数理解析研究所講究録
第1649巻 2009年 47-54

確率時間CEGAR

金沢大学 森下 篤(Atsushi Morimoto)、駒形 龍太(Ryota Komagata)、山根 智(Satoshi Yamane)
Kanazawa University

1 まえがき

本研究ではワイヤレスLANなどの確率リアルタイムシステムの仕様を確率時間オートマトン[7]として記述し, モデル検査[2]における重要な性質である, 安全性を検証する手法を提案する.

安全性とは, ``システムが動作中に危険な状態に到達しない"という性質であり, 危険な状態をターゲットとした到達可能性解析により検証可能である. この安全性検証では, システムの到達可能な全状態を網羅的に探索する必要がある.

確率時間オートマトンは時間の流れをモデル化する実数変数を持つため, その無限の状態空間を有限のリージョングラフ[7]で構築したとしても, 計算機のメモリに乗せることができないという問題が起こってしまい, その検証は困難である.

さらに確率時間オートマトンのゾーングラフ上の検証[7]では到達可能性解析の目的の状態に至る正確な確率を求めることができない. これは, 一般的な到達可能性問題は, ターゲットに至る一本のパスを発見することを目的としているが, 確率時間オートマトンにおける到達可能性問題では, ターゲットに至る複数のパスの合計確率として, 到達可能性問題が定義されることに依る.

これまでの確率時間オートマトンに対するモデル検査では, 先のゾーングラフによる検証[7]の他, 危険な状態から後方探索でグラフを構築する検証[9]が行われていた. それに比べ本研究では, 述語抽象化[11]の手法であるCEGAR(反例における抽象化と洗練の枠組み)[3]の手法を拡張した確率時間CEGARによって, 状態爆発の問題, 複数のパスの到達可能性の問題を同時に解決する. 確率時間CEGARの最大の利点は, 目的に応じて状態空間の抽象化を行うことができ, さらに抽象モデルにおいての検証によって得られた結果の正当性が証明されることにある.

CEGARのリアルタイムシステムへの適用として文献[12]が, 確率オートマトンへの適用として文献[6]が報告されているが, 確率時間オートマトンへの適用は本研究が初めてである. 本研究の新規性は以下である.

- 確率時間オートマトンの到達可能性問題への反例としてアドバサリと最小反例[4]の組を採用した
- 反例を複数のパスとすることで, 抽象化されている異なる時間経過量のパスを区別する必要があるが, 同時実行反例解析として判定し解決した
- 1.と2.を用いて確率時間オートマトンへのCEGARの適用を実現した

2 確率時間オートマトン

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について定義を行う. 確率到達可能性問題は, 確率時間オートマトンの意味に当たる時間確率システム上で定義される.

2.1 確率時間オートマトン

確率リアルタイムシステムのモデルとして確率時間オートマトン[7]を定義する.

定義 2.1 確率時間オートマトン 確率時間オートマトンGは以下となる組$(L, l_0, C, Inv, prob)$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 $prob \subseteq L \times Zones(C) \times \mathsf{Dist}(2^C \times L)$

確率時間オートマトンの例を図1に示す.

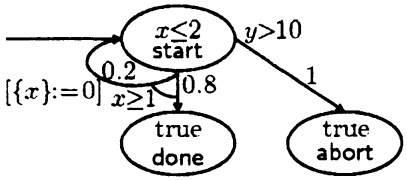

図 1:確率時間オートマトンG_1

確率時間オートマトンの状態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. 確率時間オートマトンの動作は, 初期ロケーションと全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. そこから状態間を時間経過か離散遷移を行うことによって変化する. 時間経過は同一ロケーション内で行い, 状態(l,ν)から$t\in\mathbb{R}^{>0}$時間経過する場合は, 確率1で状態$(l,\nu+t)$になる.

確率時間オートマトンGの意味を時間確率システム$\mathcal{M}$[7]とし, 抽象化における具体モデルとして利用する. 時間確率システムはマルコフ決定過程の形をとる.

定義 2.2 時間確率システム 確率時間オートマトンGの意味となる時間確率システム$\mathcal{M}_G$を以下のような組$(S,q_0,Steps)$とする.

- 状態集合 $S\subseteq L\times\mathcal{V}_C$
- 初期状態 $q_0=(l_0,\nu_0)$
- 状態遷移関係 $Steps\subseteq S\times \mathsf{Dist}(S)$

状態遷移関係$Steps$は時間経過と離散遷移からなり, 以下のように定める.

