情報処理学会論文誌 Vol.53 No.12 2671–2681 (Dec. 2012)

確率線形ハイブリッドオートマトンの到達可能性検証

畠中 克也[1] 山根 智[1,a)]

受付日 2012年3月1日, 採録日 2012年9月10日

概要：リアルタイムシステムに対するモデル検査の需要は増加しており，時間オートマトンで記述したモデルを検証可能な UPPAAL に代表される検証器の開発がさかんに行われている．本研究では，コスト付き確率時間オートマトンのモデル検査の procedure を拡張し，確率線形ハイブリッドオートマトンに対する検証を可能にする．さらに，モデルに対する検証を自動化するため，拡張した procedure を計算機上に実装する．ケーススタディとして，無線センサネットワークをとりあげ，確率線形ハイブリッドオートマトンによるモデル化と，その検証例を示す．

キーワード：確率線形ハイブリッドオートマトン，モデル検査，到達可能性解析，Fourier-Motzkin 消去法，無線センサネットワーク

Reachability Verification of Probabilistic Linear Hybrid Automata

Katuya Hatanaka[1] Satoshi Yamane[1,a)]

Received: March 1, 2012, Accepted: September 10, 2012

Abstract: Model checking is a formal method exhaustively verifying whether behaviors of a system satisfy specific characteristics. It can be applied to specification, testing or debugging stages. Priced probabilistic timed automata (PPTAs) can be used to model real-time systems with probability and cost features. In this paper, We define probabilistic linear hybrid automata (PLHAs), which are superclass of PPTAs. PLHA has real-valued variables proportional to time and discrete probabilistic distributions. Furthermore, we extend an procedure for cost-bounded probabilistic reachability problem in PPTAs. The procedure performs operations on convex polyhedra which presents symbolic states in PLHAs. As a case study, the paper presents simplified model of wireless sensor networks by use of parallel composition of PLHAs. PPTA can't handle this model because the model has multiple costs. Our verification program enables automatic verification for the such model.

Keywords: probabilistic linear hybrid automaton, model checking, reachability analysis, Fourier-Motzkin elimination, wireless sensor network

1. はじめに

モデル検査とは，システムの振舞いを網羅的に探索し，安全性や活性を満たすかどうかを検証することであり，システムの信頼性保証に貢献できることが期待できる．特に，リアルタイムシステムに対するモデル検査の需要は増加しており，時間オートマトン [2] で記述したモデルを検証可能な UPPAAL [21] に代表される検証器の開発がさかんに行われている．

要求されるシステムの複雑化にともない，仕様記述言語も拡張が続けられてきた．通信プロトコルなどで現れる乱数など，確率的動作を持つシステムをモデル化するために確率時間オートマトン [6] が提案された．Mutsuda らの論文 [11] では，確率線形ハイブリッドオートマトンにおける確率到達可能性問題の検証手法が提案されているが，これは，ターゲットへある確率以上で到達する 1 本のパスが存在するかを検証する手法である．確率到達可能性問題に正しく答えるには，確率分布による分岐の結果生じる複数のパスの合計確率を求める必要がある．Berendsen らの論文 [3] では確率線形ハイブリッドオートマトンのサブクラ

[1] 金沢大学大学院自然科学研究科電子情報科学専攻
Division of Electrical and Computer Engineering, Graduate School of Natural Science and Technology, Kanazawa University, Kanazawa, Ishikawa 920–1192, Japan
[a)] syamane@is.t.kanazawa-u.ac.jp

スであるコスト付き確率時間オートマトンに対して，前述の問題を解決する procedure が示されているが，コストを 1 つしか扱えないため，無線センサネットワークなどの重要な事例の仕様記述や検証が十分にはできない．なお，ハイブリッドオートマトンのモデル検査のアルゴリズムは停止性が保証されていないので，Alur ら [9] に従って，モデル検査の procedure と呼ぶ．

本研究では，コスト付き確率時間オートマトンのモデル検査の procedure [3] を拡張し，確率線形ハイブリッドオートマトンに対する検証を可能にする procedure を開発して，拡張した procedure を計算機上に実装する．ケーススタディとして，無線センサネットワークをとりあげ，確率線形ハイブリッドオートマトンによるモデル化と，その検証例を示して，提案手法の有効性を実証する．

Berendsen らの論文 [3] で定義されているコスト境界付き確率到達可能性問題との今回対象としている問題の差異は以下である．

(1) Berendsen らのコスト付き確率時間オートマトン [3] は各ロケーションに 1 つのみのコスト変数が記述可能であり，一方，本論文の確率線形ハイブリッドオートマトンは複数のコスト変数が記述可能である．なお，コスト変数とは時間微分係数がクロック変数と異なる変数である．

