := 0] \triangleright Inv(l')$ でなければならない．

例 2.1. 図 1 の確率時間オートマトン G_1 について，その動作例を示す．$x, y \in C$ はクロック変数であり，$\mathsf{start}, \mathsf{done}, \mathsf{abort} \in L$ はロケーションである．

図 1 における start から確率 0.2 で start に，確率 0.8 で done に離散遷移する確率分布を p_0 とする．

(1) 初期状態（$\mathsf{start}, x = 0 \wedge y = 0$）
(2) 2 単位時間の時間遷移（$\mathsf{start}, x = 2 \wedge y = 2$）（ここでは遷移条件 $x \leq 2$ より，2 単位時間以上時間遷移できない）
(3) 確率遷移関係（$\mathsf{start}, x \geq 1, p_0$）により確率 0.2 の離散遷移（$\mathsf{start}, x = 0 \wedge y = 2$）（リセット $[\{x\} := 0]$ により，クロック変数 x のみ 0 になる）

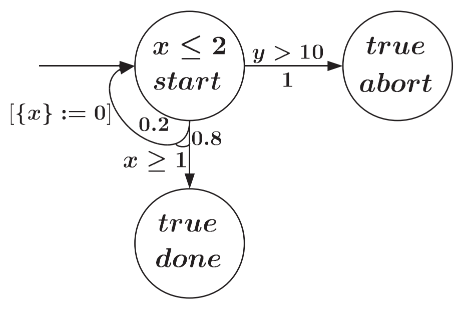

図 1　確率時間オートマトン G_1
Fig. 1 Probabilistic timed automaton G_1.

(4) 1.2 単位時間の時間遷移（$\mathsf{start}, x = 1.2 \wedge y = 3.2$）
(5) 確率遷移関係（$\mathsf{start}, x \geq 1, p_0$）により 0.8 の確率で（$\mathsf{done}, x = 1.2 \wedge y = 3.2$）

以上のように，時間遷移，離散遷移を繰り返すことにより，システムは動作する．

□

2.3 意味論

確率時間オートマトン G の意味を時間確率システム $\mathcal{M}$ [13] とし，抽象化における具体モデルとする．$\mathcal{M}$ はマルコフ決定過程の形をとる．さらに特定の状態への到達確率を定義するため，$\mathcal{M}$ の非決定を解決するアドバサリを導入することにより離散時間マルコフ連鎖の形に変換する．

定義 2.5（時間確率システム）．確率時間オートマトン $G = (L, l_0, C, Inv, prob)$ の意味となる時間確率システム $\mathcal{M}$ を組 $(S, s_0, Steps)$ とする．

- 状態集合 $S \subseteq L \times \mathcal{V}_C$
- 初期状態 $s_0 = (l_0, \nu_0)$
- 状態遷移関係 $Steps \subseteq S \times \mathbb{R}^{>0} \times \mathsf{Dist}(2^C \times S)$

状態集合 S に含まれる状態 $(l, \nu) \in S$ は $\nu \triangleright Inv(l)$ を満たしていなければならない．

状態遷移関係 $Steps$ は時間遷移と離散遷移からなる．状態遷移関係における確率分布を $\mu \in \mathsf{Dist}(2^C \times S)$ とする．

時間遷移における経過時間を $t \in \mathbb{R}^{>0}$ 単位時間，G における確率遷移関係を $(l, \zeta_g, p) \in prob$ として，$(l, \nu) \in S$ から $(l', \nu') \in S$ への遷移関係 $((l, \nu), t, \mu) \in Steps$ を以下のように定義する．また，時間遷移における確率分布を特に $\mu_\perp$ とする．

- t 単位時間の時間遷移

$$(l, \nu) \overset{t, \mu_\perp(\emptyset, (l', \nu'))}{\longrightarrow} (l', \nu')$$

ただし，

$$\mu_\perp(\emptyset, (l', \nu')) = \begin{cases} 1 & \text{if } l' = l \wedge \forall t'.(0 \leq t' \leq t \wedge \\ & \nu' = \nu + t' \wedge \nu' \triangleright Inv(l)) \\ 0 & \text{otherwise} \end{cases}$$

- (l, ζ_g, p) による離散遷移

$$(l,\nu) \overset{0,\mu(X,(l',\nu'))}{\longrightarrow} (l',\nu')$$

ただし，

$$\mu(X,(l',\nu')) = \begin{cases} p(X,l') & \text{if } \nu \triangleright \zeta_g \wedge \nu' = \nu[X:=0] \\ 0 & \text{otherwise} \end{cases}$$

□

$\mathcal{M}$ 上の状態遷移関係の確率 $\mu(X,(l',\nu'))$ による遷移は，クロック変数 X をリセットして状態 (l',ν') へ到達する遷移である．

離散遷移について，G 上の確率分布 $p(X,l')$ には $\mathcal{M}$ 上の確率分布 $\mu(X,(l',\nu'))$ が対応しているため，これらの確率は等しい．

$\mathcal{M}$ のパス ω は初期状態 s_0 から始まる以下のような非空の有限または無限列である．

$$\omega = s_0 \overset{t_0,\mu_0(X_0,s_1)}{\longrightarrow} s_1 \overset{t_1,\mu_1(X_1,s_2)}{\longrightarrow} \cdots \overset{t_{i-1},\mu_{i-1}(X_{i-1},s_i)}{\longrightarrow} s_i \overset{t_i,\mu_i(X_i,s_{i+1})}{\longrightarrow} \cdots$$

i 番目の遷移において，t_i は時間遷移量，μ_i は確率分布，X_i はリセットクロック集合，s_{i+1} は遷移先状態を表す．

ここで，本研究で扱う時間確率システムは strictdivergence [16] であることに注意する．strict divergence とは，すべてのパスが時間発散する性質である．また，strict divergence な時間確率システムのモデル検査は EXPTIME 完全であることが知られている [16]．

有限長のパスを ω_{fin} と表記する．ω_{fin} について，$|\omega_{fin}|$ をパスの長さ（遷移の回数），$last(\omega_{fin})$ をパスの最後の状態とする．すべての有限長のパスからなる集合を $Path_{fin}$ とする．

無限長のパスを ω_{ful} と表記する．すべての無限長のパスからなる集合を $Path_{ful}$ とする．

有限長および無限長のパスについて，i 番目の状態をそれぞれ $\omega_{fin}(i)$ および $\omega_{ful}(i)$ とする．

特定の状態集合 S_e に到達するパスの集合を以下のように定義する．

- 有限長のパスについて
  $Path_{fin}(S_e) = \{\omega_{fin} \in Path_{fin} | last(\omega_{fin}) \in S_e\}$
- 無限長のパスについて
  $Path_{ful}(S_e) = \{\omega_{ful} \in Path_{ful} | \exists i.\omega_{ful}(i) \in S_e\}$

次に，時間確率システムの非決定的選択を解決するものとして，アドバサリを導入する．

定義 2.6（時間確率システムのアドバサリ）．$\mathcal{M}$ のアドバサリ A は，有限長のパス ω_{fin} から，パスの最後の状態を遷移元とする状態遷移関係 $(last(\omega_{fin}),t,\mu) \in Steps$ の時間遷移量と確率分布を割り当てる関数 $A : Path_{fin} \to \mathbb{R}^{\geq 0} \times Dist(2^C \times S)$ である．すべてのアドバサリからなる集合を $A \in Adv$ とする．

□

時間確率システムはマルコフ決定過程であるので，非決定的な遷移を持つが，アドバサリを導入することでパスの次の遷移を一意に決めることができる．アドバサリは時間遷移量と確率分布を割り当てる関数であるが，時間遷移では，時間遷移量 t と確率分布 $\mu_\perp$ が 1 対 1 で対応していることに注意する．

あるアドバサリ A によって得られる無限長のパスからなる集合を $Path^A_{ful} \subseteq Path_{ful}$，有限長のパスからなる集合を $Path^A_{fin} \subseteq Path_{fin}$ として定義する．

無限長のパス $\omega_{ful} \in Path^A_{ful}$ は，いかなる i 番目の遷移でも，その i 番目の状態までのプレフィックス $\omega_{i\text{-}th} \in Path^A_{fin}$ を用意すると，$A(\omega_{i\text{-}th}) = (t_i,\mu_i)$ となるものとして定義する．

2.4　離散時間マルコフ連鎖

任意のアドバサリ A で非決定を解決することで，時間確率システム $\mathcal{M}$ から，離散時間マルコフ連鎖 $\mathcal{MC}^A = (S, s_0, \mathrm{P}^A)$ を構築することができる．S および s_0 は時間確率システムの状態集合 S と初期状態 s_0 にそれぞれ一致する．

離散時間マルコフ連鎖上の有限長のパスを用いて，遷移確率関数 $\mathrm{P}^A : Path^A_{fin} \times Path^A_{fin} \to [0,1]$ を以下のように定義する．

$$\mathrm{P}^A(\omega_{fin},\omega'_{fin}) = \begin{cases} \mu(X,s') & \text{if } \exists\mu.(A(\omega_{fin}) = (t,\mu) \wedge \\ & \quad \omega'_{fin} \text{ is the form of} \\ & \quad \omega_{fin} \overset{t,\mu(X,s')}{\longrightarrow} s') \\ 0 & \text{otherwise} \end{cases}$$

次に，有限長のパス ω_{fin} の確率 $Prob^A_{fin} : Path^A_{fin} \to [0,1]$ を以下のように定義する．

$$Prob^A_{fin}(\omega_{fin}) \overset{\text{def}}{=} \begin{cases} 1 & \text{if } |\omega_{fin}| = 0 \\ \prod_{i=0}^{|\omega_{fin}|-1} \mathrm{P}^A(\omega_{i\text{-}th}, \omega_{(i+1)\text{-}th}) & \text{otherwise} \end{cases}$$

$\omega_{i\text{-}th}$ は ω_{fin} の i 番目の状態までのプレフィックスである．

ここで，アドバサリ A における，有限長のパス ω_{fin} のシリンダ集合を以下のように定義する．

$$C^A(\omega_{fin}) \overset{\text{def}}{=} \{\omega \in Path^A_{ful} | \omega_{fin} \text{ は } \omega \text{ のプレフィックス}\}$$

$\omega_{fin} \in Path_{fin}$ におけるシリンダ集合 $C^A(\omega_{fin})$ を元として包含する $Path^A_{ful}$ 上の最小完全加法族を $C^A(\omega_{fin}) \in \Sigma^A$ とする．さらに，すべての $\omega_{fin} \in Path_{fin}$ について，Σ^A における確率測度 $Prob^A$ を以下のように定義する．

$$Prob^A(C^A(\omega_{fin})) \overset{\text{def}}{=} Prob^A_{fin}(\omega_{fin})$$

ここで，状態集合 S_e に到達する無限長のパスからなる集合は $\{\omega \in Path^A_{ful} | \exists i \in \mathbb{N}.\omega(i) \in S_e\} \in \Sigma^A$ となることに注

意する．Σ^A は $Path^A_{ful}$ のべき集合であると考えると，このパス集合は Σ^A の元となるのは自明である．このことから，初期状態 s_0 から，アドバサリ A において，ある状態集合 S_e への到達確率は $Prob^A(\{\omega \in Path^A_{ful} | \exists i \in \mathbb{N}.\omega(i) \in S_e\})$ として求められる．以降，この到達確率を単に $Prob^A(S_e)$ と記載する．

マルコフ決定過程の最大到達確率はシンプルなアドバサリによって非決定を解決したときであることが知られている [3]．しかし，本章で示した時間確率システムは時間をともなうマルコフ決定過程であり，その最大到達確率を得るためのアドバサリはシンプルとはならない [16]．シンプルなアドバサリとは，パスの最後の状態が等しければ，同じ確率分布を返すアドバサリである．

2.5　確率到達可能性問題と反例

到達可能性問題はソフトウェアの検証における最も基本的な問題であり，様々な検証問題は到達可能性問題に帰着させることができる．本研究は，時間確率システムの安全性を到達可能性問題によって検証する．

G の到達可能性問題を次のように定義する．

定義 2.7（到達可能性問題）．ロケーション集合 $L_e \subseteq L$ について，ロケーション $l_e \in L_e$ を持つ状態の集合を $S_e = \{(l_e, \nu) \in S | l_e \in L_e\}$ とする．また，$\lambda \in [0,1]$ を s_0 から状態集合 S_e への到達確率とする．

G の到達可能性問題を，到達確率 $\lambda \in [0,1]$ と目的ロケーションの集合 $L_e \subseteq L$ の組 (λ, L_e) とする．

到達可能性問題 (λ, L_e) の答えが “*yes*” であるとは，$\mathcal{M}$ において，$\forall A.Prob^A(S_e) \leq \lambda$ の場合に限る．それ以外の場合は “*no*” である．

□

L_e は到達することが望ましくないロケーションの集合，S_e はそのような状態の集合である．

また，確率時間オートマトン G の到達可能性問題 (λ, L_e) は次の命題に帰着できる．

命題 2.1（到達可能性問題の帰着）．いかなるアドバサリ A によっても，S_e へ確率 λ 以下でしか到達できないのであれば，(λ, L_e) の答えは “*yes*” である．それ以外，すなわち λ を超える確率で S_e へ到達可能なアドバサリ A が存在するのであれば，(λ, L_e) の答えは “*no*” である．

□

証明． アドバサリ A において，状態集合 S_e への到達確率が λ 以下であるとは $Prob^A(S_e) \leq \lambda$ である．よって，$\forall A.Prob^A(S_e) \leq \lambda$ であることと，いかなるアドバサリ A によっても，状態集合 S_e へ確率 λ 以下でしか到達できないことは等価である．

到達可能性問題 (λ, L_e) の答えが “*yes*” であるのは $\forall A.Prob^A(S_e) \leq \lambda$ の場合に限られており，それ以外，すなわち $\neg(\forall A.Prob^A(S_e) \leq \lambda) = \exists A.Prob^A(S_e) > \lambda$ の場合は “*no*” であることから，到達可能性問題 (λ, L_e) から命題 2.1 へ帰着できる． □

このように帰着した命題の答えが “*no*” である場合，望ましくない状態集合 S_e への到達確率が λ より大きくなるアドバサリが存在するため，システムが安全ではないことが分かる．

次に，反例を以下のように定義する．

定義 2.8（反例）．あるアドバサリ A で非決定を解決した離散時間マルコフ連鎖 $\mathcal{MC}^A$ 上で得られる有限長のパスの集合を $Path^A_{fin}$ とする．$Path^A_{fin}$ のうち，目的状態集合 S_e に到達するパスからなる有限集合を $\Omega \subseteq \{\omega \in Path^A_{fin} | last(\omega) \in S_e\}$ とする．このとき，

$$\sum_{\omega \in \Omega} Prob^A_{fin}(\omega) > \lambda$$

となる A と Ω が存在するならば，それらを組 (A, Ω) とし，その組を反例とする．

□

反例とは，到達可能性問題の解が “*no*” となる証拠である．本研究では，到達可能性問題を対象としており，状態 $s \in S_e$ で終わる有限パスによってのみ，到達可能性問題の判定ができる．よって，本研究で扱う反例は有限パスの集合とする．

到達可能性問題（定義 2.7）では，“*yes*” となる条件として等号を含む不等式に限定されていることに注意する．これにより，文献 [9] によって離散時間マルコフ連鎖における PCTL の検証において，有限長のパスの有限集合で構成される反例が存在することが示されている．本研究で対象とする到達可能性問題についても，以下の定理が成り立つ．

定理 2.1（到達確率と到達パスの集合）．もし $Prob^A(S_e) > \lambda$ ならば，そのときに限り S_e に到達する有限パスの有限集合 $\Omega \subseteq \{\omega \in Path^A_{fin} | last(\omega) \in S_e\}$ で $\sum_{\omega \in \Omega} Prob^A_{fin}(\omega) > \lambda$ となる反例が存在する．

□

証明． ある到達可能性問題 (λ, L_e) について s_0 から $s_e \in S_e$ への反例であるパス集合が有限ではないと仮定する．つまり，無限個のパスからなる反例のみが存在すると仮定する．この反例のパス集合を $\Omega = \{\omega_1, \omega_2, \cdots\}$ とする．この反例のパス集合について，その合計到達確率は

$$\sum_{i=1}^{\infty} Prob^A_{\omega_i} = \lim_{j \to \infty} \sum_{i=1}^{j} Prob^A_{\omega_i} \tag{1}$$

となる．この左辺を d，右辺を d_j とする．

ここで $\forall \epsilon > 0$，$\exists N_e \in \mathbb{N}$，$\forall n \geq N_e.|d - d_n| < \epsilon$ である ϵ について，$0 < \epsilon < d - \lambda$ とする．式 (1) より，ある $n > N_e$ に対して，$|d - d_n| < d - \lambda$ であり，$d_n > \lambda$ である．

つまり，このとき $\Omega' = \{\omega_1, \omega_2, \cdots, \omega_n\}$ は $Prob^A_{\omega_i}(\Omega') >$