- 時間経過 $(l,\nu)\xrightarrow{\mu}(l',\nu')$

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

- 離散遷移 $(l,\nu)\xrightarrow{\mu}(l',\nu')$

$$\mu((l',\nu'))=\textstyle\sum_{X\subseteq C\&\nu'=\nu[X:=0]}\mu_G(X,l')$$

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

定義 2.3 時間確率システムのアドバサリ 時間確率システム$\mathcal{M}$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$last(\omega_{fin})$を遷移元とする確率分布を割り当てる関数である. また, 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をシンプルなアドバサリA_{simple}として,

$$\forall\omega_{fin},\omega'_{fin}.\ \left(last(\omega^{fin})=last(\omega'^{fin})\right)\rightarrow \left(A_{simple}(\omega^{fin})=A_{simple}(\omega'^{fin})\right)$$

となるアドバサリと定義して区別する.

シンプルなアドバサリA_{simple}で非決定を解決することで, 時間確率システム$\mathcal{M}$は離散時間マルコフ連鎖$\mathcal{MC}^A=(S,s_0,\mathrm{P}^A)$を同じ状態集合を用いて構築することができる. その遷移確率行列$\mathrm{P}^A=S\times S\rightarrow[0,1]$は以下となる.

$$\mathrm{P}^A(s,s')=\begin{cases}\mu(s') & \text{if } \exists\omega.last(\omega)=s\wedge\\ & A(\omega)=\mu\\ 0 & \text{otherwise}\end{cases}$$

ここで有限長のパスω_{fin}の確率$Prob^A_{fin}(\omega_{fin})$を以下のように定義する.

$$Prob^A_{fin}(\omega_{fin})\stackrel{\text{def}}{=}\begin{cases}1 & \text{if } |\omega|=0\\ \mathrm{P}^A(\omega_{fin}(0),\omega_{fin}(1))\cdots & \\ \mathrm{P}^A(\omega_{fin}(n-1),\omega_{fin}(n)) & \text{otherwise}\end{cases}$$

次に, 有限長のパスω_{fin}のアドバサリAに従うシリンダー集合を以下のように定義する.

$$C^A(\omega_{fin})\stackrel{\text{def}}{=}\{\omega\in Path^A_{ful}|\omega_{fin}\omega\}$$

そして, $\omega_{fin}\in Path_{fin}$におけるシリンダー集合$C^A(\omega_{fin})$を包含する$Path^A_{ful}$上の最小完全加法族を$\Sigma^A$とする. これにより$\Sigma^A$上の確率速度$Prob$は, 全ての$\omega_{fin}\in Path_{fin}$において$Prob(C^A(\omega_{fin}))=Prob^A_{fin}(\omega_{fin})$とすることで定義できる. このことから, アドバサリAにおいて, ある状態集合S_eに到達する到達確率は以下から求められる.

$$Prob^A(S_e)=Prob(\{\omega|\omega\in Path^A_{ful}\&\exists i\in\mathbb{N}.\omega(i)\in S_e\})$$

2.2 到達可能性問題

到達可能性問題はソフトウェアの検証におけるもっと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. よって, 本研究では到達可能性問題だけを検証対象とする.

定義 2.4 到達可能性問題と反例 確率時間オートマトンGによる到達可能性問題は, 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)である. また, Gの意味である$\mathcal{M}$における目的ロケーション$l \in L_e$をもつ状態の集合を$S_e \in S$とする. この答えが'yes'であるとは, 確率時間オートマトンGの意味である時間確率システム$\mathcal{M}$において, $\forall A, s \in S_e.Prob^A(s) \leq \lambda$の場合に限る. それ以外の場合は'no'である.

アドバサリAと, 目的状態S_eに到達する有限長のパスの有限集合$\Omega \subseteq Path^A_{fin}(S_e)$の組$(A, \Omega)$の中で, $\sum_{\omega \in \Omega} Prob^A_{fin}(\omega) > \lambda$となるものを反例と呼ぶ.

ここでの到達可能性問題は, 等号を含まない形に制限している. それにより, 文献[4]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事が示されている. それにより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
$\Omega \subseteq \{\omega | \omega \in Path^A_{fin} \wedge last(\omega) \in S_e\}$で
$\sum_{\omega \in \Omega} Prob^A_{fin}(\omega) > \lambda$となる反例が存在する.