(2) Berendsen らのコスト境界付き確率到達可能性問題 [3] はコスト付き最大確率到達性の検証である．一方，本論文の事後条件付き確率到達可能性問題もコスト付きの最大確率到達性の検証である．両者の違いは与えられたコスト条件以下であるか，コスト条件と等しいかの違いだけであり，本質的な差はない．

本論文の構成は，以下のとおりである．2 章では，本文で用いる記号の説明や，前準備として必要な定義を述べる．3 章では，システム記述言語である確率線形ハイブリッドオートマトンを定義する．4 章では，本研究がターゲットとしている検証問題である確率到達可能性問題を定義し，問題を解くための具体的な procedure を説明する．5 章では，計算機上に実装した検証器を用いて，確率線形ハイブリッドオートマトンで記述したモデルを検証した結果を示す．最後に，6 章でまとめとする．

2. 準備

本章では，確率線形ハイブリッドオートマトンの定義にあたって必要な定義を述べる．なお，本論文では以下の表記を使用する．

- $[\![\cdot]\!]$ は意味を表す．
- 集合 $\mathcal{A}$，$\mathcal{B}$ に対して，$\mathcal{A}$ から $\mathcal{B}$ への写像全体の集合を $\mathcal{B}^{\mathcal{A}} = \{f \mid f : \mathcal{A} \to \mathcal{B}\}$ と書く．

2.1 離散確率分布

有限集合 Q に対して，関数 $\mu : Q \to [0,1]$ を考える．$\sum_{q \in Q} \mu(q) = 1$ を満たすとき，この関数 μ を離散確率分布という．非可算集合 Q_∞ に対して，$\mathsf{Dist}(Q_\infty)$ を Q_∞ の有限部分集合上の離散確率分布の集合とする．$\sum_{q \in Q} \mu(q) \leq 1$ を満たす場合は，離散部分確率分布といい，離散部分確率分布の集合を $\mathsf{SubDist}(Q_\infty)$ と表す．

離散確率分布 $\{q \mapsto 1\}$ を点分布という．点分布は，ある 1 つの要素 q に対して確率 1 を与え，その他の要素に対する確率はすべて 0 であるような離散確率分布である．離散確率分布 $\mu \in \mathsf{Dist}(Q')$ に対して，サポート集合を $\mathsf{support}(\mu) = \{q \in Q' | \mu(q) > 0\}$ と定義する．

2.2 変数

本論文で用いる変数とは，非負実数をとる変数であり，すべての変数は時間に比例して増加する．特に，傾きが 1 である変数をクロックと呼び，変数の増加量はクロックの増加量の定数倍となる．変数に実数値を割り当てる関数 $v : X \to \mathbf{R}_{\geq 0}$ を評価関数と呼ぶ．ここで，X は変数の集合であり，X に対する評価関数全体の集合を $\mathbf{R}^X_{\geq 0}$ と書く．すべての変数にゼロを割り当てる評価関数を $\mathbf{0}$ と書く．後述するが，これはシステムの初期状態を表すのに用いられる．評価関数 $v \in \mathbf{R}^X_{\geq 0}$，非負実数 d，および変数に傾きを与える関数 $\gamma : X \to \mathbf{Q}$ に対して，$v + \gamma d$ は $x \mapsto v(x) + \gamma(x)d$ を表す．これは，d 単位時間の経過を表す．ただし，$\mathbf{Q}$ は有理数である．

2.3 ガード条件

変数集合 $X = \{x_0, x_1, \cdots, x_{n-1}\}$ に対する一次式 E を以下の構文で定義する．

$$E ::= a_0 x_0 + a_1 x_1 + \cdots + a_{n-1} x_{n-1} + a_n$$

ここで，$a_i \in \mathbf{Q}$ である．

X におけるすべての一次式の集合を $\mathsf{Linear}(X)$ と表す．

ガード条件 G とは，変数の制約条件を記述する式であり，一次不等式の連言で表される．形式的には，以下の構文で定義される．

$$G ::= E \leq 0 | G \wedge G | true$$

ただし，$E \in \mathsf{Linear}(X)$ である．X におけるすべてのガード条件の集合を $\mathsf{Guard}(X)$ と表す．

ガード条件の意味は凸多面体である．1 つの不等式からなるガード条件 $g = e \leq 0$ の意味は，

$$[\![g]\!] = \{v \in \mathbf{R}^X_{\geq 0} \mid a_0 v(x_0) + \ldots + a_{n-1} v(x_{n-1}) + a_n \leq 0\}$$

であり，2 つ以上の不等式の連言からなるガード条件の意味は，$[\![g_1 \wedge g_2]\!] = [\![g_1]\!] \cap [\![g_2]\!]$ である．ただし，$[\![true]\!] = \mathbf{R}^X_{\geq 0}$