λ となるため反例である．よって仮定と矛盾する．

したがって，パスの有限集合で構成される反例が存在する．

□

このように，反例を有限長のパスの有限集合として扱うことにより，パスの解析によって反例の存在を確かめることができる．

2.6　制限された時間確率システム

本研究で扱う時間確率システム（定義 2.5）は strict divergence [16] であり，zeno となる動作を含まない non-zeno なシステムであるが，時間確率システムを抽象化すると，その抽象モデルにおいて zeno である動作が現れることがある．そこで，本研究では文献 [14] の時間システムに対する手法を拡張し，時間確率システムに適用することで，次のように制限された時間確率システム $\mathcal{M}_R$ のみを扱うものとする．

定義 2.9（制限された時間確率システム）．G の意味となる制限された時間確率システム $\mathcal{M}_R = (S, s_0, Steps_R)$ は，時間確率システム $\mathcal{M} = (S, s_0, Steps)$ のうち，その時間遷移を制限したものである．まず，以下を満たす $t \in \mathbb{R}$ を考える．

$$\exists x \in C.\exists k \in \{0, \cdots, c\}.(\nu(x) = k \vee (\nu(x) < k \wedge \nu(x) + t \geq k))$$

$c \in N$ は時間確率システム $\mathcal{M}$ に現れる最大の定数である．このような t に対し，次のような時間遷移を考える．

時間遷移 $(l, \nu) \xrightarrow{t, \mu_\perp(\emptyset, (l', \nu'))} (l', \nu')$

ただし，$\mu_\perp$ は

$$\mu_\perp(\emptyset, (l', \nu')) = \begin{cases} 1 & \text{if } l' = l \wedge \nu' = \nu + t \wedge \nu' \in Inv(l) \\ 0 & \text{otherwise} \end{cases}$$

時間確率システム $\mathcal{M}$ のうち，このような時間遷移のみを持つものを，制限された時間確率システム $\mathcal{M}_R$ とする．

□

定理 2.2（制限された時間確率システムの正当性）．時間確率システム $\mathcal{M}$ の到達可能性問題 (λ, L_e) の解が "*yes*" であることと，制限された時間確率 $\mathcal{M}_R$ システムの到達可能性問題 (λ, L_e) の解が "*yes*" であることは等価である．

証明． non-zeno な時間確率システムの到達可能性問題 (λ, L_e) の解が "*yes*" であるとする．$S_e = \{(l_e, \nu) \in S | l_e \in L_e\}$ として，任意のアドバサリ A に対して，$Prob^A(S_e) \leq \lambda$ となる時間確率システム上の以下のパスを考える．

$$(l_0, \nu_0) \xrightarrow{t_0, \mu_0(X_0, (l_1, \nu_1))} (l_1, \nu_1) \xrightarrow{t_1, \mu_1(X_1, (l_2, \nu_2))} \cdots \xrightarrow{t_n, \mu_n(X_n, s_e)} (l_e, \nu_e)$$

ただし，$0 \leq t'_i \leq t_i$ に対して，$\nu_i + t'_i \triangleright Inv(l_i)$ である．さらに non-zeno であるため，以下の条件が課される．任意の $t_i \in \mathbb{R}$ に対して，

$$\exists j \in \mathbb{N}. \sum_{i=0}^{j} t_i > t$$

一方，制限された時間確率システムのパスは，時間遷移に以下の制限を加えたものである．任意の $t_i \in \mathbb{R} > 0$ $(i = 0, \cdots, n)$ に対する ν_i について，

$$\exists x \in C.\exists k \in \{0, \cdots, c\}.(\nu_i(x) = k \vee (\nu_i(x) < k \wedge \nu_i(x) + t_i \geq k))$$

ここで，non-zeno な時間確率システムのパスにおいて，次のような $t_a, \cdots, t_b$ となる時間経過の列を考える．ただし，$a, b \in \mathbb{N}$，$a < b$，$x \in C$，$k \in \{0, \cdots, c\}$ とする．

$$\nu_a(x) + \sum_{i=a}^{b-1} t_i < k \wedge \nu_a(x) + \sum_{i=a}^{b} t_i \geq k$$

このように連続する時間遷移を，

$$(l_a, \nu_a) \xrightarrow{\sum_{i=a}^{b} t_i, \mu(X, (l_b, \nu_b))} (l_b, \nu_b)$$

で置き換えることで，non-zeno な時間確率システムでの (l_0, ν_0) から (l_e, ν_e) までのパスに対応する，制限された non-zeno な時間確率システムにおけるパスを構成できる．時間遷移の確率は 1 なので，$Prob^A(S_e)$ は変化しない．

また，このパスは $\nu_i(x) + t_i \geq k$ を満たしており，$\sum_{i=0}^{j}(\nu_i(x) + t_i) > \sum_{i=0}^{j} k_i$ となる．ただし，$k_i \in \{0, \cdots, c\}$ である．つまり，任意の $\sum_{i=0}^{j} k_i \in \mathbb{N}$ に対して，$\sum_{i=0}^{j}(\nu_i(x) + t_i)$ となる $j \in \mathbb{N}$ が存在する．$\sum_{i=0}^{j} k_i$ は発散するので，$\sum_{i=0}^{j}(\nu_i(x) + t_i)$ も発散する．したがって，このパスも non-zeno である [10]．

以上より，制限された non-zeno な時間確率システムにおいても，対応する non-zeno なパスを構成可能であるため，その到達可能性問題 (λ, L_e) の解は "*yes*" である．(1)

制限された non-zeno な時間確率システムの到達可能性問題 (λ, L_e) の解が "*yes*" であるとする．任意のアドバサリ A に対して，$Prob^A(S_e) \leq \lambda$ となる制限された時間確率システム上の以下の non-zeno なパスを考える．

$$(l_0, \nu_0) \xrightarrow{t_0, \mu_0(X_0, (l_1, \nu_1))} (l_1, \nu_1) \xrightarrow{t_1, \mu_1(X_1, (l_2, \nu_2))} \cdots \xrightarrow{t_n, \mu_n(X_n, s_e)} (l_e, \nu_e)$$

ただし，制限されたシステムであるので，遷移 $t_i, \mu_i(X_i, (l_{i+1}, \nu_{i+1}))$ に対し，$t_i > 0$ $(i = 0, \cdots, n)$ である任意の t_i と対応する ν_{i+1} について $\exists x \in C.\exists k \in \{0, \cdots, c\}.(\nu_i(x) = k \vee (\nu_i(x) < k \wedge \nu_i(x) + t_i \geq k))$ が成り立つ．

さらに，non-zeno であるため，任意の $t_i \in \mathbb{R}$ に対して，$\exists j \in \mathbb{N}. \sum_{i=0}^{j} t_i > t$ である．

ここで，制限された non-zeno な時間確率システムのパスのうち，次のような時間遷移を考える．

$$(l_i, \nu_i) \stackrel{t_i, \mu_i(X_i, (l_{i+1}, \nu_{i+1}))}{\longrightarrow} (l_{i+1}, \nu_{i+1})$$

システムに加えた時間遷移の制限を取り除き，時間遷移を緩和して，新たな時間遷移を追加する．

$$(l_i, \nu_i) \stackrel{t_i, \mu_i(X_i, (l_j, \nu_j))}{\longrightarrow} (l_j, \nu_j) \stackrel{t_j, \mu_j(X_j, (l_{i+1}, \nu_{i+1}))}{\longrightarrow} (l_{i+1}, \nu_{i+1})$$

ただし，$0 \leq t'_j \leq t_j$ に対して，$\nu_j + t'_j \triangleright Inv(l_j)$ である．こうすることで，(1) と同様のパスを構成できる．追加した遷移は時間遷移であるのでパスの確率は変化せず，$Prob^A(S_e)$ も変化しない．明らかに，時間確率システムの non-zeno なパスが構成できて，その到達可能性問題 (λ, L_e) の解は "*yes*" である．(2)

以上 (1) および (2) より，等価である．

□

この定理は，制限された時間確率システム $\mathcal{M}_R$ と，制限されていない時間確率システム $\mathcal{M}$ の到達可能性問題の解は一致することを示している．そこで本研究では，抽象化精錬を用いて $\mathcal{M}_R$ の到達可能性問題を検証することで，$\mathcal{M}$ の到達可能性問題を判定する．

今後，特に明示しない限り，時間確率システムはこの制限された時間確率システムのことを指すものとし，$\mathcal{M}_R$ を単に $\mathcal{M}$ と表記する．

3. 述語抽象化

述語抽象化 [8] は無限状態遷移系の有限の近似を計算し，状態爆発を抑制するために用いられる．リアルタイムシステムの検証では，述語を用いて実数変数であるクロック評価の抽象化が有効である．本手法は文献 [14] の時間オートマトンへの適用に従い，確率に拡張して利用する．

3.1 抽象化述語と述語抽象化

まず，クロック評価を抽象化する述語として，ゾーンの定義から連言を取り除いたものを抽象化述語として定義する．

定義 3.1（抽象化述語）．クロック変数の集合 C において，述語 ψ は以下のように定義される．

$$\psi ::= x_1 \leq c | x_1 < c | x_1 - x_2 < d | \mathrm{true}$$

ここで，$x_1, x_2 \in C$，$c \in \mathbb{N}$，$d \in \mathbb{Z}$ である．クロック評価 ν，抽象化述語 ψ において，ν に関する述語 ψ の真偽値を $\psi\nu \in \{true, false\}$ とすると，ψ 中のクロック変数 $x \in C$ に対応する値 $\nu(x)$ を代入した結果得られる式が真となるとき，かつそのときに限り ν は ψ を満たし，$\psi\nu = true$ であるという．$x > c$，$x \geq c$，$x_1 - x_2 \geq d$ は，それぞれゾーンの定義にある $x \leq c$，$x < c$，$x_1 - x_2 < d$ の述語で真偽値が $\psi\nu = false$ の場合ととらえることができる．また，すべてのクロック評価 $\nu \in \mathcal{V}_C$ において $\psi = \mathrm{true}$ は，$\psi\nu = true$ とする．

□

本研究ではロケーションごとに抽象化述語の集合 $\Psi^l = \{\psi_0^l, \cdots, \psi_{n-1}^l\}$ を与える．ここで，ψ^{l_i} は，ロケーション l における抽象化述語である．また，すべてのロケーションにおける抽象化述語の族を $\mathbf{\Psi} = \{\Psi^{l_0}, \cdots, \Psi^{l_k}\}$ とする．

述語集合 Ψ^l に含まれる述語の数と等しい長さの，ビットベクトル b^l を考える．そのようなビットベクトル b^l を用いて，組 (l, b^l) を抽象状態 $s^\sharp$ とする．また，すべてのロケーションにおけるビットベクトルの集合を $\mathcal{B}$，抽象状態の集合を $S^\sharp$ とする．

$\mathbf{\Psi}$ により $\mathcal{M}$ の状態 (l, ν) から抽象状態 (l, b^l) へのマッピングである抽象化関数 $\alpha : S \to S^\sharp$ が決定される．この関数によって得られる抽象状態 $(l, b^l) = \alpha((l, \nu))$ における b^l について，その i 番目の要素を $b^l(i)$ とすると，$\psi_i^l \nu = b^l(i)$ である．たとえば，$\Psi^l = \{x \leq 1, x - y < -1\}$ において状態 $s = (l, x = 1 \wedge y = 1)$ を抽象化した抽象状態は $\alpha(s) = (l, (true, false))$ となる．α の逆像を，具体化関数 $\gamma : S^\sharp \to 2^S$ とする．α，γ ともに，全射でも単射でもない．この α と γ を以下のように定義する．

定義 3.2（抽象化・具体化）．C はクロックの集合とし，$\mathcal{V}_C$ は対応するクロック評価の集合とする．述語の有限集合 $\mathbf{\Psi} = \{\Psi^{l_0}, \cdots, \Psi^{l_k}\}$ が与えられたとき，抽象化関数 $\alpha : S \to S^\sharp$ を以下のように定義する．

$$\alpha((l, \nu)) = (l, b^l) \text{ s.t. } \forall i. b^l(i) = \psi_i^l \nu$$

また具体化関数 $\gamma : S^\sharp \to 2^S$ を以下のように定義する．

$$\gamma((l, b^l)) = \{(l, \nu) \in L \times \mathcal{V}_C | Inv(l) \wedge \bigwedge_{i=0}^{n-1} b^l(i) = \psi_i^l \nu\}$$

さらに，$b^l \Psi^l$ をその述語とビットベクトルが示すゾーンとして以下のように定義する．

$$b^l \Psi^l = \{\nu \in \mathcal{V}_C | \bigwedge_{i=0}^{n-1} b^l(i) = \psi_i^l \nu\}$$

□

3.2 抽象モデル

抽象化述語と抽象化・具体化関数を用いて，ある述語集合 $\mathbf{\Psi}$ における時間確率システムの抽象モデルを構築する．抽象モデルは時間確率システムと同様にマルコフ決定過程の形をとる．抽象モデルは，文献 [14] を確率分布の導入により拡張し，オーバ近似になるように定義する．オーバ近

似とは，具体モデルが持つ遷移を抽象モデルにすべて持たせることで，抽象化に健全性を持たせる近似である．なお，この抽象モデルの確率到達可能性問題に対する健全性の証明は次節で行う．

定義 3.3（抽象モデルの形成）．確率時間オートマトン $G = (L, l_0, C, Inv, prob)$ から変換された時間確率システム $\mathcal{M} = (S, s_0, Steps)$ における，述語集合 $\mathbf{\Psi}$ による抽象モデル $\mathcal{M}^\sharp = (S^\sharp, s_0^\sharp, Steps^\sharp)$ を以下のように構築する．

- 抽象状態集合 $S^\sharp = L \times \mathcal{B}$
- 初期抽象状態 $s_0^\sharp = \alpha(s_0)$
- 抽象状態遷移関係 $Steps^\sharp \subseteq S^\sharp \times \mathsf{Dist}(2^C \times S^\sharp)$

$\exists (l, \nu) \in \gamma((l, b)).((l, \nu), \mu) \in Steps$ であるとき，遷移 $((l, b), \mu^\sharp) \in Steps^\sharp$ が存在する．

$((l, \nu), t, \mu)$ に対応する $((l, b), \mu^\sharp)$ における $\mu^\sharp$ は以下で定義される確率分布である．ただし $\alpha((l, \nu)) = (l, b)$, $\alpha((l', \nu')) = (l', b')$ である．

$$\mu^\sharp(X, (l', b')) = \mu(X, (l', \nu'))$$

□

$Steps^\sharp$ には $Steps$ と異なり時間遷移であることを示す時間遷移量の定義はないが，$Steps$ で時間遷移を示す確率分布 $\mu_\perp$ から導かれる $Steps^\sharp$ の確率分布も区別するため $\mu_\perp^\sharp$ のように記す．

また，$\mathcal{M}^\sharp$ のパスは $\mathcal{M}$ のパスと同様に以下のように示す．

$$\omega^\sharp = s_0^\sharp \xrightarrow{\mu_0^\sharp(X_0, s_1^\sharp)} s_1^\sharp \xrightarrow{\mu_1^\sharp(X_1, s_2^\sharp)} s_2^\sharp \xrightarrow{\mu_2^\sharp(X_2, s_3^\sharp)} \cdots$$

$\mathcal{M}^\sharp$ 上のアドバサリ $A^\sharp$ は，時間遷移量の t を省き，$A^\sharp : Path_{fin}^\sharp \to Dist(2^C \times S^\sharp)$ である．$\mathcal{M}$ が時間の制約をともなうマルコフ決定過程であったのに対し，$\mathcal{M}^\sharp$ は時間のない一般的なマルコフ決定過程であるが，$\mathcal{M}$ で用いていた表現に $\sharp$ を付けることで，$\mathcal{M}^\sharp$ でも同様に表現する．

次に，述語の数を有限に制限するために basis [14] という概念を導入する．

定義 3.4（basis [14]）．$\mathcal{M}^\sharp$ において，述語集合 Ψ が *basis* であるとは，すべてのクロック評価 $\nu_1, \nu_2 \in \mathcal{V}_C$ において以下を満たすものをいう．

$$\forall \psi \in \Psi \in \mathbf{\Psi}. \psi\nu_1 \iff \psi\nu_2$$

□

basis である述語集合として，最大定数 $c_{\max}$ 以下のすべての述語を持つ述語集合が考えられる．basis である述語集合で $\mathcal{M}^\sharp$ を構築した場合，それぞれの抽象状態が表現するゾーン $b^l\Psi^l$ はクロックリージョンになる．すなわち抽象モデルは文献 [12] のリージョングラフになる．リージョンは有限数で構成されることから，basis となる述語集合も有限数で構成できる．

確率時間 CEGAR では，導入する述語を basis に含まれる述語に限定する．このことと，確率時間 CEGAR のサイクルは必ず毎回新しい述語を追加することができる手法のため，確率時間 CEGAR のサイクルは必ず有限回数で終了するといえる．述語を追加する手法および証明は 6 章で解説する．

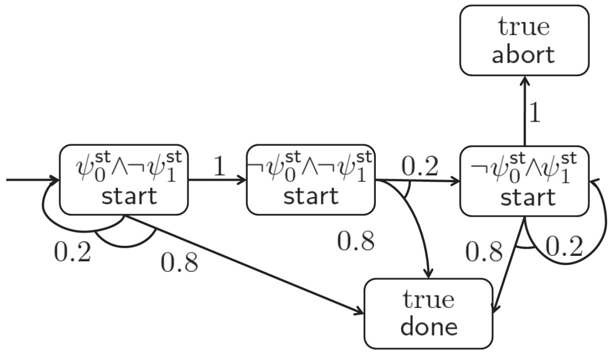

図 2　G_1 の抽象モデル $\mathcal{M}^\sharp$
Fig. 2　Abstract Model of G_1: $\mathcal{M}^\sharp$.