3 述語抽象化

述語抽象化[11]は無限状態遷移系の有限の近似を計算し, 検証の時の状態爆発を抑制するために用いられる. 本手法では確率時間オートマトンの意味である時間確率システムの時間を述語を用いて抽象化し, 状態数を削減する. 本手法は時間オートマトンに適応されている文献[12]に従い, 確率に拡張して利用する.

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.

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

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$\psi\nu \in \{ture, false\}$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をみたす.

本研究ではロケーションごとに抽象化述語の集合を与える. ロケーションlにおける抽象化述語をψ^lで表し, ψ^lの集合を$\Psi^l = \{\psi^l_1, \cdots, \psi^l_n\}$とする. ただし, 述語trueはそれ以外の述語が入っていない述語集合Ψ^lに入っているものとする. Ψ^lにより, クロック評価νから長さnのビットベクトルb^lへのマッピングである抽象化関数α^lが決定される. ここで, すべてのロケーションにおける抽象化述語の集合を$\Psi = \{\Psi^{l_0} \cup \cdots \cup \Psi^{l_k}\}$とすると, Ψにより抽象化関数αが決定される. b^lのi番目の要素は, lにおいて, $\psi^l_i\nu$が真となるときに限り真となる. いま, lにおける長さnのビットベクトルの集合B^l_nの要素であり, B^l_nはドメイン$\{0, \cdots, n-1\}$と変域$\{0,1\}$を持つ関数であると仮定する. 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$\mathcal{B}$とする. αの逆像は具体化関数γであり, ビットベクトルの集合$\mathcal{B}$からビットベクトルのi番目の要素が真であるときはψ^l_iを満たし, 偽ならば満たさないすべてのクロック評価の集合へのマッピングである. 従って, 具体状態(l, ν)の集合は抽象化関数αにより抽象状態$\alpha((l,\nu))$に変換され, 抽象状態(l, b^l)はγにより具体状態の集合$\gamma((l, b^l))$にマッピングされる.

定義3.2 抽象化・具体化 述語の有限集合$\Psi^{all} = \{\Psi^{l_0} \cup \cdots \Psi^{l_k}\}$が与えられたとき, 抽象化関数$\alpha : L \times \mathcal{V}_C \to L \times \mathcal{B}$, 具体化関数$\gamma : L \times \mathcal{B} \to L \times \mathcal{V}_C$は以下のように定義される.

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

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

また, $b^l\Psi^l$を以下のように定義して用いる.

$$b^l\Psi^l = \zeta \text{ s.t. } \nu \triangleright \zeta \wedge \psi^l_i\nu \equiv b(i)$$

抽象化述語と抽象化・具体化関数を用いて, ある述語集合Ψにおけうる時間確率システムの抽象構造を構築する. 抽象構造はマルコフ決定過程の形をとる.

定義3.3 抽象構造の形成 確率時間オートマトンGから変換された時間確率システム$\mathcal{M}$における, 述語集合Ψによる抽象構造
$\mathcal{M}^\sharp_\Psi = (S^\sharp, s^\sharp_0, Steps^\sharp)$を以下のように構築する.

- $S^\sharp = L \times \mathcal{B}$
- $s_0^\sharp = \alpha(s_0)$
- $Steps^\sharp \subseteq S^\sharp \times \mathsf{Dist}(S^\sharp)$

$((l,b),\mu^\sharp) \in Steps^\sharp$が存在するとは，
$\exists(l,\nu) \in \gamma((l,b)).((l,\nu),\mu) \in Steps$が存在する場合に限り存在する．$\mu^\sharp$は
$\mu^\sharp((l,b)) = \sum_{(l,\nu)\in\gamma((l,b))} \mu((l,\nu))$で定義される確率分布である．

この抽象構造の形成では，時間確率システムに対してオーバー近似になる．オーバー近似であるとは具体構造である時間確率システムの遷移を抽象構造は全て持っていることを示す．その方法は文献[12]に従い，確率分布の導入により拡張している．

定義 3.4 オーバー近似 抽象構造$\mathcal{M}^\sharp$が時間確率システム$\mathcal{M}$のオーバー近似にあるとは，以下の性質を満たすことである．