とする．ある評価関数 v がガード条件 g を満たすとは，$v \in [\![g]\!]$ を満たすことと等価である．

以下に，ガード条件の記述例をいくつか示す．

(1) $x \geq 0$ は $((-1)x + 0y + 0z \leq 0)$ となる．

(2) $y \leq 2x + z$ は $((-2)x + 1y + (-1)z \leq 0)$ となる．

(3) $1 \leq x \leq 2$ は $((-1)x+0y+0z-1 \leq 0 \wedge 1x+0y+0z-2 \leq 0)$ となる．

ここでは，$\mathsf{Guard}(\{x, y, z\})$ の要素の例を示した．厳密に定義に従って記述するならば，右側の括弧内に示すように，係数が 0 や 1 の場合も明記しなければならないが，表記が煩雑になるため，本論文の以降の記述では，左側の直感的な記法を使用する．

変数が非負実数をとることを考えると，$\mathsf{Guard}(X)$ に属する任意のガード条件には，$x \geq 0$（$x \in X$）という条件が暗黙的に含まれている．

2.4 更新関数

変数の更新関数 u は変数から一次式への写像として定義される．すなわち，$u : X \to \mathsf{Linear}(X)$ である．X における更新関数全体の集合を $\mathsf{Update}(X)$ と表す．更新関数を用いることで，時間オートマトンにおけるクロックのリセット [2] や，コスト付き確率時間オートマトンにおけるコストの瞬間的な増加 [3] に相当する記述を統一的に扱うことができる．

ある変数 x の評価関数 v に更新関数 u を適用することを $v[u](x)$ と表記して，$v[u](x) = [\![u(x)]\!]$ となる評価関数を表す．例として，評価関数 $v(x) = 1$，$v(y) = 2$ に対して，更新関数 $u(x) = 0$，$u(y) = y + 1$ を適用した場合を考える．更新後の各変数の値を計算してみると，$v[u](x) = 0$，$v[u](y) = 3$ となる．

3. 確率線形ハイブリッドオートマトン

この章では，前章で述べた諸定義を用いて確率線形ハイブリッドオートマトン（Probabilistic Linear Hybrid Automaton, PLHA）の形式的定義を与える．同期アクション（イベント）による並列システムの協調動作を記述するために，PLHA の並列合成も定義する．

3.1 構文

定義 1 確率線形ハイブリッドオートマトンは，

$$M = \langle L, l, X, I, \Gamma, \Sigma, T \rangle$$

という組で定義される．各要素の詳細は以下のとおりである．

- L — ロケーションの有限集合
- $l \in L$ — 初期ロケーション
- X — 変数の有限集合
- $I : L \to \mathsf{Guard}(X)$ — ロケーションに不変条件を割り当てる関数
- $\Gamma : L \to X \to \mathbf{Q}$ — ロケーションごとに，変数に傾きを割り当てる関数
- Σ — アクションの有限集合
- $T \subseteq L \times \mathsf{Guard}(X) \times \Sigma \times \mathsf{Dist}(\mathsf{Update}(X) \times L)$ — 確率的遷移関係の有限集合

確率的遷移関係 $(l, g, a, p) \in T$ において，l は遷移元のロケーション，g は遷移するためのガード条件，a は遷移時のアクションを表す．アクションは，並列合成のために必要な構文であり，動作的な意味はない．並列合成については後述する．p は，遷移時における変数値の更新式と遷移先ロケーションを決定するための確率分布である．

PLHA M における遷移関係（すなわち，遷移先と遷移元が 1 対 1 に対応している関係）の集合を $\mathsf{edges}(M)$ とすると，$(l, g, a, p) \in T$ かつ $p(u, l') > 0$ のとき，$(l, g, a, p, u, l') \in \mathsf{edges}(M)$ である．

ここで，PLHA の記述例を示す．Kwiatkowska らの論文 [8] の Fig. 2 で示されている確率時間オートマトンのモデルを PLHA で記述して，**図 1** に示す．ただし，自明な箇所は読みやすさのために適宜省略している．このモデルでは，クロック変数に加えて，整数変数が用いられているが，PLHA では，変数に割り当てる傾きを 0 とすることで，そのような変数を扱うことができる．

- $L = \{s_0, s_1, s_2, s_3\}$
- $l = s_0$
- $X = \{x, e, try\}$
- $I = \{s_0 \mapsto x \leq 2, s_1 \mapsto x \leq 5, s_2 \mapsto true, s_3 \mapsto true\}$
- $\Gamma = \{$
  $s_0 \mapsto \{x \mapsto 1, e \mapsto 2.5, try \mapsto 0\}, s_1 \mapsto \{x \mapsto 1, e \mapsto 0, try \mapsto 0\},$
  $s_2 \mapsto \{x \mapsto 1, e \mapsto 0, try \mapsto 0\}, s_3 \mapsto \{x \mapsto 1, e \mapsto 0, try \mapsto 0\}\}$
- $\Sigma = \{\mathsf{send}, \mathsf{retry}, \mathsf{quit}\}$
- $T = \{(s_0, x \geq 1 \wedge try \leq N, \mathsf{send}, p_1), (s_1, x \geq$