例 3.1. 図 1 の確率時間オートマトン G_1 を抽象化した場合の抽象モデル $\mathcal{M}^\sharp$ を図 **2** に示す．抽象化にともなう述語集合は，$\psi_0^{\mathsf{st}} = y \le 8$，$\psi_1^{\mathsf{st}} = x - y < -8$ である．

□

また，G_1 における最大定数 $c_{\max}$ は 10 である．したがって basis である述語集合 Ψ は

$$\begin{aligned}\Psi = \{&x \le 0, x \le 1, \cdots, x \le 10,\\ &y \le 0, y \le 1, \cdots, y \le 10,\\ &x < 0, x < 1, \cdots, x < 10,\\ &y < 0, y < 1, \cdots, y < 10,\\ &x - y < 0, x - y < 1, \cdots, x - y < 10,\\ &y - x < 0, y - x < 1, \cdots, y - x < 10\}\end{aligned}$$

となる．

3.3　抽象モデルと時間確率システムの関係

抽象化を行うと抽象化したシステムは元のシステムの性質を持たない場合や，またそれ以上の性質を持つ場合がある．ここでは，パスとアドバサリについて，抽象モデルと時間確率システムの対応関係を述べ，さらに $\mathcal{M}$ 上の反例に対応する $\mathcal{M}^\sharp$ 上の反例の候補を定義する．

3.3.1　パス

$\mathcal{M}$ 上の 2 つの状態 s_1，s_2 間の時間遷移は，$\mathcal{M}^\sharp$ では $\alpha(s_1) = \alpha(s_2) = s^\sharp$ であるとき 1 つの抽象状態 $s^\sharp$ 内で行われてしまう．よって反例解析で $\omega^\sharp$ から ω を導く際，$\omega^\sharp$ 内の離散遷移 $s^\sharp \xrightarrow{\mu^\sharp}$ はその抽象状態で何らかの時間遷移後に離散遷移した $s_1 \xrightarrow{t, \mu_\perp} s_2 \xrightarrow{0, \mu}$ となる．

ω に対応する $\omega^\sharp$ にはこの 1 つの抽象状態内の時間遷移を含めないことにする．

定義 3.5（パスの対応関係）．$\mathcal{M}$ と $\mathbf{\Psi}$ によって構成された

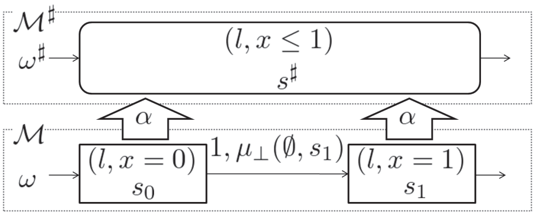

図 3 パスの対応関係
Fig. 3 Correspondence relation of paths.

$\mathcal{M}^\sharp$ において，$\mathcal{M}$ のパス ω に対応した $\mathcal{M}^\sharp$ のパス $\omega^\sharp$ とは，ω のすべての遷移に対して，以下の手順によって構成される．

(1) ω の遷移が離散遷移 $s \overset{0,\mu(X,s')}{\longrightarrow} s'$ ならば，$\omega^\sharp$ の遷移は $\alpha(s) \overset{\mu^\sharp(X,\alpha(s'))}{\longrightarrow} \alpha(s')$

(2) ω の遷移が時間遷移 $s \overset{t,\mu_\perp(\emptyset,s')}{\longrightarrow} s'$ かつ $\alpha(s) = \alpha(s')$ ならば，$\omega^\sharp$ の遷移は存在しない．

(3) ω の遷移が時間遷移 $s \overset{t,\mu_\perp(\emptyset,s')}{\longrightarrow} s'$ かつ $\alpha(s) \neq \alpha(s')$ ならば，$\omega^\sharp$ の遷移は $\alpha(s) \overset{\mu^\sharp_\perp(\emptyset,s')}{\longrightarrow} \alpha(s')$

また，このような $\omega \in Path_{fin}$ と $\omega^\sharp \in Path^\sharp_{fin}$ の対応を，抽象化関数と同じ記号を用いて，$\alpha_{\mathrm{Path}} : Path_{fin} \rightarrow Path^\sharp_{fin}$ と定義する．

□

定義 3.5 に基づき，抽象化されたパスの対応関係の例を**図 3** に示す．$\mathcal{M}$ のパス ω は s_0，s_1 を 1 単位時間で時間遷移しているが，$s^\sharp = \alpha(s_0) = \alpha(s_1)$ の場合，$\mathcal{M}^\sharp$ 上の対応するパス $\omega^\sharp$ には時間遷移が現れないものとして定義している．

3.3.2 アドバサリ

アドバサリの対応関係を，パスの対応関係を用いて定義する．

定義 3.6（アドバサリの対応関係）．$\mathcal{M}^\sharp$ のアドバサリ $A^\sharp : Path^\sharp_{fin} \rightarrow Dist(2^C \times S^\sharp)$ が $\mathcal{M}$ のアドバサリ $A : Path_{fin} \rightarrow \mathbb{R}^{\geq 0} \times Dist(2^C \times S)$ に対応しているとは，以下を満たす場合である．

$\forall \omega \in Path^A_{fin}.\forall s \in S.\forall X \subseteq C.A(\omega) = (t, \mu) \wedge A^\sharp(\alpha_{Path}(\omega)) = \mu^\sharp$ に対して，$\mu(X, s) = \mu^\sharp(X, \alpha(s))$

また，このような $A \in Adv$ と $A^\sharp \in Adv^\sharp$ の対応を抽象化関数と同じ記号を用いて，$\alpha_{\mathrm{Adv}} : Adv \rightarrow Adv^\sharp$ と定義する．

□

あるアドバサリ $A^\sharp$ について，抽象モデルにおける遷移確率行列 $\mathrm{P}^{A^\sharp} : S^\sharp \times S^\sharp \rightarrow [0, 1]$ を以下のように定義する．

$$\mathrm{P}^{A^\sharp}(s^\sharp, s^{\sharp\prime}) \overset{\mathrm{def}}{=} \begin{cases} \displaystyle\sum_{X \subseteq C} \mu^\sharp(X, s^{\sharp\prime}) & \\ \quad \text{if } \exists \omega^\sharp \in Path^{A^\sharp}_{fin}.(last(\omega^\sharp) = s^\sharp \wedge & \\ \quad \exists \mu^\sharp.A^\sharp(\omega^\sharp) = \mu^\sharp) & \\ 0 \quad \text{otherwise} & \end{cases}$$

さらに，有限長のパス $\omega^\sharp_{fin} \in Path^{A^\sharp}_{fin}$ の確率 $Prob^{A^\sharp}_{fin}(\omega^\sharp_{fin})$ を以下のように定義する．

$$Prob^{A^\sharp}_{fin}(\omega^\sharp_{fin}) \overset{\mathrm{def}}{=} \begin{cases} 1 & \text{if } |\omega^\sharp_{fin}| = 0 \\ \displaystyle\prod_{i=0}^{|\omega^\sharp_{fin}|-1} \mathrm{P}^{A^\sharp}(\omega^\sharp_{fin}(i), \omega^\sharp_{fin}(i+1)) & \\ & \text{otherwise} \end{cases}$$

定義 3.5 および，定義 3.6 より，次の定理が成り立つ．

定理 3.1（パスの対応定理）．あらゆる $\omega \in Path^A_{fin}$ において，いかなる $\mathbf{\Psi}$ による $\mathcal{M}^\sharp$ であっても，対応するパス $\alpha_{Path}(\omega) = \omega^\sharp \in Path^{A^\sharp}_{fin}$ および対応するアドバサリ $\alpha_{\mathrm{Adv}}(A) = A^\sharp \in Adv^\sharp$ が存在し，$Prob^A_{fin}(\omega) = Prob^{A^\sharp}_{fin}(\omega^\sharp)$ である．

証明． $s^\sharp$ は s から $\alpha(s) = s^\sharp$ として一意に求まるため，ω と同じ長さで ω の各状態を α で抽象化した状態列が存在する．この状態列について，初めの状態から ω に対応する遷移を追加することで，$\omega^\sharp$ を構築できることを示す．

定義 3.5 の (1) と (3) の遷移の場合，定義 3.2 より，ω の遷移に用いる状態遷移関係 $(\omega(i), t, \mu)$ に対応する $\mathcal{M}^\sharp$ の状態遷移関係 $(\alpha(\omega(i)), \mu^\sharp)$ は必ず存在し，この遷移を追加できる．(2) の遷移の場合は $\omega^\sharp$ 上で同じ状態が時間遷移で続き，遷移と遷移後の状態を取り除くことで構築することができる．この操作により ω に対応する $\omega^\sharp$ が必ず存在することを示せた．

このことから，すべての $\omega \in Path^A_{fin}$ について，対応パス $\alpha_{Path}(\omega) = \omega^\sharp$ が存在することが分かる．このようなすべての $\omega^\sharp$ について，定義 3.6 を満たすようなアドバサリ $A^\sharp$ を構成することは可能である．したがって，A に対応する $\alpha_{\mathrm{Adv}}(A) = A^\sharp \in Adv^\sharp$ が必ず存在することを示せた．

次に，$Prob^A_{fin}(\omega) = Prob^{A^\sharp}_{fin}(\omega^\sharp)$，すなわちそれらのパスについて，その確率が等しいことを示す．$|\omega| = 0$ であるなら，パスに遷移が存在しないため，対応する $\omega^\sharp$ にも遷移が構成されず，$|\omega^\sharp| = 0$ である．したがって，$Prob^A_{fin}(\omega) = Prob^{A^\sharp}_{fin}(\omega^\sharp) = 1$ である．

$|\omega| > 0$ であるなら，パスの確率は定義より

$$Prob^A_{fin}(\omega) = \prod_{i=0}^{|\omega|-1} \mu_i(X_i, \omega(i))$$

であるのに対し，$\omega^\sharp(i)$ を $\omega^\sharp$ の i 番目の抽象状態とすると，対応するパスの確率は定義より

$$Prob^{A^\sharp}_{fin}(\omega^\sharp) = \prod_{i=0}^{|\omega^\sharp|-1} \mu^\sharp_i(X_i, \omega^\sharp(i))$$

である．

ω の各遷移において，時間遷移はすべて $\mu_\perp(\emptyset, \omega(i{+}1)) = 1$ であるため，$Prob^A_{fin}(\omega)$ は離散遷移の確率について積をとったものと等しくなる．

定義 3.5 における (2) および (3) の遷移の場合，$\omega^\sharp$ において遷移が存在しない，または存在したとしても，対応する時間遷移の確率は $\mu_\perp(\emptyset, \omega(i+1)) = \mu^\sharp(\emptyset, \alpha(\omega(i+1))) = 1$ であるため，$Prob^{A^\sharp}_{fin}(\omega^\sharp)$ における確率の計算で考慮する必要がない．ω の遷移のうち，(1) の遷移の場合，$\alpha_{\mathrm{Adv}}(A) = A^\sharp$ であるため，$\mu(X, \omega(i+1)) = \mu^\sharp(X, \alpha(\omega(i+1)))$ である．定義 3.5 より，このように対応する遷移は $\omega^\sharp$ 上に必ず存在するため，ω と $\omega^\sharp$ に現れるすべての離散遷移の遷移確率をそれぞれ掛け合わせると必ず等しくなる．

したがって，$Prob^A_{fin}(\omega) = Prob^{A^\sharp}_{fin}(\omega^\sharp)$ であることを示せた．

□

定理 3.2（パスの抽象化）．$\mathcal{M}$ において，アドバサリ A によって導出されるパス集合 $Path^A_{fin}$ のうち，異なる 2 つのパス $\omega_1, \omega_2 \in Path^A_{fin}$ について，次の関係が成り立つ．

$$\alpha_{Path}(\omega_1) \neq \alpha_{Path}(\omega_2)$$

証明． 仮に $\omega^\sharp = \alpha_{Path}(\omega_1) = \alpha_{Path}(\omega_2)$ であるとする．

ω_1，ω_2 が i 番目の遷移 $\omega_1(i) \xrightarrow{t_1, \mu(X_1, \omega_1(i+1))}$，$\omega_2(i) \xrightarrow{t_2, \mu(X_2, \omega_2(i+1))}$ で初めて異なる動作をする場合を考える．

離散遷移で $X_1 \neq X_2$ の場合は，$\mathcal{M}^\sharp$ 上の $\omega^\sharp$ は遷移に X_1，X_2 の情報を持つため，1 つの $\omega^\sharp$ になることはない．

離散遷移で $\omega_1(i+1) \neq \omega_2(i+1)$ の場合は，$\mathcal{M}$ の離散遷移の確率分布は，標本空間が $2^C \times L$ である G の確率分布を継承しているため，$\omega_1(i+1)$，$\omega_2(i+1)$ のロケーションは必ず異なる．抽象化はロケーションごとに行われるため，$\alpha(\omega_1(i+1)) \neq \alpha(\omega_2(i+1))$ である．

時間遷移の場合は，アドバサリ A によって時間遷移量 t が決まるため，必ず $t_1 = t_2$ であり，$\omega_1 \neq \omega_2$ に反する．

よって，$\alpha_{Path}(\omega_1) \neq \alpha_{Path}(\omega_2)$ である．　□

3.3.3 反例の候補

前節で定義したパスとアドバサリの対応関係を用いて，反例の候補を定義する．

定義 3.7（反例の候補）．G と到達可能性問題 (λ, L_e) において，G の意味 $\mathcal{M}$ の抽象モデル $\mathcal{M}^\sharp$ の反例の候補とは，$l_e \in L_e$ を持つ $\mathcal{M}^\sharp$ 上の集合を $S^\sharp_e \subseteq S^\sharp$ として，

$$\Sigma_{\omega^\sharp \in \{\omega'^\sharp \in Path^{A^\sharp}_{fin} | last(\omega'^\sharp) \in S^\sharp_e\}} Prob^{A^\sharp}_{fin}(\omega^\sharp) > \lambda$$

となる，$\mathcal{M}^\sharp$ のアドバサリ $A^\sharp$ とパス集合 $\Omega^\sharp$ の組である．反例 (A, Ω) が反例の候補 $(A^\sharp, \Omega^\sharp)$ に対応しているとは，以下の関係を満たしているときである．

$$\alpha_{Adv}(A) = A^\sharp \wedge \forall \omega \in \Omega. \exists \omega^\sharp \in \Omega^\sharp. \alpha_{Path}(\omega) = \omega^\sharp$$

□

以降，

$$\Sigma_{\omega^\sharp \in \{\omega'^\sharp \in Path^{A^\sharp}_{fin} | last(\omega'^\sharp) \in S^\sharp_e\}} Prob^{A^\sharp}_{fin}(\omega^\sharp)$$

を単に

$$Prob^{A^\sharp}_{fin}(S^\sharp_e)$$

と表記する．

ここで反例と反例の候補に関する 2 つの定理を示す．

定理 3.3（合計到達確率の関係）．$\mathcal{M}$ 上のアドバサリ A と $\mathcal{M}^\sharp$ 上のアドバサリ $A^\sharp$ が対応している場合，それぞれの到達確率について以下が成り立つ．

$$Prob^A_{fin}(S_e) \leq Prob^{A^\sharp}_{fin}(S^\sharp_e)$$

$Prob^A_{fin}(S_e)$ は，アドバサリ A における $\mathcal{M}$ での目的状態集合 S_e への到達確率である．

証明． あるアドバサリ A によって導出されるパス集合を $Path^A_{fin}$ とする．また，A に対応する $\mathcal{M}^\sharp$ 上のアドバサリを $A^\sharp$ とする．

A によって導出された，目的状態へ到達可能なパス $\omega \in \{\omega' \in Path^A_{fin} | last(\omega') \in S_e\}$ を考える．

定理 3.1 より，このような ω には対応する $\omega^\sharp$ が存在し，それらの到達確率は等しい．また，定理 3.2 より，ω に対応する抽象パス $\omega^\sharp$ は唯一である．

このことから，目的状態へ到達可能なパス $\omega \in \{\omega' \in Path^A_{fin} | last(\omega') \in S_e\}$ には等しい到達確率で対応する抽象パスがそれぞれ 1 つ存在するため，それら各パスの到達確率の合計は必ず等しくなる．この到達確率の合計は $Prob^{A^\sharp}_{fin}(S^\sharp_e)$ と等しい．また，このような抽象パスの集合を $\Omega^\sharp$ とする．

ところが，$Path^A_{fin}$ から $Path^{A^\sharp}_{fin}$ へは全単射とは限らない．そのため，目的状態へ到達可能な抽象パス $\omega^\sharp \in Path^{A^\sharp}_{fin} \setminus \Omega^\sharp$ が存在する場合がある．このような $\omega^\sharp$ が存在する場合，抽象モデル上の到達確率である $Prob^{A^\sharp}_{fin}(S^\sharp_e)$ が $Prob^{A^\sharp}_{fin}(S^\sharp_e)$ より大きくなる．