全ての$(l,\nu),(l',\nu')$において，$((l,\nu),\mu) \in Steps \wedge \mu((l',\nu')) > 0$ならば，$(\alpha((l,\nu)),\mu^\sharp) \in Steps^\sharp \wedge \mu^\sharp(\alpha(l',\nu')) > \mu((l',\nu'))$である．

また，このオーバー近似の性質は，具体構造である時間確率システムで到達するパスは必ず抽象構造でも到達することを示している．このことは到達確率において，オーバー近似で構成された抽象構造の最大到達確率は，時間確率システムの最大到達確率以上になることを意味する．

次に，述語の数を有限に制限するために basis[12]を導入する．

定義 3.5 basis[12] 述語集合ψが basis であるとは，全てのクロック評価$\nu_1,\nu_2 \in \mathcal{V}_C$において，全ての述語$\psi \in \Psi \in \mathbf{\Psi}$で，$\psi\nu_1 = \psi\nu_2$となるものである．

以後，導入する述語を basis に含まれる述語に制限する．なお，basis である述語集合で抽象構造を構築した場合，その抽象構造はリージョングラフになる．

図 1 確率時間オートマトンG_1を抽象化した場合の抽象構造$\mathcal{M}^\sharp$を図 2 に示す．抽象化に伴う述語集合は，$\psi_0^{\mathrm{st}} = y \leq 8$, $\psi_1^{\mathrm{st}} = x - y < -8$である．

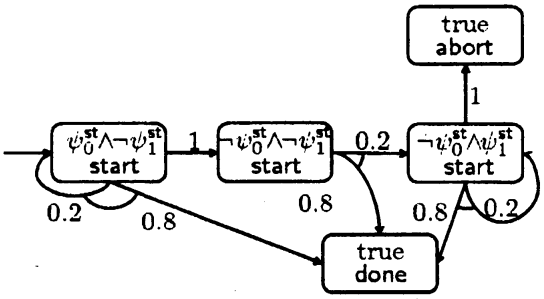

図 2:G_1の抽象構造$\mathcal{M}^\sharp$

抽象化を行うと元のシステムの性質を抽象化したシステムは持たない場合や，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，抽象構造と時間確率システムの対応関係を定義し，反例の候補を定義する．

定義 3.5 パスの対応関係 時間確率システム$\mathcal{M}$と述語集合$\mathbf{\Psi}$によって構成された抽象構造$\mathcal{M}^\sharp$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の全ての遷移で以下の 1.〜3.が導ける順序通り構成されたものを言う．

1. $(l,\nu) \xrightarrow{\mu} (l',\nu')$ならば，抽象構造の遷移は$\alpha((l,\nu)) \xrightarrow{\mu} \alpha((l',\nu'))$
2. $(l,\nu) \overset{\mu}{\nrightarrow} (l,\nu')$かつ $\alpha((l,\nu)) = \alpha((l,\nu'))$ならば，抽象構造の遷移は存在しない．
3. $3.(l,\nu) \overset{\mu}{\nrightarrow} (l,\nu')$かつ $\alpha((l,\nu)) \neq \alpha((l,\nu'))$ならば，抽象構造の遷移は$\alpha((l,\nu)) \overset{\mu}{\nrightarrow} \alpha((l,\nu'))$

これは時間確率システムにおける時間経過が抽象構造において一つの状態の中で行われる場合，遷移が現れない事を示している．オーバー近似による抽象化のため，抽象構造の全てのパスが対応する時間確率システムのパスを持たないが，時間確率システムの全てのパスは対応する抽象構造のパスを持つ．今後，時間確率システムのパスωに抽象構造のパス$\omega^\sharp$が対応している場合，状態の抽象化関数と同じ記号を用いて$\alpha_{\mathrm{Path}}(\omega) = \omega^\sharp$と表記する．

定義 3.6 アドバサリの対応関係 抽象構造のアドバサリ$A^\sharp$が時間確率システムのアドバサリAに対応しているとは，全ての確率分布$A^\sharp((l,b)) = \mu^\sharp$と，全ての遷移先$(l',b')$において，全ての具体化し

た状態$s \in \gamma((l,b))$, $s' \in \gamma((l',b'))$が$\mu^\sharp((l,',b')) > \mu(s')$である確率分布$A(s) = \mu$となる時に限る.

このアドバサリの対応関係は, 後の反例解析にて役に立つ.