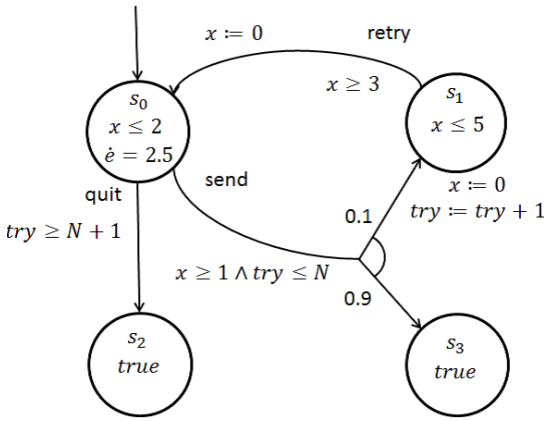

図 1 PLHA の記述例
Fig. 1 Example of PLHA.

$3, \mathsf{retry}, p_2), (s_0, try \geq N+1, \mathsf{quit}, p_3)\}$

3.2 並列合成

2 つの PLHA の並列合成規則を定義する．両方のオートマトンに含まれているアクションを同期アクションといい，同期アクションを持つ遷移どうしは，同時に起きることを意味する．これにより，並列に動作するシステムの協調動作を表現できる．

定義 2 $M_1 = \langle L_1, l_{init1}, X_1, I_1, \Gamma_1, \Sigma_1, T_1 \rangle$ と $M_2 = \langle L_2, l_{init2}, X_2, I_2, \Gamma_2, \Sigma_2, T_2 \rangle$ の並列合成 $M_1 \parallel M_2 = \langle L, l, X, I, \Gamma, \Sigma, T \rangle$ の定義は以下のとおりである．ただし，並列合成は，$X_1 \cap X_2 = \emptyset$ のときに限り定義される．

- $L = L_1 \times L_2$
- $l = (l_{init1}, l_{init2})$
- $X = X_1 \cup X_2$
- I は，(l_1, l_2) に $I_1(l_1) \wedge I_2(l_2)$ を割り付ける関数である．
- Γ は以下を満たす関数である．

$$\Gamma((l_1, l_2))(x) = \begin{cases} \Gamma_1(l_1)(x) & x \in X_1 \\ \Gamma_2(l_2)(x) & x \in X_2 \end{cases}$$

- $\Sigma = \Sigma_1 \cup \Sigma_2$
- $T \subseteq L \times \mathsf{Guard}(X) \times \Sigma \times \mathsf{Dist}(\mathsf{Update}(X) \times L)$
  以下に，確率的遷移関係の合成規則を示す．
  - 非同期アクションの合成
    $(l_1, g_1, a_1, p_1) \in T_1 \wedge a_1 \notin \Sigma_1 \cap \Sigma_2$ ならば，すべての $l_2 \in L_2$ に対して，$((l_1, l_2), g_1, a_1, p_1') \in T$ である．ただし，p_1' は，すべての $(u_1, l_1') \in \mathsf{support}(p_1)$ に対して，$p_1'(u_1, (l_1', l_2)) = p_1(u_1, l_1')$ となる確率分布である．同様に，$(l_2, g_2, a_2, p_2) \in T_2 \wedge a_2 \notin \Sigma_1 \cap \Sigma_2$ ならば，すべての $l_1 \in L_1$ に対して，$((l_1, l_2), g_2, a_2, p_2') \in T$ である．ただし，p_2' は，すべての $(u_2, l_2') \in \mathsf{support}(p_2)$ に対して，$p_2'(u_2, (l_1, l_2')) = p_2(u_2, l_2')$ となる確率分布である．
  - 同期アクションの合成
    2 つの遷移 $(l_1, g_1, a_1, p_1) \in T_1$ および $(l_2, g_2, a_2, p_2) \in T_2$ において，$a_1 = a_2 = a \in \Sigma_1 \cap \Sigma_2$ ならば，$((l_1, l_2), g_1 \wedge g_2, a, p') \in T$ である．ただし，p' は，すべての $(u_1, l_1') \in \mathsf{support}(p_1)$ および $(u_2, l_2') \in \mathsf{support}(p_2)$ に対して，$p'(u, (l_1', l_2')) = p_1(u_1, l_1') p_2(u_2, l_2')$ となる確率分布である．ただし，$u : X \to \mathsf{Linear}(X)$ は以下のような関数である．

$$u(x) = \begin{cases} u_1(x) & x \in X_1 \\ u_2(x) & x \in X_2 \end{cases}$$

3.3 意味論

3.3.1 確率システム

定義 3 [3], [6] 確率システム（Probabilistic System, PS）は，組 $\langle S, Steps \rangle$ で定義される．