したがって

$$Prob^A_{fin}(S_e) \leq Prob^{A^\sharp}_{fin}(S^\sharp_e)$$

□

定理 3.3 は，$\mathcal{M}$ において到達確率が最大となるアドバサリを考えたとき，$\mathcal{M}$ の最大到達確率が抽象モデル上の最大到達確率以下になることを示す．

定理 3.4（反例の存在）．$\mathcal{M}$ で反例 (A, Ω) が存在するならば，必ず $\mathcal{M}^\sharp$ で反例の候補 $(A^\sharp, \Omega^\sharp)$ が存在する．

証明． 定理 3.1 より，$A^\sharp = \alpha_{Adv}(A)$ となる $A^\sharp$ が存在する．定義 2.8（反例の定義）より，Ω の到達確率の合計は λ より大きい．また，定理 3.1 より，Ω に対応する $\Omega^\sharp = \{\omega^\sharp \in Path^{A\sharp}_{fin} | \exists \omega \in \Omega. \omega^\sharp = \alpha(\omega)\}$ が存在する．

定理 3.3 より，これらの到達確率の合計は等しくなるので，$\Omega^\sharp$ の合計到達確率は λ より大きい．つまり，$(A^\sharp, \Omega^\sharp)$ は反例の候補となりうる．

したがって，$\mathcal{M}$ で反例 (A, Ω) が存在するならば，必ず $\mathcal{M}^\sharp$ で反例の候補 $(A^\sharp, \Omega^\sharp)$ が存在する．　□

この定理は抽象モデルで反例の候補が存在しなければ反例が存在しないことを示している．よって，抽象モデルで反例の候補が存在しないことが分かれば，確率到達可能性問題に対して “*yes*” ということができる．

3.4　同時実行

ここで，確率時間オートマトンの同時実行性の問題について述べる．

定理 3.4 では，反例が存在するならば，反例の候補も存在することを述べたが，反例の候補が存在するならば反例が存在するとは限らない．そこで，反例の候補 $(A^\sharp, \Omega^\sharp)$ から対応する (A, Ω) を求める反例解析を 5 章で提案する．しかしその際，$\Omega^\sharp$ のそれぞれのパス対応するパスの集合 Ω を求めて反例とするだけでは不十分である．これは到達するパスを求めただけでは，同時実行不可能である可能性があるためである．

このことを**図 4** の確率時間オートマトン G_2 と，**図 5** の抽象化述語 $\mathbf{\Psi}$（$\Psi^{l_0} = \{\text{true}\}$, $\Psi^{l_1} = \{\text{true}\}$, $\Psi^{l_2} = \{\text{true}\}$, $\Psi^{l_e} = \{\text{true}\}$）による抽象モデル $\mathcal{M}_0^\sharp$ を用いて例を示す．なお，検証する到達可能性問題は $(0.5, \{l_e\})$ とする．

$\mathcal{M}_1^\sharp$ から反例の候補のパス集合 $\Omega^\sharp$ として，2 つのパス $\omega_1^\sharp = s_0^\sharp \to s_1^\sharp \to s_e^\sharp$ と $\omega_2^\sharp = s_0^\sharp \to s_2^\sharp \to s_e^\sharp$ があげられたとする．G_2 の $\mathcal{M}$ 上において，この 2 つのパスに対応するパスはそれぞれ存在する．$\omega_1^\sharp$ に対応するパス ω_1 では，l_0 において 1 単位時間以上の時間遷移をしなければ l_e に到達できない．一方 $\omega_2^\sharp$ に対応するパス ω_2 では，l_0 において時間遷移をしていては l_e に到達することはできない．初期状態 (l_0, ν_0) で ω_1 では時間遷移をし，ω_2 では離散遷移をする．つまり，それぞれ選択する動作が異なり，異なるアドバサリで動作している．よって，$(A^\sharp, \Omega^\sharp)$ から Ω だけでなく A の対応も調べる必要がある．本研究では，後の反例解析で，同時実行反例解析を提案することでこの問題を解決する．

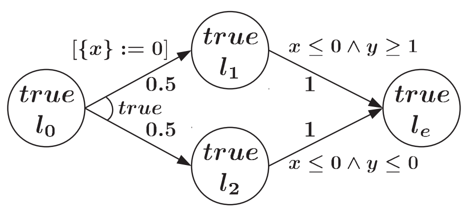

図 4　確率時間オートマトン G_2
Fig. 4　Probabilistic timed automaton G_2.

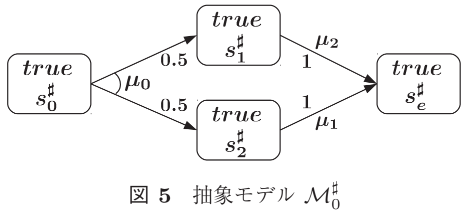

図 5　抽象モデル $\mathcal{M}_0^\sharp$
Fig. 5　Abstract model $\mathcal{M}_0^\sharp$.

4.　確率時間 CEGAR

述語抽象化および反例による精錬を自動的に検証に適用していくアプローチが CEGAR（反例による抽象化と精錬）[7] の枠組みである．ここまでに説明した抽象化を CEGAR に適用した確率時間 CEGAR の検証の流れを**図 6** に従い説明する．

(1)　抽象化（Abstraction）：述語集合 $\mathbf{\Psi}$ から抽象モデル $\mathcal{M}^\sharp$ を計算する．述語集合 $\mathbf{\Psi}$ は，$\forall l \in L.\Psi^l = \{\text{true}\}$ として開始する．

(2)　反例の候補の導出（Candidate Computation of Counterexample）：$\mathcal{M}_{\mathbf{\Psi}}^\sharp$ 上で反例の候補 $(A^\sharp, \Omega^\sharp)$ を求める．反例の候補が存在しない場合，“*yes*” を出力し検証を終了する．

(3)　反例解析（Counterexample Analysis）：反例の候補 $(A^\sharp, \Omega^\sharp)$ に対応する反例 (A, Ω) が存在するかどうかを求める．存在すれば “*no*” を出力し，検証を終える．存在しなければ，$(A^\sharp, \Omega^\sharp)$ は具体モデルで実現できないと判断する．このような反例の候補を偽反例とする．

(4)　精錬（Refinement）：反例解析の結果，偽反例であればその偽反例 $(A^\sharp, \Omega^\sharp)$ を $\mathcal{M}^\sharp$ から取り除くための新しい述語を導出し，新しい $\mathbf{\Psi}$ を得る．

(5)　(1) に戻る．

これらのサイクルを繰り返していくことにより，最終的にシステムが確率到達可能性問題に対し “*yes*” か “*no*” か

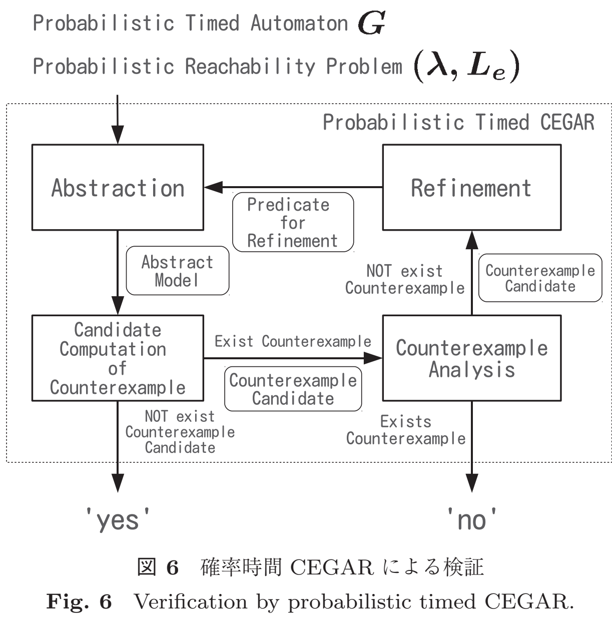

図 6　確率時間 CEGAR による検証
Fig. 6　Verification by probabilistic timed CEGAR.

を判定する．

この確率時間 CEGAR による検証を実現するためには，以下の手法が必要になる．

- $\mathcal{M}^\sharp$ から反例の候補 $(A^\sharp, \Omega^\sharp)$ を導出する手法
- 反例の候補 $(A^\sharp, \Omega^\sharp)$ から対応する反例 (A, Ω) が存在するか求める反例解析手法
- $\mathcal{M}^\sharp$ から $(A^\sharp, \Omega^\sharp)$ を取り除く述語 Ψ を導出する精錬手法

ここで，確率時間 CEGAR の定理を述べる．

定理 4.1（検証の正当性）．この検証によって求めた解は必ず時間確率システムの解と一致する．

証明．“*yes*” と解を出した場合は，いかなる述語集合による $\mathcal{M}^\sharp$ であっても，定理 3.4 より，$\mathcal{M}$ の解と一致する解である．“*no*” と解を出した場合は，反例の候補より $\mathcal{M}$ 上の反例を導出しているため，$\mathcal{M}$ の解と一致する． □

5. 反例解析

本章では，確率時間 CEGAR における反例解析の手法を提案する．反例解析は以下の 2 つの手順で行われる．

(1) 反例の候補の導出：$\mathcal{M}^\sharp$ に反例の候補が存在するか否かを求め，適当な反例の候補 $(A^\sharp, \Omega^\sharp)$ を導出する．

(2) 反例解析：$(A^\sharp, \Omega^\sharp)$ から対応する反例 (A, Ω) が存在するかどうか求める．

本章では上記の 2 つの手順についてそれぞれアルゴリズムを提案する．

5.1 反例の候補の導出

(1) の反例の候補を導出する手順を以下に示す．

手順 1. アドバサリを導出：検証確率 λ より大きい到達確率になるアドバサリ $A^\sharp$ を求める．この $A^\sharp$ が存在しなければ反例の候補は存在しない．

手順 2. パス集合を導出：$A^\sharp$ におけるパスのうち，パスの合計確率が，検証確率 λ より大きくなるような，有限長のパスからなる有限集合 $\Omega^\sharp$ を求める．

手順 1. によって検証確率を超える到達確率になるアドバサリ $A^\sharp$ が求まれば，定理 2.1 より必ずパスの集合 $\Omega^\sharp$ は存在し，手順 2. によりそのパス集合を求めることができる．この 2 つの手順により，反例の候補 $(A^\sharp, \Omega^\sharp)$ を導出する．また，ここでは以下の 2 つのテクニックを用いる．

a. $A^\sharp$ として，最大到達確率になるシンプルなアドバサリを用いる．

b. $\Omega^\sharp$ として，パスの数の少ない最小反例 [9] を用いる．

まず，a. について説明する．

抽象モデル $\mathcal{M}^\sharp$ は $\mathcal{M}$ を時間について抽象化したマルコフ決定過程であり，反例の候補の導出において，時間に関する制約を考慮する必要がないことに注意する．

$\mathcal{M}$ は時間をともなうマルコフ決定過程であり，その最大到達確率を得られるアドバサリはシンプルとはならないことを 2.4 節で述べた．これは $\mathcal{M}$ ではその遷移関係において時間に関する制約を考慮する必要があるためである．一方，$\mathcal{M}^\sharp$ は時間について抽象化されており，遷移関係における時間に関する制約が抽象化によって取り除かれているため，時間のともなわない一般的なマルコフ決定過程と同様に扱うことができる．一般的なマルコフ決定過程において，最大到達確率を得られるアドバサリはシンプルなアドバサリであり [3]，その最大到達確率は線形計画法によって計算可能であることが知られている [3]．この方法では，すべての状態から目的状態集合への最大到達確率を求めることができる．また，その結果から各状態について，最大到達確率を得るためにどの確率分布が選ばれたかを得ることができる．このようにして選ばれた確率分布を返すようなシンプルなアドバサリを考えたとき，それは最大到達確率を返すアドバサリ $A^\sharp$ となる．よって，到達可能性解析ではこのシンプルなアドバサリ 1 つのみについて考えれば十分であり，任意のアドバサリについて考える必要はない．

ここで求めた最大到達確率が検証確率以下であれば反例の候補は存在しないということができる．さらに定理 3.4 より，具体モデルにおいて反例が存在しないといえる．したがって，最大到達確率が検証確率以下であれば確率到達可能性問題に対して “*yes*” と答えることができる．

ここで，$\mathcal{M}^\sharp$ において最大到達確率を与えるシンプルなアドバサリ $A^\sharp$ と，$\mathcal{M}$ において最大到達確率を与えるアドバサリ A の間には対応関係がないことに注意する．また，$\mathcal{M}$ において最大到達確率を与えるアドバサリ A はシンプルではないことが知られている [16]．それにもかかわらず，ここでは $A^\sharp$ にシンプルなアドバサリを選んでいる．これは，定理 3.3 に基づいて検証確率と比較するために，抽象モデル $\mathcal{M}^\sharp$ における最大到達確率を得る必要があるためである．

次に，b. について説明する．$A^\sharp$ によって $\mathcal{M}^\sharp$ の非決定が解決されると，その動作は離散時間マルコフ連鎖として記述される．そこで，文献 [9] によって離散時間マルコフ連鎖の反例として提案され，確率 CEGAR [11] でも利用されている，最小反例（Smallest Counterexample）を，本手法においても利用する．

定義 5.1（最小反例 [9]）．離散時間マルコフ連鎖 $MC = (S, s_0, \mathbf{P})$ と目的状態 $S_e \subseteq S$，検証確率 λ において，最小反例 $\Omega_{smallest}$ とは，合計到達確率が検証確率 λ より大きいパスの集合の中で，最も要素数が少なく，さらにその中で合計到達確率が最も大きいものをいう．

□

この最小反例の採用により，後述する反例解析の手法において，以下の 2 つの効率化が期待できる．

- パス集合の要素数を少なくすることで，反例解析に用いるパスの数を抑える．

- 合計到達確率を大きくすることで，有力な到達確率の大きいパスを優先して解析する．

最小反例は *k-shortest path* アルゴリズムを用いることで導出可能である．このアルゴリズムは，目的状態集合へ到達可能なパスのうち，その到達確率が大きいものから順に k 本のパスを導出するアルゴリズムである．定義より，最小反例は目的状態集合へ到達可能なパスのうち，合計到達確率が λ より大きくなるまで，到達確率が大きいパスから順に含むことは自明である．よって，次のような手法で最小反例を導出可能である．

(1)　$k = 1$ とする．

(2)　*k-shortest path* アルゴリズムを用いてパス集合を導出する．

(3)　その合計到達確率が λ 以下となった場合は k を 1 増やして (2) に戻る．合計到達確率が λ を上回った場合，そのパス集合を最小反例とする．

この手順において，合計到達確率が λ を上回るパス集合が導出されたとき，そのパス集合は到達確率が大きいパスから順に，λ を上回るために必要な最も要素数が少ないパス集合となっている．したがって，導出されたパス集合は最小反例となる．

ここで述べた，$\mathcal{M}^\sharp$ において最大到達確率を与えるアドバサリ $A^\sharp$ と，この $A^\sharp$ で離散時間マルコフ連鎖を構築したときの最小反例 $\Omega^\sharp$ の組 $(A^\sharp, \Omega^\sharp)$ を反例の候補として利用する．

5.2　反例解析

本節ではまず反例解析の概略を説明し，2 つの手順からなる反例解析のアルゴリズムを提案する．また，解析の結果から偽反例であることが分かった場合，次章で説明する精錬において必要な情報についても説明する．

5.2.1　反例解析の概要

(2) の反例解析では，前節で求めた $\mathcal{M}^\sharp$ の反例の候補 $(A^\sharp, \Omega^\sharp)$ に対し，対応する反例 (A, Ω) を求める．本手法の反例解析では $(A^\sharp, \Omega^\sharp)$ のうち，パスの集合 $\Omega^\sharp$ のみを用いる．反例解析は以下の 2 つの手順に分かれる．

手順 1.　パス反例解析：$\Omega^\sharp$ の各パス $\omega^\sharp$ に対し，対応する $\mathcal{M}$ の ω を求め，反例となる $\mathcal{M}$ のパスの集合 Ω の集合 $\mathbf{\Omega}$ を得る．

手順 2.　同時実行反例解析：ある 1 つのアドバサリ A によって，手順 1. で求めた $\Omega \in \mathbf{\Omega}$ に含まれるパスすべてを導出できるか確かめる．

まず手順 1. では，$\Omega^\sharp$ のそれぞれのパス $\omega^\sharp$ について，対応する $\mathcal{M}$ 上のパス ω が存在するかどうかを求める．定義 3.5 より，$\mathcal{M}^\sharp$ 上のパス $\omega^\sharp$ を具体化すると，$\mathcal{M}$ 上のパス集合 $\{\omega \in Path_{fin} | \alpha_{Path}(\omega) = \omega^\sharp\}$ となる．もし，$\mathcal{M}$ 上で対応するパスがなければ，$\{\omega \in Path_{fin} | \alpha_{Path}(\omega) = \omega^\sharp\} = \emptyset$ となる．

このようにして得られた各パス集合から，任意のパスをそれぞれ 1 つ選んで集合 Ω とし，これを反例のパスの集合とする．この Ω のすべての組合せからなる集合を $\mathbf{\Omega}$ とする．