今後, 時間確率システムのアドバサリAに抽象構造のアドバサリ$A^\sharp$が対応している場合も同様に$\alpha_{Adv}(A) = A^\sharp$と表記する. ただし, $\alpha(A)$が定義されない場合もある.

このパスとアドバサリの対応関係を利用して, 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$\mathcal{M}$の抽象構造$\mathcal{M}^\sharp$の反例の候補とは, $l \in L_e$をもつ抽象状態の集合を$S_e^\sharp \in S^\sharp$として, $\Sigma_{\omega \in \Omega^\sharp} Prob_{fin}^{A^\sharp}(\omega) > \lambda$となる, 抽象構造上のアドバサリ$A^\sharp$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,
$\alpha_{Adv}(A) = A^\sharp \wedge \forall \omega^\sharp \in \Omega^\sharp . \alpha_{Path}(\omega) = \omega^\sharp$
の場合に限る.

ここで反例と反例の候補に関する重要な2つの定理を挙げる.

定理 3.1. 時間確率システムで反例が存在するならば, 必ず抽象構造で反例の候補が存在する.
証明. 抽象構造は時間確率システムのオーバー近似であるから, 時間確率システムのパスをすべて抽象構造はもつことになるため,
$\max Prob^A(S_e) \leq \max Prob^{A^\sharp}(S_e^\sharp)$である.
マルコフ決定過程の到達確率が検証確率を上回るならば必ず反例が存在することを述べた定理2.1により, 検証確率を時間確率システムの最大確率が上回り反例が存在する場合, 抽象構造は時間確率システム以上の到達確率になり検証確率をより大きくなり反例の候補が存在する.

この定理は抽象構造で反例の候補が存在しなければ反例が存在しない事を示している. よって, 抽象構造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で'no'と言うことができる.

定理 3.2. 述語集合がbasisであるとき, 反例が存在する場合は反例の候補が存在し, 反例が存在しなければ反例の候補は存在しない.
証明. この定理は文献[12]で時間システムにおけるμ計算において証明されている. このことは述語集合がbasisであるとき, 確率時間システムで到達するパスに対応する抽象構造のパスは存在する事を示している. 反例のパス集合Ωの全てのパスが抽象構造でも存在する事になり, これで反例の候補が構成できる.

この定理は述語集合がbasisであれば, 確率到達可能性問題が解ける事を示している.

4 反例解析

本研究における反例解析は以下の手順で行われる.

1. 反例の候補の導出(Compute Candidate Counter-example):抽象構造に反例の候補が存在するか否かを求め, 適当な反例の候補を導出する
2. 反例解析(Counter-example Analysis):反例の候補から対応する反例が存在するかどうか求める
3. 精錬(Refinement):反例の候補に対応する反例が存在しなかった場合, その反例の候補を持たない抽象構造を構築できる新しい述語を導出する

本節では上記の3つの手順についてそれぞれアルゴリズムを提案する.

4.1 反例の候補の導出

まず、反例の候補を導出する手順は以下である。

1. 検証確率λより大きい到達確率となるアドバサリ$A^\sharp$を求める. このアドバサリが存在しなければ反例の候補は存在しない. これを満たすアドバサリとして最大到達確率となるアドバサリをを採用する.
2. $A^\sharp$上で合計到達確率が検証確率λより大きい有限数有限長のパス集合$\Omega^\sharp$を求める. これを満たすアドバサリとして最小反例[4]を採用する.

1.によってアドバサリ$A^\sharp$が求まれば, 定理2.1.より必ず2.においてパス集合$\Omega^\sharp$を求めることが可能であり, 反例の候補を導出できる. ここでアドバサリに最大確率のアドバサリを用いることで, 最大確率の

アドバサリが検証確率を満たさなければ全てのアドバサリで検証確率を満たさず，反例の候補が存在しないといえる．さらに，反例の候補が存在しないということは定理2.1.より，反例が存在しないといえ，確率到達可能性問題で'no'と答えることができる．

2.ではまず最大到達確率を求め，その最大到達確率になるアドバサリを求める．前述の通り抽象構造はマルコフ決定過程で定義されている．そこで，文献[1]によって報告されているマルコフ決定過程で最大到達確率を線形計画法によって求める方法を利用する．この方法では全ての状態の目的状態への到達確率を求めることになるため，この各状態の最大到達確率を利用することにより，各状態がどの確率分布を選択することによって最大到達確率になるのかわかる．各状態でその確率分布を返すアドバサリが最大到達確率となるアドバサリ$A^\sharp$になる．