- S — 状態集合
- $Steps \subseteq S \times \mathsf{Dist}(S)$ — 確率的遷移関係
  すべての $s \in S$ に対して，$(s, \mu) \in Steps$ となる μ が存在する．

確率システムの定義において，$Steps \subseteq S \times \mathsf{SubDist}(S)$ となるものは，部分確率システム（Sub-Probabilistic System, SPS）と呼ばれる．SPS では，ある確率でどこにも遷移しないという遷移関係が存在しうる．しかし，トラップ状態を 1 つ追加することで，部分確率システムは容易に確率システムに変換可能であることが知られている [3].

確率システムの動作

確率システム $< S, Steps >$ において，現在の状態が $s \in S$ であるとき，$(s, \mu) \in Steps$ であるような確率分布 μ が決定される．このような μ は 1 つであるとは限らず，確率分布の選択は非決定的に行われる．遷移時に，どの確率分布を選択するかを決定する関数をアドバサリという．遷移先の状態 s' は，選択した確率分布によって決定される．つまり，$\mu(s')$ の確率で，s' に遷移する．確率システムの動作は，次のような無限パス ω で表現できる．

$$\omega = s_0 \xrightarrow{\mu_0} s_1 \xrightarrow{\mu_1} s_1 \xrightarrow{\mu_2} \cdots$$

ここで，$(s_i, \mu_i) \in Steps$ であり，すべての $i \in \mathbf{N}$ に対して，$\mu_i(s_{i+1}) > 0$ である．無限パスのプレフィックスを有限パスいい，無限パスと有限パスをあわせて単にパスという．パス ω の i 番目の状態 s_i を $\omega(i)$ と表し，有限パス ω_{fin} の最後の状態を $\mathsf{last}(\omega_{fin})$ と表す．

定義 4 確率システム $\langle S, Steps \rangle$ のアドバサリ A は，確率システム上の有限パス ω_{fin} を確率分布 μ に写像する．すなわち，

$$A(\omega_{fin}) = \mu$$

ただし，$(last(\omega_{fin}), \mu) \in Steps$ である．

到達確率

状態 s から開始し，アドバサリ A によって生じる無限パス集合を $\mathsf{Path}^A_{\mathsf{full}}(s)$ と表す．Prob^A_s を $\mathsf{Path}^A_{\mathsf{full}}(s)$ 上の確率測度とすると，確率システム $\langle S, Steps \rangle$ とアドバサリ A に対して，状態 $s \in S$ から目的状態集合 $S^T \subseteq S$ への到達確率は以下の式で定義される [7].

$$\mathsf{ProbReach}^A(s, S^T) \stackrel{\mathsf{def}}{=} \mathsf{Prob}^A_s\{\omega \in \mathsf{Path}_{\mathsf{full}}{}^A(s) \mid \exists i \in \mathbf{N}.\omega(i) \in S^T\}$$

また，確率システム $Q = \langle S, Steps \rangle$ におけるすべてのアドバサリの集合を $\mathsf{Adv}(Q)$ とすると，確率システム Q において，状態 $s \in S$ から目的状態集合 $S^T \subseteq S$ への最大到達確率は，以下の式で定義される [3].

$$\mathsf{MaxProbReach}_Q(s, S^T) \stackrel{\mathsf{def}}{=} \sup_{A \in \mathsf{Adv}(Q)} \mathsf{ProbReach}^A(s, S^T)$$

有限の状態集合を持つ確率システムにおいて，最大到達確率を求める問題は，線形計画問題となることが知られている [1].

定義 5 確率システム $Q = \langle S, Steps \rangle$ と目的状態集合 $S^T \subseteq S$ とする．アドバサリ $A \in \mathsf{Adv}(Q)$ と $s \in S$ から開始される有限パス $\omega_{fin} \in \mathsf{Path}^A_{\mathsf{fin}}(s)$ に対して，

$$P_0{}^A(\omega_{fin} \Rightarrow S^T) = \begin{cases} 1 & last(\omega_{fin}) \in S^T \\ 0 & \text{otherwise} \end{cases}$$

ただし，$P_0{}^A(\omega_{fin} \Rightarrow S^T)$ は 0 回の遷移で，パス ω_{fin} が目的状態集合 $S^T \subseteq S$ に到達する確率を表す．つまり，パスの最後の状態が目的状態集合 $S^T \subseteq S$ の要素である確率を表す．任意の $n \in N$ と $\nu = A(\omega_{fin})$ に対して以下である：

$$P_{n+1}{}^A(\omega_{fin} \Rightarrow S^T) = \begin{cases} 1 & last(\omega_{fin}) \in S^T \\ \Sigma_{s \in S} \nu(s) \cdot P^A_n(\omega_{fin} \to s \Rightarrow S^T) & \text{otherwise} \end{cases}$$