ここで，反例の候補のパスのうち，1 つでも $\{\omega \in Path_{fin} | \alpha_{Path}(\omega) = \omega^\sharp\} = \emptyset$ であれば，この反例の候補 $(A^\sharp, \Omega^\sharp)$ は偽反例となることに注意する．$\Omega^\sharp$ は最小反例（定義 5.1）であるため，$\mathcal{M}^\sharp$ において，合計到達確率が検証確率 λ より大きいパスの集合のうち，最も要素数が少ないものであった．そのため，$\Omega^\sharp$ のうち 1 つでも $\mathcal{M}$ 上に対応するパスが存在しないならば，残りのパスの合計到達確率は検証確率 λ 以下となるため，偽反例であることが分かる．

次の手順 2. では，求めたパスの集合 $\Omega \in \mathbf{\Omega}$ が 1 つのアドバサリ A で構成可能かどうかを確かめる．つまり，$\exists \Omega \in \mathbf{\Omega}. \exists A \in Adv. \Omega \subseteq Path^{A}_{fin}$ を確かめる．このような Ω および A が存在しない場合，この反例の候補は偽反例であることが分かる．

手順 1. と手順 2. のそれぞれの反例解析によって，導出した $(A^\sharp, \Omega^\sharp)$ が偽反例であると判断された場合，同じ反例の候補を導出しないように，新しい述語を追加して抽象モデルの精錬を行う．

手順 1. で偽反例と判断されれば，$\mathcal{M}$ 上で対応するパスがない $\omega^\sharp$ を $\mathcal{M}^\sharp$ から取り除くように精錬する．手順 2. で偽反例と判断されれば，$\mathcal{M}$ 上では 1 つのアドバサリ A で構成できない $\omega^\sharp$ が $\mathcal{M}^\sharp$ 上でも 1 つのアドバサリ A で構成できないように精錬する．精錬については 6 章で説明する．

各手順で求められたパス集合 Ω とアドバサリ A を反例とし，反例が求まれば確率到達可能性問題に対して，“*no*” という解を示すことができる．このように，本手法における反例解析は，反例の候補のパスの集合のみを用いて解析する．

5.2.2　準備（ゾーンによる解析）

この反例解析の手法では，$\omega^\sharp$ に対応するパス集合 $\{\omega | \alpha_{Path}(\omega) = \omega^\sharp\}$ を，$\omega^\sharp$ の各抽象状態に対応するゾーンとすることで取り扱う．つまり，ある $\omega^\sharp$ 中の抽象状態 $s^\sharp$ について，そのゾーンに着目し，以下のように取り扱う．

$$\{\nu | \alpha_{Path}(\omega) = \omega^\sharp \wedge \exists i.(\alpha(\omega(i)) = s^\sharp \wedge \omega(i) = (l, \nu))\}$$

定義 3.5 の (2) より，$\omega^\sharp$ に現れない時間遷移を考慮する必要があることに注意する．そのため，集める各抽象状態のゾーンを，時間遷移前のゾーンである到達条件と，時間遷移後のゾーンである出発条件の 2 つとする．$\omega^\sharp$ の i 番目の状態 (l, b) の出発条件を $\zeta^{dep}_{i,\omega^\sharp}$，到達条件を $\zeta^{rea}_{i,\omega^\sharp}$ とし，この概略を図 **7** に示す．$\omega^\sharp$ から ω を導出するために，$\omega^\sharp$ の各状態について出発条件と到達条件を求める（導出手法については次節で述べる）．

ここで，反例解析で用いるゾーン演算を定義する．

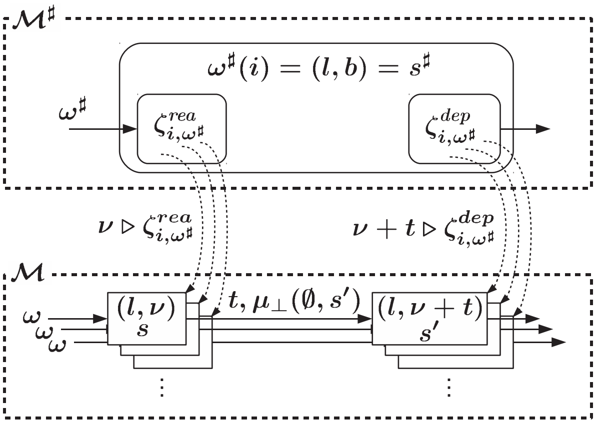

図 7　ゾーンによる到達可能性の解析
Fig. 7　Reachability analysis by zones.

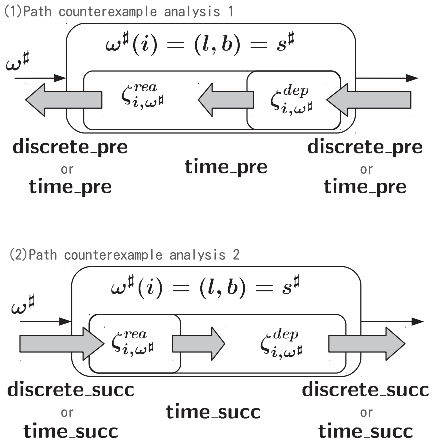

図 8　パス反例解析
Fig. 8　Path counterexample analysis.

定義 5.2（ゾーン演算）．時間確率システムのゾーンを $\zeta \in Zones(C)$，確率遷移関係を $(l, \zeta_g, p) \in prob$，リセットクロック変数の集合を $X \subseteq C$ として，ゾーンを変形する以下の演算を定義する．

$$\mathsf{time\_succ}[\zeta] = \{\nu | \exists t \in \mathbb{R}^{\geq 0}. \nu - t \triangleright \zeta\}$$
$$\mathsf{time\_pre}[\zeta] = \{\nu | \exists t \in \mathbb{R}^{\geq 0}. \nu + t \triangleright \zeta\}$$
$$\mathsf{reset}[\zeta, X] = \{\nu[X := 0] | \nu \triangleright \zeta\}$$
$$\mathsf{free}[\zeta, X] = \{\nu | \nu[X := 0] \triangleright \zeta\}$$
$$\mathsf{discrete\_succ}[\zeta, \zeta^g, X] = \mathsf{reset}[\zeta \wedge \zeta^g, X]$$
$$\mathsf{discrete\_pre}[\zeta, \zeta^g, X] = \mathsf{free}[\zeta, X] \wedge \zeta^g$$

ここで $(\nu - t)$ は，すべてのクロック $x \in C$ について $(\nu - t)(x) = \nu(x) - t$ となるクロック評価であることに注意する（定義 2.2）．また，入力するゾーンがクロック評価の空間の凸部分集合の場合，演算結果も凸部分集合になる．□

time_succ[ζ]，discrete_succ[ζ, ζ^g, X] 演算は，あるゾーン ζ から時間遷移，または離散遷移によって到達可能な状態集合を得る演算である．

time_pre[ζ]，discrete_pre[ζ, ζ^g, X] 演算は，あるゾーン ζ へ時間遷移，または離散遷移によって到達可能な状態集合を得る演算である．状態集合のうち，各状態のクロック評価のみにに着目することで，それをゾーンとして扱うことができる．そのため，これらのゾーン演算によって，到達可能な状態集合を求めることができる [2]．

time_succ，time_pre，discrete_succ，discrete_pre についてそれぞれ説明する．

- time_succ[ζ]，time_pre[ζ]：時間遷移の演算である．
  time_succ[ζ] はあるゾーン ζ から時間遷移可能なゾーンを返す．
  time_pre[ζ] はあるゾーン ζ へ時間遷移可能なゾーンを返す．この演算では時間遷移量を考慮せず，時間遷移可能なゾーンすべてを得る．
- discrete_succ[ζ, ζ^g, X], discrete_pre[ζ, ζ^g, X]：離散遷移の演算である．
  discrete_succ[ζ, ζ^g, X] は，ガード条件が ζ^g，リセットクロック集合が X である離散遷移によって，あるゾーン ζ からガード条件およびリセットクロックを考慮して遷移可能なゾーンを返す．
  discrete_pre[ζ, ζ^g, X] は同様の離散遷移によって，あるゾーン ζ へ遷移可能なゾーンを返す．

以後，discrete_succ，discrete_pre 演算を $\mathcal{M}^\sharp$ 上のパス $\omega^\sharp$ の遷移 $(l, b) \xrightarrow{\mu^\sharp(X,(l',b'))} (l', b')$ に対して用いた場合, 定義 2.5 と定義 3.2 によって，その演算に用いる ζ_g は，$\mu^\sharp$ を含む状態遷移関係 $((l, b), \mu^\sharp) \in prob^\sharp$ に対応する G の確率遷移関係 (l, ζ_g, p) の ζ_g を用いる．

5.2.3　パス反例解析

パス反例解析の概略を図 8 に示す．パス反例解析では，反例の候補の $\Omega^\sharp$ から $\boldsymbol{\Omega}$ を求めるために，各抽象パス $\omega^\sharp \in \Omega^\sharp$ 上の，各抽象状態の到達条件と出発条件を以下の手順で求める．

手順 1.　目的状態に到達可能な到達条件と出発条件を，$\omega^\sharp$ の目的状態から time_pre または discrete_pre 演算を用いて後方から求める（図 8 の (1)）

手順 2.　求めた到達条件と出発条件上で，初期状態から到達可能な到達条件と出発条件を，$\omega^\sharp$ の初期状態から time_succ または discrete_succ 演算を用いて前方から求める（図 8 の (2)）

$\omega^\sharp(i)$ を $\omega^\sharp$ の i 番目の抽象状態 $(l_{\omega^\sharp,i}, b^{\omega^\sharp,i})$ とし，このロケーション $l_{i,\omega^\sharp}$ での G での不変条件を $Inv(l_{i,\omega^\sharp})$ とする．また，i 番目の遷移のクロック変数のリセットを $X_{i,\omega^\sharp}$ とし，この遷移に対応する G でのガード条件を $\zeta^g_{i,\omega^\sharp}$ と表現する．

まず，手順 1. の目的状態からの到達条件と出発条件の求め方について説明する．$\omega^\sharp$ の i 番目の状態 $\omega^\sharp(i)$ の出発条件 $\zeta^{dep}_{i,\omega^\sharp}$ は，先の状態の到達条件 $\zeta^{rea}_{i+1,\omega^\sharp}$ に対し，離

散遷移であれば discrete_pre 演算を，時間遷移であれば time_pre 演算を用いて求める．到達条件 $\zeta^{rea}_{i,\omega^\sharp}$ は，$\zeta^{dep}_{i,\omega^\sharp}$ に対し，time_pre 演算を用いて求めることができる．これは定義 3.5 に基づき，$\omega^\sharp(i)$ 内で $\zeta^{rea}_{i,\omega^\sharp}$ から $\zeta^{dep}_{i,\omega^\sharp}$ へ時間遷移があったと考えるためである．なお，目的状態の到達条件 $\zeta^{rea}_{|\omega^\sharp|,\omega^\sharp}$ はこの状態の不変条件 $Inv(l_{|\omega^\sharp|,\omega^\sharp})$ とする．これにより，各状態の到達条件，出発条件は目的状態に到達可能な最大の状態集合となる．

もし $\omega^\sharp(i)$ の出発条件 $\zeta^{dep}_{i,\omega^\sharp}$ が false になった場合，この $\omega^\sharp$ 上の $\omega^\sharp(i)$ に対応する状態集合からは目的状態に到達できないことを示している．このような $\omega^\sharp$ は $\mathcal{M}$ 上には対応するパスが存在しないとして偽反例であると判断し，この情報を用いて $\mathcal{M}^\sharp$ を精錬する．具体的には，

- $l\ s.t.\ \omega^\sharp(i+1) = (l, b)$
- $\zeta^{rea}_{i+1,\omega^\sharp}$
- 時間遷移：time_succ($b^{i,\omega^\sharp}\Psi^{l_{i,\omega^\sharp}}$)
  離散遷移：discrete_succ($b^{i,\omega^\sharp}\Psi^{l_{i,\omega^\sharp}}, \zeta^g, X$)

の情報が必要になる．これらを用いてどのように精錬するかについては，6.1.1 項で詳しく説明する．

このように，パスの最初の抽象状態 $\omega^\sharp(0)$ における到達条件 $\zeta^{rea}_{0,\omega^\sharp}$ を求めるまで，この手順を繰り返す．

ここで，初期状態のクロック評価は必ず ν_0 であることに注意する．つまり，$\zeta_0 = \{\nu_0\} \subseteq \zeta^{rea}_{0,\omega^\sharp}$ でなければならない．$\zeta_0 \subseteq \zeta^{rea}_{0,\omega^\sharp}$ を確かめるために，$\zeta^{rea}_{0,\omega^\sharp}$ と ζ_0 の積をとって，新たな $\omega^\sharp(0)$ の到達条件 $\zeta^{rea}_{0,\omega^\sharp}$ として更新する．もし，この更新された $\omega^\sharp(0)$ の到達条件が false になれば，$\omega^\sharp$ によって初期状態 $s_0 = (l_0, \nu_0)$ から目的状態に到達できないことが分かるため，この反例の候補を偽反例とすることができる．精錬では，

- $l_0\ s.t.\ \omega^\sharp(0) = (l_0, b)$
- $\zeta^{rea}_{0,\omega^\sharp}$
- ζ_0

の情報が必要になる．こちらも精錬の詳細については 6.1.2 項で説明する．

このように，手順 1. によって，この抽象パス $\omega^\sharp$ に対応する具体モデル上のパスが必ず存在することが分かった．そこで，手順 2. では，初期状態から後方に向かって，到達可能な出発条件および到達条件を求めることで，この抽象パスを実行可能な，具体モデル上でのパス集合を導出する．

手順 2. では前方から，$\omega^\sharp(i-1)$ の出発条件 $\zeta^{rea}_{i-1,\omega^\sharp}$ に対し，離散遷移であれば discrete_succ，時間遷移であれば time_succ 演算を用いて $\omega^\sharp(i)$ の到達条件 $\zeta^{dep}_{i,\omega^\sharp}$ を求める．出発条件 $\zeta^{rea}_{i,\omega^\sharp}$ は到達条件 $\zeta^{dep}_{i,\omega^\sharp}$ に対して time_succ 演算を用いて求める．

なお，手順 1. によって，$\omega^\sharp$ 上で ζ_0 は目的状態に到達可能であることが分かっているため，ζ_0 を初期状態とする具体モデル上のパスは必ず存在する．すなわち，time_succ または discrete_succ 演算によって到達条件および出発条件が false になることはない．

パス反例解析によって，各抽象パス $\omega^\sharp \in \Omega^\sharp$ について，具体モデル上で実行可能なパス集合 Ω を求めた．なお，手順 2. で求めた出発条件および到達条件は，次の同時実行反例解析に必要となる．

パス反例解析の手順 1. のアルゴリズムを Algorithm 1 に示す．入力は反例の候補の $\Omega^\sharp$ である．偽反例であると判断されれば spurious として，さらに述語を追加するロケーションと分割すべき 2 つのゾーンを返す．すべての $\omega^\sharp \in \Omega^\sharp$ について，具体モデル上に対応するパスが存在することが分かった場合，exist を返す．

以降，$Inv(l_{i,\omega^\sharp}) \wedge b^{i,\omega^\sharp}\Psi^{l_{i,\omega^\sharp}}$ を単に $\zeta_{i,\omega^\sharp}$ と表記する．$\zeta_{i,\omega^\sharp}$ は，抽象パス $\omega^\sharp$ の i 番目の抽象状態について，ロケーションの不変条件と，述語から作られるゾーンとの積を意味する．

パス集合 $\Omega^\sharp$ 中のすべてのパス $\omega^\sharp$ に対し (line:1)，time_pre または discrete_pre 演算によって，目的状態（パスの末尾の状態 $\omega^\sharp(|\omega^\sharp|)$）から (line:2) 初期状態（パスの先頭の状態 $\omega^\sharp(0)$）へ順次 (line:3) 出発条件 $\zeta^{dep}_{i,\omega^\sharp}$ と到達条件 $\zeta^{rea}_{i,\omega^\sharp}$ を求めていく．出発条件 $\zeta^{dep}_{i,\omega^\sharp}$ の計算は，時間遷移 $\stackrel{\mu_\perp(\emptyset,\omega(i+1))}{\longrightarrow}$ と離散遷移 $\stackrel{\mu(X,\omega(i+1))}{\rightarrow}$ で異なる (line:4-14)．このとき，出発条件 $\zeta^{dep}_{i,\omega^\sharp}$ が false であれば，そのパスは偽反例である (line:6-8，11-13)．偽反例となった場合，精錬に必要な情報を *spurious* として出力して終了する (line:7，12)．次に到達条件 $\zeta^{rea}_{i,\omega^\sharp}$ を求める．定義 3.5 の (2) を考慮し，時間遷移で到達可能な状態集合を到達条件 $\zeta^{rea}_{i,\omega^\sharp}$ として求める (line:15)．初期状態の到達条件 $\zeta^{rea}_{0,\omega^\sharp}$ まで求めた後，$\zeta^{rea}_{0,\omega^\sharp}$ に初期状態 $\zeta_0 = \{\nu_0\}$ が含まれているかを確認する (line:17)．初期状態が含まれていなければ，このパスは偽反例であるので，精錬に必要な情報を spurious として出力して終了する (line:18)．反例の候補のパス集合 $\Omega^\sharp$ に含まれるすべてのパスについて，具体モデルにおいて対応するパス集合が存在するとき，exist が出力される．Algorithm 1 で spurious が出力されれば，反例解析を終了して精錬を行う．exist が出力されれば，パス反例解析の手順 2. を行う．