2.では，求めたアドバサリ上で到達合計確率が検証確率λより大きい有限数有限長のパス集合$\Omega^\sharp$を求める．求めたアドバサリ$A^\sharp$を用いて，マルコフ決定過程である抽象構造は離散時間マルコフ連鎖に変換できる(2節時間確率システムと同様)．そこで，文献[4]によって離散時間マルコフ連鎖の反例として提案されている，最小反例(smallest ecounterexample)を利用する．

定義4.1 最小反例(smallest counterexample)[4]
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$\lambda$において，最小反例$\Omega_{smallest}$とは，合計到達確率が検証確率$\lambda$より大きいパスの集合の中で，最も要素数が少ないく，さらにその中で最も合計到達確率が最も大きいものを言う．

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つの理由により検証を効率化させることができる．

- 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
- 合計到達確率が大きくすることで，単体で到達確率の大きいパスを優先して解析する

以上の抽象構造上で最大到達確率となるアドバサリ$A^\sharp$と，このアドバサリ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．

4.2 反例解析

前節で求めた抽象構造の反例の候補に対し，時間確率システムに対応する反例が存在するか否かを求める．本手法では反例解析では反例の候補のうち，パスの集合のみを用いる．反例解析は以下の2つの手順とに分かれる．

1. パス反例解析:反例の候補のパス集合$\Omega^\sharp$の一つ一つのパス$\omega^\sharp$に対し，定義3.6を満たす対応する時間確率システムのパスωが存在するかどうかを確かめ，反例となる時間確率システムのパス集合Ωを得る
2. 同時実行反例解析:パス反例解析で求めた時間確率システムのパス集合Ωが，一つの時間確率システムのアドバサリAによって得られるかどうかを求める

この各手順で求められたパス集合とアドバサリを反例とし，反例が求まれば確率到達可能性問題に対して，'yes'と示すことができる．この手順を用いるため，反例解析では反例の候補のパス集合のみを用いることになり，求められる反例は定義されている反例の候補に対応した反例だけではなくなる．

4.2.1 パス反例解析

反例の候補である抽象構造のパス$\omega^\sharp$に対応する確率時間システムのパス集合$\{\omega|\alpha(\omega) = \omega^\sharp\}$を求めるために，$\omega^\sharp$のそれぞれの状態を具体化し，初期状態から到達可能かつ目的状態に到達可能なものに絞る．本研究では，パスの最後から最弱前条件[5]を満たす状態を求め，その後前方から最弱後条件を満たす状態を再び求めその積をとる．つまり，最弱前条件によって目的状態に到達可能な最大の状態集合を求め，最弱後条件によって初期状態から到達可能な最大の状態集合を求め，その積をとることで初期状態から目的状態に到達する最大の状態集合を求める．

最弱前条件を求めるには，抽象構造のパスの最後の状態を具体化関数によって時間確率システムの状態の集合に戻し，さらに不変条件を満たすもののみに絞る．そしてそこから一つ前の状態でも具体化関数で時間確率システムの状態に戻し，今度は不変条件だけではなく，次の遷移の状態集合に到達可能なものだけを残す．これをパスの最初の状態まで繰り返すことにより，初期状態を含んでいるか評価することで時間確率システム上の対応す

るパスが存在するか否かを求める. また最弱後条件を求めるには, 初期状態から後方へ同様に行う.

このパスの各抽象状態に対応する時間確率システムの状態の集合を, ゾーンを用いて領域として考える. そして集める状態のゾーンは, 定義 3.5 の 2. の条件を考慮し, 一つの状態につき 2 つにする. 一つは到達条件として 2.の遷移を行う前のゾーン ζ^{get}であり, もう一つは出発条件として 2.の遷移を行った後のゾーンζ^{go}とする. これを後方に目的状態から初期状態まで求め, その後前方に初期状態から目的状態へ連言を取り求める. 後方から行い最初の状態の到達条件が初期条件を含んでいれば, 到達可能と判断する. 後方から最弱前条件で到達可能であれば, 当然ながら前方からの最弱後条件で目的状態に到達可能である.

本検証法は, 一つでも時間確率システムで対応するパスがない抽象構造のパスがあれば, 即座に反例の候補を偽反例と判断し, 精錬に移る.