任意の s に対して，以下が定義される．

$$P_n{}^{max}(s \Rightarrow S^T) \stackrel{\mathsf{def}}{=} \sup_{A \in \mathsf{Adv}(Q)} P_n{}^A(s \Rightarrow S^T)$$

3.3.2 時間確率システム

定義 6 [3], [6] 時間確率システム（Timed Probabilistic System, TPS）は，組 $\langle S, Steps \rangle$ である．ここで，S は状態集合であり，$Steps$ は以下の式で表される遷移関係である．

$$Steps \subseteq S \times \mathbf{R}_{\geq 0} \times \mathsf{Dist}(S)$$

ある遷移 $(s, d, \mu) \in Steps$ において，d は状態 s にとどまっている時間を表す．遷移の種類には，離散的な遷移と時間的な遷移があり，以下に示すルールがある．

- 時間遷移：$s \xrightarrow{d,\cdot} t$
  μ は点分布であり，状態 s から d だけ時間が経過して状態 t につねに確率 1 で遷移することを意味している．$s \xrightarrow{d,\cdot} t$ かつ $s \xrightarrow{d,\cdot} t'$ ならば，$t = t'$ が成り立つ．
- 離散遷移：$t \xrightarrow{0,\mu} w$
  $d = 0$ であり，確率分布 μ に従って，ある確率で状態 w に遷移する．

時間遷移の後に続けて離散遷移が起きるとき．形式的に次のように書く．

$$s \xrightarrow{d,\cdot} t \xrightarrow{0,\mu} w \iff s \xrightarrow{d,\mu} w$$

時間確率システムの動作は，確率システムと同様に，次のような無限パス ω で表現できる．

$$\omega = s_0 \xrightarrow{d_0,\mu_0} s_1 \xrightarrow{d_1,\mu_1} s_2 \xrightarrow{d_2,\mu_2} \cdots$$

ここで，$(s_i, d_i, \mu_i) \in Steps$ であり，すべての $i \in \mathbf{N}$ に対して，$\mu_i(s_{i+1}) > 0$ である．時間確率システムにおけるパスやアドバサリおよび最大到達確率は，確率システムと同様に定義できる．

定義 7 時間確率システム TPS $\langle S, Steps \rangle$ のアドバサリ A は，時間確率システム上の有限パス ω_{fin} を確率分布 μ に写像する．すなわち，

$$A(\omega_{fin}) = \mu$$

ただし，$(\mathsf{last}(\omega_{fin}), d, \mu) \in Steps$ である．

3.3.3 確率線形ハイブリッドオートマトンの意味

PLHA $M = \langle L, l, X, I, \Gamma, \Sigma, T \rangle$ の意味は，時間確率システム TPS $\langle S, Steps \rangle$ である．ここで，

$$S = \{(l, v) \mid l \in L \wedge v \in [\![I(l)]\!]\}$$

$$Steps \subseteq S \times \mathbf{R}_{\geq 0} \times \mathsf{Dist}(S)$$

TPS の初期状態は，$s_{init} = (l, \mathbf{0})$ である．$((l, v), d, \mu) \in Steps$ であるとき，以下のいずれかの条件を満たす．

- 時間遷移
  $v + \Gamma(l)d \in [\![I(l)]\!]$ かつ $\mu(l, v + \Gamma(l)d) = 1$
- 時間&離散遷移
  $\exists (l, g, a, p) \in T. v \in [\![g]\!]$ かつ $\forall (l', v') \in S : \mu(l', v') = \sum_{u \in \mathsf{support}(p) \wedge v' = (v + \Gamma(l)d)[u]} p(u, l')$

図 1 に示す PLHA の動作例を説明する．ここでは，$N = 1$ と仮定する．遷移元がない矢印は，s_0 が初期ロケーションであることを表している．したがって，初期状態は $(s_0, x = 0, e = 0, try = 0)$ である．ここで，時間が 1 だけ経過したとすると，状態は $(s_0, x = 1, e = 2.5, try = 0)$ となる．x は時間に同期し，e は与えられた傾きに従って 2.5 だけ増加する．このとき，ガード条件 $x \geq 1 \wedge try \leq N$ を満たすので，アクション send を持つ遷移が可能である．そして確率分布に従って，0.1 の確率で，状態 $(s_1, x = 0, e = 2.5, try = 1)$ に遷移し，0.9 の確率で，状態 $(s_3, x = 1, e = 2.5, try = 0)$ へ遷移する．ロケーションが s_2 や s_3 である状態に到達した場合は，そのロケーションにとどまり，時間遷移し続けることになる．