反例解析の手順 2. では，$\omega^\sharp$ に対応するパス集合 Ω の集合である $\mathbf{\Omega}$ を導出する．ここで求めた各抽象状態についての $\zeta^{rea}_{i,\omega^\sharp}$ および $\zeta^{dep}_{i,\omega^\sharp}$ は，次の同時実行反例解析で必要となる．反例解析の手順 2. を Algorithm 2 に示す．

すべてのパスに対して (line:1)，Algorithm 1 とは逆向きに，初期状態から，time_succ または discrete_succ 演算によって，目的状態へ順に (line:3)，出発条件 $\zeta^{dep}_{i,\omega^\sharp}$ と到達条件 $\zeta^{rea}_{i,\omega^\sharp}$ を求める．Algorithm 1 と同様に，到達条件 $\zeta^{rea}_{i+1,\omega^\sharp}$ は，$\omega^\sharp(i)$ と $\omega^\sharp(i+1)$ 間の遷移が，時間遷移と離散遷移のいずれであるかによって，導出に用いる演算が異なることに注意する (line:5-8)．

このように，Algorithm 1，Algorithm 2 によって，抽象

Algorithm 1 パス反例解析 手順 1

for $\omega^\sharp \in \Omega^\sharp$ **do**
$\zeta^{rea}_{|\omega^\sharp|,\omega^\sharp} \leftarrow \zeta_{|\omega^\sharp|,\omega^\sharp}$
for $(i = |\omega^\sharp| - 1, \dots, 0)$ **do**
if $\omega^\sharp(i) \xrightarrow{\mu^\sharp_\perp}$ is a time transition **then**
$\zeta^{dep}_{i,\omega^\sharp} \leftarrow \mathsf{time\_pre}[\zeta^{rea}_{i+1,\omega^\sharp}] \wedge \zeta_{i,\omega^\sharp}$
if $\zeta^{dep}_{i,\omega^\sharp} = \mathrm{false}$ **then**
return spurious $l_{i+1,\omega^\sharp}, \zeta^{rea}_{i+1,\omega^\sharp}, \mathsf{time\_succ}[\zeta_{i,\omega^\sharp}] \wedge \zeta_{i+1,\omega^\sharp}$
end if
else
$\zeta^{dep}_{i,\omega^\sharp} \leftarrow \mathsf{discrete\_pre}[\zeta^{rea}_{i+1,\omega^\sharp}, \zeta^{g}_{i,\omega^\sharp}, X_{i,\omega^\sharp}] \wedge \zeta_{i,\omega^\sharp}$
if $\zeta^{dep}_{i,\omega^\sharp} = \mathrm{false}$ **then**
return spurious $l_{i+1,\omega^\sharp}, \zeta^{rea}_{i+1,\omega^\sharp}, \mathsf{discrete\_succ}[\zeta_{i,\omega^\sharp}, \zeta^{g}_{i,\omega^\sharp}, X_{i,\omega^\sharp}] \wedge \zeta_{i+1,\omega^\sharp}$
end if
end if
$\zeta^{rea}_{i,\omega^\sharp} \leftarrow \mathsf{time\_pre}[\zeta^{dep}_{i,\omega^\sharp}] \wedge \zeta_{i,\omega^\sharp}$
end for
if $\zeta^{rea}_{0,\omega^\sharp} \wedge \zeta_0 = \mathrm{false}$ **then**
return spurious $l_0, \zeta^{rea}_{0,\omega^\sharp}, \zeta_0$
end if
end for
return exist

Algorithm 2 パス反例解析 手順 2

for $\omega^\sharp \in \Omega^\sharp$ **do**
$\zeta^{rea}_{0,\omega^\sharp} \leftarrow \zeta_0$
for $(i = 0, \dots, |\omega^\sharp| - 1)$ **do**
$\zeta^{dep}_{i,\omega^\sharp} \leftarrow \mathsf{time\_succ}[\zeta^{rea}_{i,\omega^\sharp}] \wedge \zeta^{dep}_{i,\omega^\sharp}$
if $\omega^\sharp(i) \xrightarrow{\mu_\perp}$ is a time transition **then**
$\zeta^{rea}_{i+1,\omega^\sharp} \leftarrow \mathsf{time\_succ}[\zeta^{dep}_{i,\omega^\sharp}] \wedge \zeta^{rea}_{i+1,\omega^\sharp}$
else
$\zeta^{rea}_{i+1,\omega^\sharp} \leftarrow \mathsf{discrete\_succ}[\zeta^{dep}_{i,\omega^\sharp}, \zeta^{g}_{i,\omega^\sharp}, X_{i,\omega^\sharp}] \wedge \zeta^{rea}_{i+1,\omega^\sharp}$
end if
end for
end for

パス $\omega^\sharp$ の各状態における到達条件および出発条件を求めることで，$\omega^\sharp$ に対応する具体モデル上でのパス集合を導出できた．この到達条件と出発条件を用いて，次の同時実行反例解析を行う．

5.2.4　同時実行反例解析

パス反例解析（5.2.3 項）によって，各抽象パス $\omega^\sharp$ に対応する具体モデル上のパス集合 Ω の集合 $\mathbf{\Omega}$ を求めた．

同時実行反例解析では，各 Ω から任意のパス $\omega \in \Omega$ を 1 つずつ選び出して新たな集合 Ω' とし，Ω' に含まれるすべてのパスを導出可能なアドバサリ A が存在するかを確かめる．以下にその形式的な記述を示す．

具体モデル上のパス集合 Ω に，それぞれ添字を付与する．

$$\mathbf{\Omega} = \{\Omega_0, \Omega_1, \cdots, \Omega_{|\Omega^\sharp|-1}\}$$

それらの直積集合の任意の元である，ある列

$$\Omega' = (\omega_0, \omega_1, \cdots, \omega_{|\mathbf{\Omega}|-1}) \in \prod_{i=0}^{|\mathbf{\Omega}|-1} \Omega_i$$

について，

$$\exists A.(\forall i \in \mathbb{N}. 0 \leq i \leq |\mathbf{\Omega}| - 1. \omega_i \text{ は } A \text{ によって導出可能})$$

を確かめる．このような列 $\Omega' = (\omega_0, \omega_1, \cdots, \omega_{|\mathbf{\Omega}|-1})$ について，その列要素を元とする集合を $\Omega = \{\omega_0, \omega_1, \cdots, \omega_{|\mathbf{\Omega}|-1}\}$ とする．構成可能なアドバサリ A が見つかった場合，そのような A とパス集合 Ω の組を反例 (A, Ω) として示すことができる．

では具体的に，そのようなアドバサリ A およびパス集合 Ω の存在を確かめる手法について述べる．

定義 2.6 より，アドバサリはパスを入力することで，次の遷移における時間遷移量と確率分布を返すものであった．よって，あるアドバサリ A で導出可能なパス集合 Ω に含まれる任意の 2 つのパス $\omega_1, \omega_2 \in \Omega$ について，それぞれのプレフィックス ω'_1，ω'_2 が $\omega'_1 = \omega'_2$ であるとき，ω'_1，ω'_2 からの時間遷移量と確率分布が等しいといえる．すなわち同時実行反例解析では，パス集合 Ω に含まれる各パス $\omega \in \Omega$ について，等しいプレフィックス ω' があるならば，その次の遷移は，同じ時間経過量による時間遷移，または同じ確率分布による離散遷移であることを確かめればよい．

ここで，離散遷移については，同時実行が可能であるかについて考慮する必要がないことに注意する．離散遷移の場合は，反例の候補のパスの集合 $\Omega^\sharp$ は抽象モデル上の反例の候補のアドバサリ $A^\sharp$ 上で動作する $\Omega^\sharp \in Path^{A^\sharp}_{fin}$ ため，集合の要素のある 2 つのパス $\omega_1, \omega_2 \in \Omega$ のプレフィックス ω'_1，ω'_2 が $\alpha_{Path}(\omega'_1) = \alpha_{Path}(\omega'_2)$ である場合，ω'_1 の次の遷移が離散遷移ならば ω'_2 の次の遷移もロケーション中の同一の離散遷移である．よって，プレフィックス ω'_1，ω'_2 が $\omega'_1 = \omega'_2$ でその次の遷移が離散遷移ならば等しい確

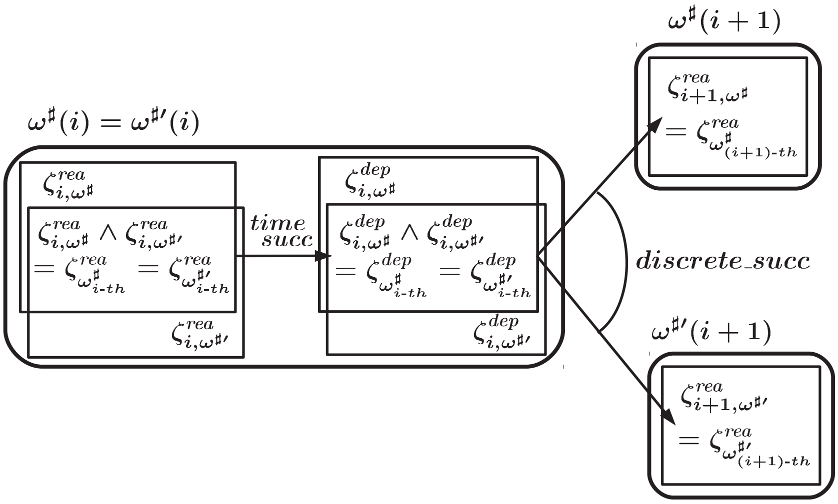

図 9　同時実行反例解析

Fig. 9　Concurrency counterexample analysis.

率分布であるといえる．つまり同時実行反例解析では，時間遷移において同一の時間経過量によって遷移が行えるかについてのみ確認すればよい．

パス反例解析では，各パスについてどのような時間遷移量をとることが可能であるかを，各抽象状態の出発条件および到達条件を求めることによって解析した．

同時実行反例解析では，共通のプレフィックスを持つパスについて，それらパスの各状態の出発条件，到達条件それぞれについて積をとることで，そのプレフィックスの出発条件と到達条件をゾーンとして求める．

このプレフィックスの出発条件が false でなければ，同じプレフィックスを持つパスでは，それらのゾーンに含まれる状態へ共通の時間経過量によって到達可能であることが分かる．したがって，それら共通のプレフィックスを持つすべてのパスで動作可能であるといえる．

なお，到達条件を求める場合は前の抽象状態における出発条件に対し，遷移に対応する time_succ または discrete_succ 演算を，出発条件の場合は同じ抽象状態における到達条件に対して time_succ 演算の結果との積をとることで，初期状態から到達可能な状態集合のみを計算する．この概略を**図 9** に示す．ここでは，$\omega^\sharp_{i\text{-}th}$ を $\omega^\sharp$ の i 番目までの遷移を持つプレフィックスとし，このプレフィックスに対する到達条件を $\zeta^{rea}_{\omega^\sharp_{i\text{-}th}}$，出発条件を $\zeta^{dep}_{\omega^\sharp_{i\text{-}th}}$ とする．

同時実行反例解析では，いずれかのプレフィックスの出発条件または到達条件が false となったとき，複数のパスで共通の時間経過量によって具体モデル上で実行できないとして，反例の候補を偽反例とし，精錬を行う．この精錬の詳細については 6.2 節で説明する．

同時実行反例解析のアルゴリズムを Algorithm 3 に示す．アルゴリズムに対する入力は，パス反例解析の結果得られた $\omega^\sharp \in \Omega^\sharp$ の各抽象状態における到達条件と出発条件である．出力は (A, Ω) が存在していれば exist，偽反例であると分かれば spurious を返す．このアルゴリズムは，Algorithm 2 で求めた，各パスの各抽象状態における到達条件および出発条件を利用し，前方からプレフィックスに対する出発条件と到達条件を time_succ または discrete_succ 演算によって計算する．

Algorithm 3 について説明する．

抽象モデルに存在するすべての抽象パスのプレフィックスに対する出発条件 $\zeta^{dep}_{\omega^\sharp}$，到達条件 $\zeta^{rea}_{\omega^\sharp}$ の初期化を行う．(line:1-4)．初期状態のみからなる長さ 0 のパスについて，到達条件を ζ_0 とする (line:5)．短いプレフィックスから順に (line:6)，到達条件 (line:7-15)，出発条件 (line:8-28) の順で求める．

まず到達条件では，反例の候補の各パス $\omega^\sharp \in \Omega^\sharp$ について (line:7)，プレフィックスの到達条件 $\zeta^{rea}_{\omega^\sharp_{i\text{-}th}}$ とそのパスの到達条件 $\zeta^{rea}_{i,\omega^\sharp}$ の積を求めてプレフィックスを更新し (line:9)，さらにこの到達条件から時間遷移可能な出発条件を求める (line:10)．

次に出発条件では，同様に反例の候補の各パス $\omega^\sharp \in \Omega^\sharp$ について (line:13)，$\zeta^{dep}_{\omega^\sharp_{i\text{-}th}}$ とそのパスの出発条件 $\zeta^{dep}_{i,\omega^\sharp}$ の積を求める (line:15)．

この積が false でなければ $\zeta^{dep}_{\omega^\sharp_{i\text{-}th}}$ を更新し (line:21)，さらにこの出発条件から遷移可能な到達条件を時間遷移，離散遷移を考慮しながら求める (line:19-23)．

もし積が false になれば，同様に偽反例と判断する (line:15-17)．具体的には，

- $l_{i,\omega}$
- $\zeta^{dep}_{\omega^\sharp_{i\text{-}th}}$
- $\zeta^{dep}_{i,\omega^\sharp}$

の情報を用いて精錬を行う (line:16)．

Algorithm 3 では，すべてのプレフィックスの出発条件および到達条件が false とならなければ，ある 1 つのアドバ

Algorithm 3 同時実行反例解析

1: **for** $\omega^\sharp \in Path^\sharp_{fin}$ **do**
2: $\zeta^{rea}_{\omega^\sharp} \leftarrow \text{true}$
3: $\zeta^{dep}_{\omega^\sharp} \leftarrow \text{true}$
4: **end for**
5: $\zeta^{dep}_{\omega^\sharp=s_0} \leftarrow \zeta_0$
6: **for** $i = 0, \cdots, C_{\Omega_{\max}}$ **do**
7: **for** $\omega^\sharp \in \Omega^\sharp$ **do**
8: **if** $|\omega^\sharp| > i$ **then**
9: $\zeta^{rea}_{\omega^\sharp_{i\text{-}th}} \leftarrow \zeta^{rea}_{\omega^\sharp_{i\text{-}th}} \wedge \zeta^{rea}_{i,\omega^\sharp}$
10: $\zeta^{dep}_{\omega^\sharp_{i\text{-}th}} \leftarrow \mathsf{time\_succ}[\zeta^{rea}_{\omega^\sharp_{i\text{-}th}}] \wedge \zeta_{i+1,\omega^\sharp}$
11: **end if**
12: **end for**
13: **for** $\omega^\sharp \in \Omega^\sharp$ **do**
14: **if** $|\omega^\sharp| > i$ **then**
15: **if** $\zeta^{dep}_{\omega^\sharp_{i\text{-}th}} \wedge \zeta^{dep}_{i,\omega^\sharp} = \text{false}$ **then**
16: **return** spurious $l_{i,\omega^\sharp}, \zeta^{dep}_{\omega^\sharp_{i\text{-}th}}, \zeta^{dep}_{i,\omega^\sharp}$
17: **end if**
18: $\zeta^{dep}_{\omega^\sharp_{i\text{-}th}} \leftarrow \zeta^{dep}_{\omega^\sharp_{i\text{-}th}} \wedge \zeta^{dep}_{i,\omega^\sharp}$
19: **if** $\omega^\sharp(i) \xrightarrow{\mu^\sharp_\perp}$ is a time transition **then**
20: $\zeta^{rea}_{\omega^\sharp_{(i+1)\text{-}th}} \leftarrow \mathsf{time\_succ}[\zeta^{dep}_{\omega^\sharp_{i\text{-}th}}] \wedge \zeta_{i+1,\omega^\sharp}$
21: **else**
22: $\zeta^{rea}_{\omega^\sharp_{(i+1)\text{-}th}} \leftarrow \mathsf{discrete\_succ}[\zeta^{dep}_{\omega^\sharp_{i\text{-}th}}, \zeta^{g}_{i,\omega^\sharp}, X_{i,\omega^\sharp}] \wedge \zeta_{i+1,\omega^\sharp}$
23: **end if**
24: **end if**
25: **end for**
26: **end for**
27: **return** exist

サリで実行可能であることが分かる．したがって，exists と出力されれば (line:27)，反例が存在することが分かり，確率到達可能性問題に対して “yes” と答えることができる．