4.2.3 同時実行反例解析

同時実行反例解析ではパス反例解析で求めた$\mathbf{\Omega}$の要素Ωの一つでもある時間確率システムのアドバサリAで構成できるかどうかを確かめる. 構成できれば, 反例(A, Ω)を示すことができる.

これらのパス集合の集合$\mathbf{\Omega}$は反例の候補のアドバサリ$A^\sharp$下の$\omega^\sharp$から生成されているため, 同一の状態から離散遷移を行う場合同一の離散遷移の確率分布を用いる. よって, 遷移が離散遷移の部分に関してはアドバサリは問題にならない. しかし, 遷移が時間経過の部分に関しては, 定義 3.5 の 2. によって, 異なる時間経過量を示すアドバサリが必要になる場合がある. 同時実行反例解析では時間経過について異なるアドバサリが必要でないかのみ判断すれば良い.

定義 2.3 よりアドバサリは, パスを入力することで次の遷移の確率分布を返すものであった. このことより, ある 2 つのパスを構成するために同一のアドバサリで構成するためには, 初期状態から同一の確率分布を選びながら異なる確率で遷移する状態までの動作が等しい必要がある. よって, 同時実行反例解析では任意の 2 つのパスが初期状態からたどり初めて異なる動作を示す状態の出発条件のみを比べる. もしこの状態の出発条件で共通のクロック評価があれば, その遷移でそのクロック評価になる時間経過量を示すアドバサリが存在すると言える.

このことは, そのクロック評価から初期状態へ最弱前条件となる出発条件, 到達条件を求め, そのゾーン内のクロック評価となる時間経過量の動作をするアドバサリが共通するアドバサリであると言える.

4.3 精錬

反例解析によって, 導出した反例の候補$(A^\sharp, \Omega^\sharp)$から反例の組$(A, \Omega)$を示せなかった場合, 同じ反例の候補を導出しないために新しい述語を追加し, 抽象構造の精錬を行う.

述語には, 2 つのゾーンを分割可能な述語を選ぶ. 抽象述語は定義の上で, 空間$\mathbb{R}^C$の凸部分集合であるゾーンの辺の形をとる. よって, いかなる 2 つゾーンζ_1, ζ_2も有限の述語集合Ψで

$\forall b.b\Psi \wedge \zeta_1 \neq b\Psi \wedge \zeta_2$のように分割することができる. ここで$\Psi$の要素数は最小にすることが状態数削減には望ましいが, 効率的なアルゴリズムは知られていない.

パス反例解析では、反例の候補として導出したパス集合の一つのパス$\omega^\sharp \in \Omega^\sharp$に対し, 時間確率システムの対応するパス$\omega$が存在する事を調べていた. 偽反例と判断されるのは, あるパスにおいて$\zeta_0^{get} \wedge \zeta_0 = \mathsf{false}$となる場合であるので、この 2 つを分割すればよい.

同時実行反例解析では, 時間確率システムの単一のアドバサリで, 反例の候補のパス集合$\Omega^\sharp$の全てのパスが対応する時間確率システムのパスを構成可能かどうかを調べていた. 偽反例と判断されるのは, あるロケーションlにおいてあるζ_1^{go}とζ_2^{go}が$\zeta_1^{go} \neq \zeta_2^{go}$にもかかわらず同一の抽象状態で構成されている時であった. よって, ζ_1^{go}とζ_2^{go}で分割すればよい.

5 確率時間 CEGAR の提案

述語抽象化, 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)[3]の枠組みである. ここまでに説明した抽象化と反例解析を CEGAR に適応した確率時間 CEGAR の検証の流れを図 3 に従い説明する.

1. Abstraction: 初期の述語集合から抽象構造$\mathcal{M}_\Psi^\sharp$を計算する. 初期の述語集合Ψは, $\forall l \in L.\Psi^l = \{\mathrm{true}\}$である.
2. Compute Candidate Counter-example : $\mathcal{M}_\Psi^\sharp$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, 'no'を出力し検証を終了する.

3. Counter-example Analysis: 反例の候補$(A^\sharp, \Omega^\sharp)$に対応する反例$(A, \Omega)$が存在するかどうかを求める. 存在すれば'yes'を出力し, 検証を終える.
4. Refinement: 反例解析の結果から新しい述語を導出し, 新しい述語集合Ψ'を得る.
5. Abstraction: 述語が追加された述語集合Ψ'から抽象構造$\mathcal{M}^\sharp_{\Psi'}$を計算する.
6. 2.に戻る.