4. 到達可能性検証手法

この章では，本研究が対象としている確率線形ハイブリッドオートマトンに対する確率到達可能性問題を定義する．また，凸多面体表現により，無限に存在するシステムの状態を記号化するための記号状態を導入し，それを用いた到達可能性検証手法を述べる．

4.1 確率到達可能性問題

確率システムに対する到達可能性検証では，目的状態に

どのくらいの確率で到達するかという不確実性を検証する要素がある．本研究では，Berendsen らの論文 [3] で定義されているコスト境界付き確率到達可能性問題を変更した事後条件付きの確率到達可能性問題を考える．

定義 8 （事後条件付き確率到達可能性問題）
PLHA $M = \langle L, l, X, I, \Gamma, \Sigma, T\rangle$，目的ロケーション $l_{target} \in L$，確率 $\lambda \in [0,1]$，および事後条件 c の組 $\langle M, l_{target}, \lambda, c\rangle$ を事後条件付き確率到達可能性問題と呼ぶ．形式的には，以下のように定義される．

“$s_{init} = (l, \mathbf{0})$，$S^T = \{(l,v) \mid l = l_{target} \wedge v \in [\![c]\!]\}$ に対して，$\mathsf{ProbReach}^A(s, S^T) > \lambda$ となるアドバサリ $A \in \mathsf{Adv}([\![M]\!])$ が存在するか．”

ここで，s_{init} は TPS の初期状態であり，S^T は目的状態集合である．

事後条件付き確率到達可能性問題において，目的状態をエラー状態など望ましくない状態と見なし，問題に対する答えが “Yes” であるとき，そのシステムは安全ではないということが証明できる．すなわち，安全性の検証が可能になる．5 章では，安全性検証の事例を示している．

4.2 記号状態

時間オートマトンは状態として実数値をとるため，状態数が無限にあり，計算機上では状態を表現できない．記号状態とは，数式で変数のとりうる範囲をまとめたものである．記号状態により，システムの時間経過が記号化され，状態集合を計算機で扱うことが可能になる．確率線形ハイブリッドオートマトンにおける記号状態は凸多面体で表現できるため，ガード条件を用いて定義する．

定義 9 PLHA $M = \langle L, l, X, I, \Gamma, \Sigma, T\rangle$ の記号状態を $\sigma = (l, Z)$ と定義する．ここで，$l \in L, Z \in \mathsf{Guard}(X)$ である．記号状態の意味は，$[\![\sigma]\!] = \{l\} \times [\![Z]\!]$ である．つまり，ロケーション l において，条件 Z を満たす状態集合を表す．

記号状態により，同一ロケーションにおける時間遷移を数式により記号的に表現できる．記号状態上での時間遷移や離散遷移を計算するために，凸多面体に対する演算を定義する．

$$[\![Z^{\downarrow\gamma}]\!] \stackrel{\mathsf{def}}{=} \{v \in \mathbf{R}^X_{\geq 0} \mid \exists d \geq 0. v + \gamma d \in [\![Z]\!]\}$$
$$[\![[u]Z]\!] \stackrel{\mathsf{def}}{=} \{v \in \mathbf{R}^X_{\geq 0} \mid [\![v[u]]\!] \in [\![Z]\!]\}$$

$Z^{\downarrow\gamma}$ は，変数に勾配を割り当てる関数 γ に従って時間経過することで，Z に到達できる条件を表している．$[u]Z$ は，変数更新関数 u により各変数の値が更新されたときに，Z を満たすことができる条件を表している．これらの演算は，ガード条件の表現を崩すことなく計算可能である．

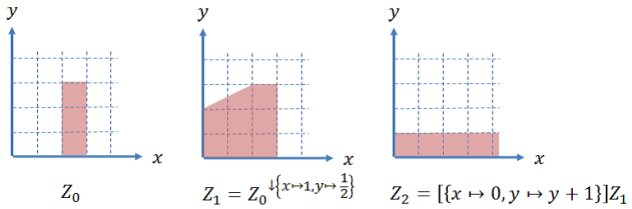

図 2 凸多面体に対する演算
Fig. 2 Operations on convex polyhedra.

図 2 に示すような具体例とその計算方法を示す．$Z_0 = 2 \leq x \leq 3 \wedge 0 \leq y \leq 3$，関数 γ の集合 $\{x \mapsto 1, y \mapsto \frac{1}{2}\}$，関数 u の集合 $\{x \mapsto 0, y \mapsto y+1\}$ とする．定義に従うと，$Z_0^{\downarrow\gamma}$ は以下の数式で表現できる．なお，非負条件は，本論文で用いる変数の定義によるものである．

$$(\exists d \geq 0.2 \leq x + d \leq 3 \wedge 0 \leq y + \frac{1}{2}d \leq 3) \wedge x \geq 0 \wedge y \geq 0$$

これは線形不等式に対する一階述語論理となる．このような論理式から限量子記号を消去することを QE（Quantifier Elimination）という．QE の手法については，文献 [5], [9] などで述べられており，Fourier-Motzkin Elimination としても知られている [18].