5.3　反例解析と精錬

本章では反例解析の手法を示した．反例解析はパス反例解析（5.2.3 項）および同時実行反例解析（5.2.4 項）によって構成されている．

反例解析を行った結果，反例が存在する場合，確率到達可能性問題に対して “yes” と答えて検証を終えることができる．

反例の候補から反例を導出できない，すなわち反例の候補が偽反例であることが分かるのは，反例解析において以下のいずれかが false になった場合である．

- パス反例解析
  - $\omega^\sharp(i)$ の出発条件 $\zeta^{dep}_{i,\omega^\sharp}$
  - $\omega^\sharp(0)$ の到達条件 $\zeta^{rea}_{0,\omega^\sharp}$
- 同時実行解析
  - プレフィックスの出発条件 $\zeta^{dep}_{\omega^\sharp_{i\text{-}th}}$

反例の候補が偽反例であった場合，精錬を行う必要がある．本章では精錬を行ううえで必要になる情報として，1 つのロケーションと 2 つのゾーンを，これら 3 つのケースについて示した．

次章では，これらの情報を用いて精錬を行う手法を，それぞれのケースについて示す．

6.　精錬

反例解析において，反例の候補が偽反例であることが分かった場合，精錬を行う．精錬では，その偽反例を導出する原因となった抽象パスを持たない抽象モデル $\mathcal{M}^\sharp$ を構築できる新しい述語 Ψ を導出する．

具体的には，ある 1 つのロケーションと，そのロケーションに追加するための 1 つ以上の述語を導出する．述語は，ある 2 つのゾーンを分割可能な述語とする．これは，述語によって分割された抽象状態間に遷移が構築されないようにすることで，原因となった抽象パスを抽象モデルから取り除くためである．

5 章において，反例解析において反例の候補が偽反例であることが分かる場合は 3 つであることを述べた．本章ではそれぞれの場合について，偽反例を導出する原因となった抽象パスと，その抽象パスを取り除くために追加する述語について述べる．

6.1　パス反例解析による精錬

パス反例解析（Algorithm 1）において，ある抽象パス $\omega^\sharp$ が実行不可能であると分かるのは，

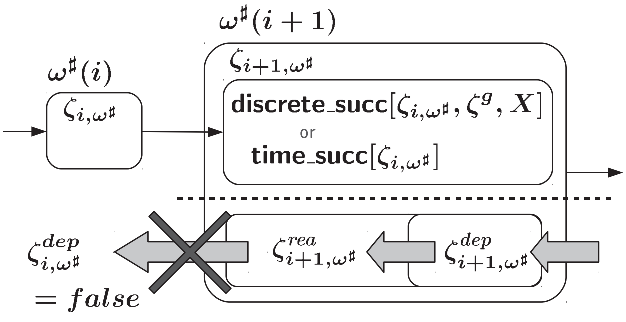

図 10　パス反例解析における精錬

Fig. 10　Refinement by path counterexample analysis.

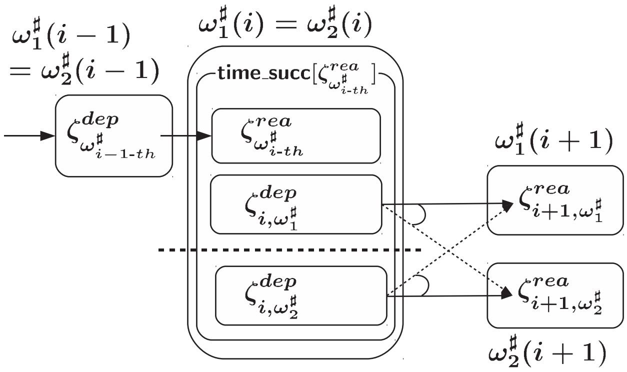

図 11　同時実行反例解析における精錬

Fig. 11　Refinement in concurrency counterexample analysis.

- $\omega^\sharp(i)$ の出発条件 $\zeta^{dep}_{i,\omega^\sharp}$ が false になった場合
- $\zeta^{rea}_{0,\omega^\sharp}$ と ζ_0 の積が false になった場合

のいずれかである．このようなパスが導出された場合，精錬によって取り除く必要がある．

6.1.1　$\omega^\sharp(i)$ の出発条件 $\zeta^{dep}_{i,\omega^\sharp}$ が false になった場合

この概要を図 10 に示す．パス反例解析において，後方解析によって導出される到達条件 $\zeta^{rea}_{i+1,\omega^\sharp}$ と出発条件 $\zeta^{dep}_{i,\omega^\sharp}$ は，目的状態に到達可能な最大の状態集合である．よって，パスが具体モデル上で実行可能であるなら，$\omega^\sharp(i+1)$ の到達条件 $\zeta^{rea}_{i+1,\omega^\sharp}$ は目的状態に到達するクロック評価を含んでいるはずである．したがって，この抽象パスは具体モデルで実行できないことが分かる．

ここで，この抽象パスが導出された原因について考える．抽象モデル構築において $\omega^\sharp(i)$ から $\omega^\sharp(i+1)$ への遷移が構築されるのは，$\omega^\sharp(i)$ に含まれる状態から $\omega^\sharp(i+1)$ に含まれる状態への遷移が存在するためであることに注意する．

つまり，この抽象パスが導出された原因は，$\zeta_{i,\omega^\sharp}$ から $\zeta^{rea}_{i+1,\omega^\sharp}$ へは到達できなかったが，discrete_succ$(\zeta_{i,\omega^\sharp}, \zeta^g, X) \subseteq (\zeta_{i+1,\omega^\sharp} \setminus \zeta^{rea}_{i+1,\omega^\sharp})$ または time_succ$(\zeta_{i,\omega^\sharp}) \subseteq (\zeta_{i+1,\omega^\sharp} \setminus \zeta^{rea}_{i+1,\omega^\sharp})$ へ到達できたためである．いい換えると，$\zeta^{rea}_{i+1,\omega^\sharp}$ と discrete_succ$(\zeta_{i,\omega^\sharp}, \zeta^g, X)$ または time_succ$(\zeta_{i,\omega^\sharp})$ が同じ抽象状態のゾーンに含まれるためである．

よって，この抽象状態のゾーン $\zeta_{i+1,\omega^\sharp}$ を $\zeta^{rea}_{i+1,\omega^\sharp}$ を含むゾーンと discrete_succ$(\zeta_{i,\omega^\sharp}, \zeta^g, X)$ または time_succ$(\zeta_{i,\omega^\sharp})$ を含むゾーンに分割することで，このパスを取り除くことができる．

なお，到達条件 $\zeta^{rea}_{i+1,\omega^\sharp}$ は出発条件 $\zeta^{dep}_{i+1,\omega^\sharp}$ の time_pre 演算の結果であるため，false になることはない．

6.1.2　$\zeta^{rea}_{0,\omega^\sharp}$ と ζ_0 の積が false になった場合

具体モデル上のすべてのパスは ν_0 から始まる．よって，すべての抽象パスは $\zeta_0 = \{\nu_0\}$ を含む抽象状態から始まる．ここで，パス反例解析では目的状態に到達可能な最大の状態集合を導出しており，それぞれの到達条件 $\zeta^{rea}_{i,\omega^\sharp}$ および出発条件 $\zeta^{dep}_{i,\omega^\sharp}$ に含まれる状態のみによって具体モデルで実行可能であることに注意する．

すなわち，このケースでは抽象パス $\omega^\sharp$ の最初の抽象状態 $\omega^\sharp(0)$ の到達条件 $\zeta^{rea}_{0,\omega^\sharp}$ に $\zeta_0 = \{\nu_0\}$ 含まれていないため，具体モデルでは初期状態 ν_0 からこのパスを実行できないことが分かる．具体モデル上のすべてのパスは ν_0 から始まっていなければならないので，このパスは具体モデルで実行できないことが分かる．

この抽象パスが導出された原因は 6.1.1 項と同様に，ζ_0 と $\zeta^{rea}_{0,\omega^\sharp}$ が同じ抽象状態のゾーンに含まれるためである．よって，この抽象状態のゾーン $\zeta_{0,\omega^\sharp}$ を ζ_0 を含むゾーンと $\zeta^{rea}_{0,\omega^\sharp}$ を含むゾーンに分割することで，このパスを取り除くことができる．

6.2　同時実行反例解析による精錬

同時実行反例解析（Algorithm 3）において，反例の候補が偽反例であると分かるのは，プレフィックスの出発条件 $\zeta^{dep}_{\omega^\sharp_{i\text{-}th}}$ が false になる場合である．このようなパス集合が導出された場合，パス集合を構成する一部の抽象パスを精錬によって取り除く必要がある．

同時実行反例解析では，反例の候補であるすべてのパスが同じアドバサリで実行可能であるかを確かめるものであった．このとき，$\zeta^{dep}_{\omega^\sharp_{i\text{-}th}}$ が false になることと，反例の候補であるすべてのパスを同じアドバサリで実行できないことは同値である．図 11 を用いてこれを説明する．

抽象パス $\omega^\sharp_1$ および $\omega^\sharp_2$ はそれぞれの i 番目までの遷移において共通のプレフィックス $\omega^\sharp_{i\text{-}th}$ を持つとし，$i+1$ 番目の遷移において，初めて異なる抽象状態へ遷移するものとする．

パス反例解析の Algorithm 1 および Algorithm 2 によって，$\zeta^{rea}_{i+1,\omega^\sharp_1}$ に遷移可能な状態集合として，$\zeta^{dep}_{i,\omega^\sharp_1}$ を計算した．つまり $\zeta^{dep}_{i,\omega^\sharp_1}$ に含まれる状態からのみ $\zeta^{rea}_{i+1,\omega^\sharp_1}$ に含まれる状態へ遷移できる．$\omega^\sharp_2$ についても同様である．よって，$\zeta^{dep}_{i,\omega^\sharp_1}$ に含まれるが，$\zeta^{dep}_{i,\omega^\sharp_2}$ には含まれない状態からは $\zeta^{rea}_{i+1,\omega^\sharp_2}$ へ遷移できない．

アドバサリは，あるパスを入力することで次の遷移における時間遷移量と確率分布を返すものであった．すなわち，あるアドバサリが存在するとは，すべての状態においてそ

の時間遷移量と確率分布が一意に決まることを意味する．

ところが，$\zeta^{dep}_{i,\omega^\sharp_1}$ と $\zeta^{dep}_{i,\omega^\sharp_2}$ に共通部分がない場合，$\zeta^{rea}_{i+1,\omega^\sharp_1}$ と $\zeta^{rea}_{i+1,\omega^\sharp_2}$ の両方へ遷移可能な状態は存在しない．そのため，図 11 では $\zeta^{dep}_{i,\omega^\sharp_1}$ に含まれる状態から $\omega^\sharp_1$ に対応するパスを導出するためには $\zeta^{rea}_{i+1,\omega^\sharp_1}$ に含まれる状態へ遷移する確率分布を選択しなければならず，$\omega^\sharp_2$ に対応するパスを導出するためには，$\zeta^{dep}_{i,\omega^\sharp_2}$ に含まれる状態へ時間遷移し，そこから $\zeta^{rea}_{i+1,\omega^\sharp_1}$ に含まれる状態へ遷移する確率分布を選択しなければならない．つまり，両方のパスを導出可能な確率分布は存在しないことになる．したがって，これらのパスを導出可能なアドバサリが存在しないことが分かる．

このように，対応するパス集合を同一のアドバサリによって実行できないことが分かった場合，$\zeta^{dep}_{\omega^\sharp_{i\text{-}th}}$ と $\zeta^{dep}_{i,\omega^\sharp}$ を分割することで精錬を行う．なぜなら，これらを分割する述語を加えることで，これまで $\omega^\sharp(i)$ のゾーン内に隠れていた $\zeta^{dep}_{\omega^\sharp_{i\text{-}th}}$ と $\zeta^{dep}_{i,\omega^\sharp}$ 間の時間遷移が抽象モデル上に確率分布として現れるようになるためである（図 11 では，$\zeta^{dep}_{i,\omega^\sharp_1}$ から $\zeta^{dep}_{i,\omega^\sharp_2}$ への遷移が時間遷移としてモデル上に現れた）．よって，反例解析の $A^\sharp$ の導出において，$\zeta^{dep}_{\omega^\sharp_{i\text{-}th}}$ または $\zeta^{dep}_{i,\omega^\sharp}$ を含む抽象状態は，これまで存在した抽象確率分布に加え，この新たに現れた時間遷移の抽象確率分布を選択できるようになり，これらの確率分布からいずれか 1 つが選ばれるため，同様の同時実行の問題は発生しない．

6.3　新たな述語の導出と検証の停止性

確率時間 CEGAR は，抽象化，反例の候補の導出，反例解析，精錬のサイクルを，解が得られるまで繰り返す検証の手法である．ここでは，サイクルごとに新たな述語を導出可能であることと，検証ループの停止性について述べる．

これまでに，ある抽象状態における 2 つゾーンを分割することで，偽反例を導出する原因となったパスを取り除く手法について示した．これらのゾーンを分割する 1 つ以上の述語は任意とし，導出手法としては文献 [15] 等が知られている．本研究で用いる述語導出アルゴリズムは 7 章で示す．

確率時間 CEGAR では次の定理が成り立つ．

定理 6.1（サイクルごとに新たな述語を導出可能）．確率時間 *CEGAR* は，到達可能性問題の解が得られるまで，サイクルごとに新たな述語を導出可能である．

証明． 抽象化の定義 3.2 より，抽象状態 (l, b^l) が持つゾーンは，述語集合 ψ^l と，対応するビットベクトル b^l が持つ各要素の真偽により，すでに分割されている．よって，ψ^l が含む述語によって (l, b^l) が持つゾーンを分割することはできない．つまり，すでに加えられている述語を新たに加えても，これ以上抽象状態を分割できない．これは，ある抽象状態が持つゾーンを 2 つに分割するには，まだ ψ^l に加えられていない述語を用いる必要があることを示している．したがって，CEGAR のサイクルの精錬において，basis に含まれる述語をすべて加えるまで，毎回必ず新たな述語を導出可能である．□

したがって，次の定理が成り立つ．

定理 6.2（検証の停止性）．確率時間 *CEGAR* による検証は有限回数のサイクルで解を得ることができる．

証明． 不変条件と遷移条件で現れる最大整数に対して，basis は有限の述語で構成される [14]．basis に含まれるすべての述語を用いて抽象モデルを構築すると，もとの確率時間オートマトンと等価なリージョングラフ [13] が得られる．したがって，必ず新しい述語が導出される時間確率 CEGAR による検証は有限回数のサイクルで解を得ることができる．□

7.　実験

本章では，確率時間 CEGAR のプロトタイプを Scala によって実装し，実験を行って既存手法とその状態数について比較する．以下に実験環境を示す．

- Intel Core2 Quad CPU Q9650 3.00 GHz
- MemTotal 8,174,136 kB
- CentOS 5.6
- Scala version 2.9.0.1

Scala によるソースコードは 2,000 行程度である．Scala は言語内 DSL の構築が容易なプログラミング言語であり，検証対象となるモデルを記述する言語やパーザ等を新たに実装する必要がなく，Scala のコードとして記述可能であるという点で有効であった．

確率時間 CEGAR で取り扱うゾーンはすべて凸ゾーンであるため，ゾーンの実装には DBM [2] を用いた．

7.1　述語の選択

確率時間 CEGAR を実装するにあたり，精錬における述語選択のアルゴリズムを決定する必要がある．ゾーンの実装として DBM を用いることで，ゾーンによって表される領域の辺を容易に取り扱うことができる．これを利用し，述語の選択は以下のアルゴリズムにより行う．

(1) 分割したい 2 つのゾーンを構成する辺を元とする集合から冪集合を得る．
(2) その冪集合の元のうち，要素が少ないものから順に，2 つのゾーンが分割できるものを探す．

CEGAR における述語抽象化では，分割に用いる述語の増加に応じて抽象モデルの状態数が増える．そのため，このアルゴリズムは，2 つのゾーンを分割可能な述語の組合せのうち，より少ない数の述語で分割しようとするものである．

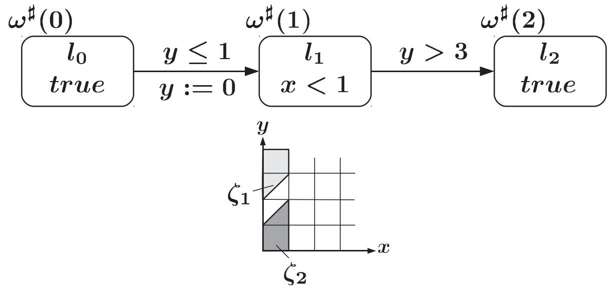

図 **12** 述語とパスの排除 1
Fig. 12 Exclusion of path by predicate 1.

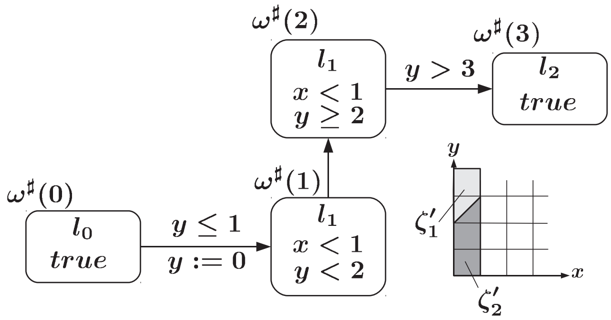

図 **13** 述語とパスの排除 2
Fig. 13 Exclusion of path by predicate 2.