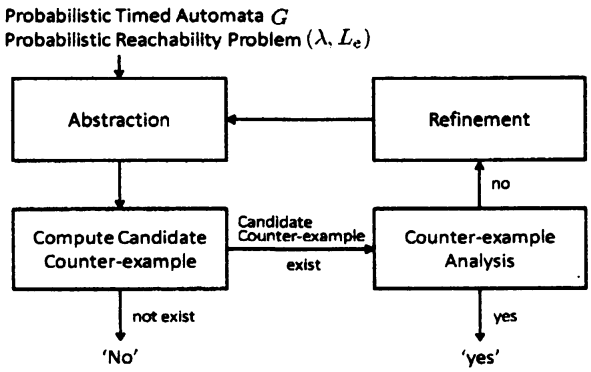

図 3 確率時間 CEGAR による検証

これらのループを繰り返していくことにより, 最終的にシステムが``Safety"か``Danger"であるかを判定する.

次に確率時間 CEGAR における検証の正当性と停止性についての定理を与える.

定理 5.1 検証の正当性 この検証は必ず時間確率システムの解と一致する解を得ることができる.

証明. 抽象構造はいかなる述語集合であっても, 定理 3.1 より健全性を保つ. CEGAR サイクルの繰り返しによる述語の追加によって述語集合がいつかは basis になることによって, 定理 3.2 より検証は必ず解を得る.

証明 5.2 検証の停止性 この検証は必ず有限時間内に終了する.

証明. 文献[12]により不変条件, 遷移条件で現れる最大整数が存在するとき, basis は有限の述語で構成される. そのため, 必ず新しい述語が導出される時間確率 CEGAR のサイクルは有限回数となり, この検証は有限時間で終了すると言える.

6 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, CEGAR を導入することにより, 効率的な検証手法を確立した. 本論文の主な貢献は以下の 2 点である.

- 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, そしてその偽反例による抽象構造の精錬手法を実現した.
- 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, 検証対象に応じた効率的な状態空間の構築を可能にした.

参考文献

[1] Bianco, A. and de Alfaro, L.: Model checking of probabilistic and nondeterministic systems, LNCS 1026, 1995, pp. 499–513.
[2] Clarke, E. M., Grumberg, O., and Peled, D.: Model Checking, MIT, (2000).
[3] Clarke, E. M., Grumberg, O., Jha, S., Lu, Y., and Veith, H.: Counterexample-Guided Abstraction Refinement, LNCS 1855, 2000, pp. 154–169.
[4] Han, T. and Katoen, J. P.: Counterexamples in probabilistic model checking, LNCS 4424, 2007, pp. 72–86.
[5] Henzinger, T. A., Nicollin, X., Sifakis, J., and Yovine, S.: Symbolic Model Checking for Real-Time Systems, Information and Computation, Vol. 111(1994), pp. 394–406.
[6] Hermanns, H., Wachter, B., and Zhang, L.: Probabilistic CEGAR, LNCS 5123, 2008, pp. 162–175.
[7] Kwiatkowska, M., Norman, G., Segala, R., and Sproston, J.: Automatic Verification of Real-Time Systems with Discrete Probability Distributions, LNCS 1601, (1999), pp. 75–95.
[8] Kwiatkowska, M., Norman, G., and Sproston, J.: Symbolic Computation of Maximal Probabilistic Reachability, LNCS 2154, (2001), pp. 169–183.
[9] Kwiatkowska, M. Z., Norman, G., and Sproston, J.: Symbolic Model Checking of Probabilistic Timed Automata Using Backwards Reachability, Technical Report CSR-00-01, 2000.
[10] Puterman, M.: Markov Decision Processes: Discrete Stochastic Dynamic Programming, Wiley- Interscience, 1994.
[11] S. Graf and H. Saidi: Construction of Abstract State Graphs with PVS, LNCS 1254, 1997, pp. 72–83.
[12] Sorea, M., Oller, M. O. M., Oller, M. O. M., Rue, H., and Rue, H.: Predicate abstraction for dense real-time systems, Electronic Notes in Theoretical Computer Science 65(6), 2001, pp. 2002.