さらに，述語として $x_1 - x_2 \leq d$ や $x_1 - x_2 < d$ のような diagonal 制約を優先して用いる．これは，反例となった抽象パスを次の抽象モデルの再構築で排除するために必要な述語の数を抑えるためである．以下に例を示す．クロック変数 x と y を持つ確率時間オートマトンにおいて，反例として**図 12** のような，長さ 3 の抽象パス $\omega^\sharp$ を考える．

このパスを実行不能であることは自明である．パス反例解析 1 (Algorithm 1) を行うと，$\zeta^{rea}_{1,\omega^\sharp} = x < 1 \wedge x - y < -2$ であり，$\mathsf{discrete\_pre}[\zeta^{rea}_{1,\omega^\sharp}, \zeta^{g}_{1,\omega^\sharp}, X_{1,\omega^\sharp}] = \mathrm{false}$ となるため，具体モデル上で実行できないことが分かり，精錬を行う必要がある．精錬では，$\zeta^{rea}_{1,\omega^\sharp} = x < 1 \wedge x - y < -2 = \zeta_1$ と $\mathsf{discrete\_succ}[\zeta_{0,\omega^\sharp}, \zeta^{g}_{0,\omega^\sharp}, X_{0,\omega^\sharp}] \wedge \zeta_{1,\omega^\sharp} = x < 1 \wedge y - x < 1 = \zeta_2$ を分割する述語を l_1 に追加すればよい（Algorithm 1, line:12）．

ここで，ζ_1 と ζ_2 を分割可能な述語は，$y < 2$，$y - x < 1$，$x - y < -2$ のいずれかである．

これらのうち，まず $y - x < 1$ または $x - y < -2$ で分割した場合を考える．述語を加えた後の抽象モデルの再構築では，l_1 の抽象状態として $(l_1, x < 1 \wedge y - x < 1)$ と $(l_1, x < 1 \wedge x - y \leq -1)$ または $(l_1, x < 1 \wedge x - y < -2)$ と $(l_1, x < 1 \wedge y - x \leq 2)$ が生成される．いずれの場合も，分割後の抽象状態間に時間遷移は構築されない．

次に，$y < 2$ で分割した場合を**図 13** に示す．

述語 $y < 2$ を加えた後の l_1 の抽象状態は $(l_1, x < 1 \wedge y < 2) = \omega^\sharp(1)$ と $(l_1, x < 1 \wedge -y \leq -2) = \omega^\sharp(2)$ になる．ここで，抽象モデル構築において，$\omega^\sharp(1)$ から $\omega^\sharp(2)$ へ時間遷移が構築されてしまうため，到達可能性解析において $\omega^\sharp(0)$ から $\omega^\sharp(3)$ へ到達可能なパスが新たな反例として再び導出されてしまうことに注意する．新たな反例が導出されたため，同様に反例解析 1 を行うと，ζ'_1 と ζ'_2 を分割しなければならないことが分かり，これらを分割可能である述語は diagonal 制約のみである．

このように，述語として diagonal 制約を優先して用いることで，不要な述語の追加を回避し，検証の効率化が期待できる．

7.2 実験モデルの制限

確率時間 CEGAR は，反例解析において前方解析の手法を用いている．DBM のように，モデルに表れる最大定数を用いてゾーンを表現する場合，前方解析が正しく行えないことが知られているが [4]，以下のいずれか 1 つ以上を満たすモデルであれば，前方解析を正しく行えることも知られている [4]．

- $x_1 - x_2 \leq d$ や $x_1 - x_2 < d$ を不変条件や遷移条件に持たない，diagonal-free なモデルである．
- クロック変数の数が 3 以下のモデルである．

そこで本実験では，クロック変数が 3 以下のモデルのみを対象とする．

7.3 FireWire Root Contention Protocol

本実験で扱う IEEE 1394 FireWire Root Contention Protocol は，IEEE 1394 FireWire 規格において，2 つの競合するノードから，一方をリーダとして選出するプロトコルである．本実験ではこのプロトコルに対して，リーダ選出にデッドライン以上の時間を要する場合があるかどうか，という性質について検証を行う．このプロトコルに対する同様の性質についての検証実験は Symbolic Model Checking [13] によってすでに行われており，本実験では同手法との比較を行う．

本実験では，Symbolic Model Checking [13] による検証実験で使用されたモデルを一部を変更し，新たに作成したモデルに対して検証実験を行う．そのモデルを，**図 14** に示す．新たに作成したモデルでは，リーダ選出が完了したことを示すロケーション $elect$ に不変条件 $t \geq D$ を追加している．ここで，t はリーダ選出に要した時間を表現するために，新たに追加したクロック変数である．また，D はデッドラインを表す定数である．したがって，$elect$ へ到達可能であることは，デッドライン以上の時間を要してリーダ選出を行ってしまう場合があることを意味する．

このような変更を加えるのは，Symbolic Model Checking では検証する性質としてデッドラインを指定できるのに対し，本手法はロケーションに対する到達可能性のみを対象としているためである．

なお，既存手法との比較を行うため，デッドラインが 2,000 から 60,000 までのモデルを用意し，それぞれについ

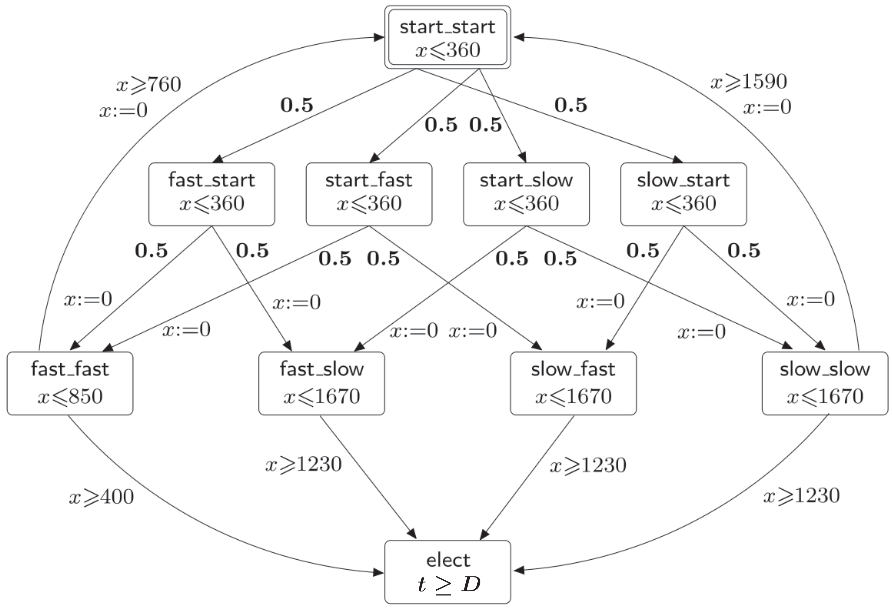

図 14　FireWire root contention protocol

Fig. 14　FireWire root contention protocol.

表 1　確率時間 CEGAR の実験結果：FireWire root contention protocol

Table 1　Result of probabilistic timed CEGAR: FireWire root contention protocol.

D	ループ回数	確率分布数	辺数	述語数	状態数	計算時間 [秒]
2,000	1	12	18	0	10	0.419
4,000	5	21	33	4	14	0.684
6,000	9	30	48	8	18	0.988
8,000	13	39	63	12	22	1.457
10,000	17	48	78	16	26	2.139
20,000	37	93	153	36	46	8.324
30,000	54	135	222	53	63	21.459
40,000	69	175	287	68	78	39.552
50,000	84	215	352	83	93	65.072
60,000	99	255	417	98	108	99.284

表 2　状態数の比較：FireWire root contention protocol

Table 2　Comparison in number of states: FireWire root contention protocol.

D	確率時間 CEGAR の状態数	Symbolic Model Checking の状態数	状態数比
2,000	10	15	0.6667
4,000	14	25	0.5600
6,000	18	47	0.3830
8,000	22	81	0.2716
10,000	26	126	0.2063
20,000	46	528	0.0871
30,000	63	1,206	0.0522
40,000	78	2,168	0.0360
50,000	93	3,426	0.0271
60,000	108	4,964	0.0218

て実験を行う．

このように構成したモデルに対し，Symbolic Model Checking で検証された性質と等価な到達可能性問題を次のように定める．

$$(\lambda, L_e) = (0, \{elect\})$$

すなわち，

$$\forall A.Prob^A(S_e) \leq 0.S_e = \{(elect, \nu) \in S\}$$

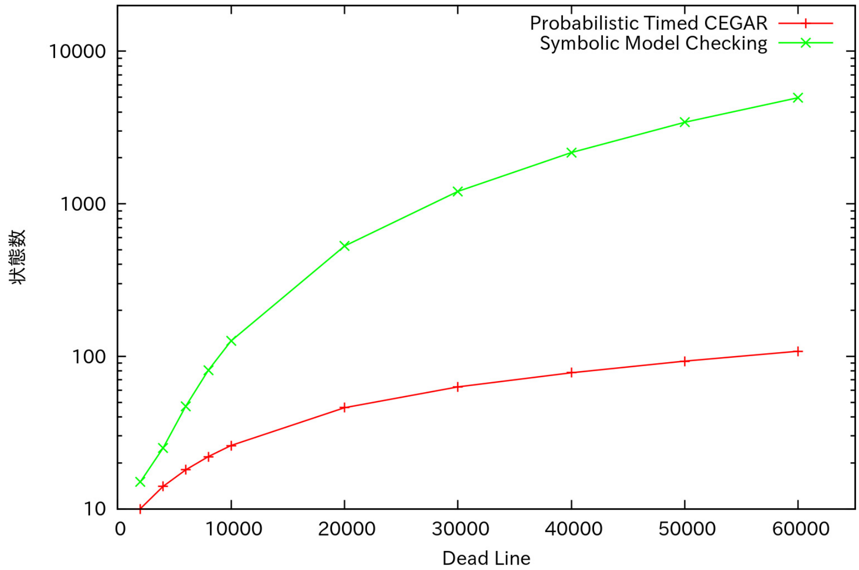

図 **15**　状態数の比較：FireWire root contention protocol

Fig. 15　Comparison in number of states: FireWire root contention protocol.

を満たすかどうかについて検証を行う．この検証結果が“*no*” である場合，すなわち

$$\exists A.Prob^A(S_e) > 0.S_e = \{(elect, \nu) \in S\}$$

が真である場合，リーダ選出にデッドライン以上の時間を要する場合があることが分かる．

7.4　実験結果

表 1 に実験結果を，**表 2** および**図 15** に既存手法との比較結果を示す．

すべての Dead Line について，既存手法よりも少ない状態数で検証を終えている．また，状態数比の推移から，Dead Line の増加にともない，状態数がより削減されていることが分かる．

8.　まとめ

本論文では，確率時間オートマトンの到達可能性解析において，新たに CEGAR による枠組みを適用した検証手法を確立した．本論文の主な貢献は以下の 3 点である．

- 確率時間オートマトンの到達可能性解析において，CEGAR を導入することにより，検証対象に応じた状態空間の構築を可能にした．
- 確率時間オートマトンにおいて，確率分岐による複数パスの同時の実行可能性を同時実行反例解析として判定し，その偽反例による抽象モデルの精錬手法を実現した．
- 本手法を実装して実験を行い，既存手法と比較し，より少ない状態数での検証が可能であることを示した．

今後の課題として，以下のことがあげられる．

- 精錬における述語の導出において，CEGAR で検証を行ううえでより適した述語の導出法の検討
- 本手法をモデル検査に拡張し，PTCTL による到達可能性以外の性質の検証
- より多くのモデルや性質に対する実験

本研究における実装では，7.1 節で示したアルゴリズムにより述語を選択している．しかし，一般に CEGAR における述語の選択はヒューリスティクスであるため，我々の手法が最適であるとは考えにくい．また，diagonal 制約を優先して用いることで効率的にパスを排除できる例を示したが，その有効性について比較実験を行う必要がある．

確率時間オートマトンに対する検証手法である Symbolic Model Checking [13] では，PTCTL（Probabilistic Timed Computation Tree Logic）によって性質を記述したモデル検査が可能であるが，確率時間 CEGAR では PTCTL で表現可能な性質のうち到達可能性のみ検証可能である．一方，CEGAR を用いた PTCTL のサブクラスによる性質について検証可能な手法として，確率時間 WiGAR [17] が提案されている．この手法では PTCTL のサブクラスではあるが，到達可能性だけでなく，いくつかの性質について CEGAR を用いた検証が可能であることが報告されている．

本研究では FireWire Root Contention Protocol の到達可能性問題のみを扱っており，より多くのモデル，条件における実験が必要であると考えられる．特に，確率時間 CEGAR は計算時間を犠牲にして状態数を抑える手法であるため，従来の手法で状態爆発するようなモデルに対し，本手法の有効性を示す必要があると考えられる．

謝辞　トリノ大学の J.Sproston 博士の文献 [16] に関して，我々と有益な議論をしていただいたことを氏に感謝します．また，本研究に関連する，確率時間オートマトンの

CEGAR に関してご協力いただいた，駒形龍太氏（現在，野村総合研究所），陸田陽介氏（現在，NEC），加藤位明氏（現在，PFU）に感謝します．

参考文献

[1] Alur, R., Dang, T. and Ivancic, F.: Predicate abstraction for reachability analysis of hybrid systems, *TECS*, Vol.5, No.1, pp.152–199 (2006).

[2] Bengtsson, J. and Yi, W.: Timed Automata: Semantics, Algorithms and Tools, *LNCS*, Vol.3098, pp.87–124 (2004).

[3] Bianco, A. and de Alfaro, L.: Model checking of probabilistic and nondeterministic systems, *LNCS*, Vol.1026, pp.499–513 (1995).

[4] Bouyer, P.: Untameable Timed Automata!, *LNCS*, Vol.2607, pp.620–631 (2003).

[5] Clarke, E.M., Grumberg, O. and Peled, D.: *Model Checking*, MIT (2000).

[6] Clarke, E.M., Fehnker, A., Han, Z., Krogh, B.H., Ouaknine, J., Stursberg, O. and Theobald, M.: Abstraction and Counterexample-Guided Refinement in Model Checking of Hybrid Systems, *IJFCS*, Vol.14, No.4, pp.583–604 (2003).

[7] Clarke, E.M., Grumberg, O., Jha, S., Lu, Y. and Veith, H.: Counterexample-Guided Abstraction Refinement, *LNCS*, Vol.1855, pp.154–169 (2000).

[8] Graf, S. and Saïdi, H.: Construction of Abstract State Graphs with PVS, *LNCS*, Vol.1254, pp.72–83 (1997).

[9] Han, T. and Katoen, J.P.: Counterexamples in probabilistic model checking, *LNCS*, Vol.4424, pp.72–86 (2007).

[10] Henzinger, T.A., Nicollin, X., Sifakis, J. and Yovine, S.: Symbolic Model Checking for Real-Time Systems, *Information and Computation*, Vol.111, pp.394–406 (1994).

[11] Hermanns, H., Wachter, B. and Zhang, L.: Probabilistic CEGAR, *LNCS*, Vol.5123, pp.162–175 (2008).

[12] Kwiatkowska, M., Norman, G., Segala, R. and Sproston, J.: Automatic verification of real-time systems with discrete probability distributions, *Theor. Comput. Sci.*, Vol.282, No.1, pp.101–150 (2002).

[13] Kwiatkowska, M., Norman, G., Sproston, J. and Wang, F.: Symbolic model checking for probabilistic timed automata, *Information and Computation*, Vol.205, No.7, pp.1027–1077 (2007).

[14] Moller, M.O., Rues, H. and Sorea, M.: Predicate Abstraction for Dense Real-Time Systems, *Electronic Notes in Theoretical Computer Science*, Vol.65, No.6, pp.218–237 (2002).

[15] Rybalchenko, A. and Sofronie-Stokkermans, V.: Constraint solving for interpolation, *J. Symb. Comput.*, Vol.45, No.11, pp.1212–1233 (2010).

[16] Sproston, J.: Strict Divergence for Probabilistic Timed Automata, *LNCS*, Vol.5710, pp.620–636 (2009).

[17] 高橋正樹，清水隆也，山根　智：確率時間 WiGAR による PTCTL サブクラスのモデル検査，数理解析研究所講究録，Vol.1744, pp.25–34, 京都大学 (2011).

清水 隆也

2010 年金沢大学工学部卒業．同年金沢大学大学院自然科学研究科電子情報工学専攻入学，現在も在学．確率リアルタイムシステム等の形式的検証の研究に従事．

森下 篤

2010 年金沢大学大学院自然科学研究科電子情報工学専攻修士課程修了．確率リアルタイムシステム等の形式的検証の研究に従事．

山根 智　（正会員）

1984 年京都大学大学院修了．現在，金沢大学大学院理工研究域電子情報学系教授．博士（京都大学）．リアルタイムハイブリッドシステム等の形式的検証の研究に従事．EATCS，日本ソフトウェア科学会等各会員．