定理 1 Fourier-Motzkin Elimination
$a_i, b_j, c_i, d_j \in \mathbf{R}$，$a_i > 0$，$b_j > 0$ のとき，以下が成り立つ．

$$\exists x.(\bigwedge_i c_i \leq a_i x) \wedge (\bigwedge_j b_j x \leq d_j) \equiv \bigwedge_{i,j} b_j c_i \leq a_i d_j$$

Fourier-Motzkin Elimination により，

$$\begin{aligned} Z_1 &\equiv (\exists d \geq 0.2 \leq x + d \leq 3 \wedge 0 \leq y + \frac{1}{2}d \leq 3) \\ &\quad \wedge x \geq 0 \wedge y \geq 0 \\ &\equiv (0 \leq -x + 3 \wedge 0 \leq -y + 3 \wedge -x + 2 \leq -x + 3 \\ &\quad \wedge -x + 2 \leq -2y + 6 \wedge -y \leq -y + 3) \\ &\quad \wedge x \geq 0 \wedge y \geq 0 \\ &\equiv 0 \leq x \leq 3 \wedge 0 \leq y \leq 3 \wedge y \leq \frac{1}{2}x + 2 \end{aligned}$$

となり，図 2 の Z_1 と一致する．Z_1 は，時間経過の後，Z_0 を満たす変数の評価関数の集合を意味する．$[u]Z_1$ は，更新関数 u に従って，Z_1 中に現れる変数を一次式に置換することで計算できる．つまり，

$$\begin{aligned} Z_2 &\equiv (0 \leq 0 \leq 3 \wedge 0 \leq y + 1 \leq 3 \wedge y + 1 \leq 2) \\ &\quad \wedge x \geq 0 \wedge y \geq 0 \\ &\equiv 0 \leq x \wedge y \leq 1 \end{aligned}$$

となる．この例で Z_2 が意味するのは，Z_2 を x を 0 にリセットし，かつ y を 1 増加させた後に Z_1 を満たす変数の評価関数の集合である．

次章で示す到達可能性検証 procedure は目的状態から開始して後方探索を行う．このとき，ロケーション不変条件

およびエッジのガード条件を考慮した後方演算を行う必要がある．

定義 10 （記号状態に対する後方演算）
$M = \langle L, l, X, I, \Gamma, \Sigma, T\rangle$，$l \in L, Z \in \mathsf{Guard}(X)$，$e := (l, g, a, p, u, l') \in \mathsf{edges}(M)$ のとき，

$$tpre(l, Z) = (l, Z^{\downarrow\Gamma(l)} \wedge I(l))$$
$$dpre_e(l', Z) = (l, [u]Z \wedge g \wedge I(l))$$

4.3 Procedure

事後条件付き確率到達可能性問題を解くための procedure を示す．この procedure は，コスト付き確率時間オートマトンを対象とする CBPRalg [3] をベースに，確率線形ハイブリッドオートマトンに適用できるように拡張したものである．

Procedure 1 の説明をする．目的記号状態 σ_{target} は，目的ロケーション l_{target} および事後条件 c によって決定される（2 行目）．もし，目的記号状態に初期状態が含まれていれば，到達確率は 1 となる（6–7 行目）．この procedure では，σ_{target} から始めて，n 回の離散遷移によって，目的状態へと到達可能な記号状態集合を幅優先探索で求めている．探索途中に，ある深さで到達確率が λ を超えた場合，procedure は “Yes” を出力する（14–15 行目）．また，すべての状態を探索しても到達確率が λ を超えなかった場合は，“No” を出力する（17–18 行目）．21–25 行目では，過去 1 回分の遷移を使って，次に探索すべき状態集合を求めている．詳しくは，Procedure 2 で説明する．探索済みの状態から構成した確率システム（定義 11 を参照）において，目的状態への “最大到達確率” を求める（26–27 行目）．なぜならば，到達確率が検証確率 λ を “超える” アドバサリに興味があるからである．したがって，より大きな到達確率を与えるアドバサリを見つけるため，最大到達確率を求めている．

Procedure 2 は，記号状態 τ と τ に到達できる遷移 e が与えられたとき，τ に到達できる記号状態集合を求める procedure である．遷移 e を用いて，τ に到達できる状態 σ を求め（2 行目），σ が空集合でなければ，次に探索する状態集合に加える．さらに，同じ確率分布を持つ遷移関係において，遷移元の記号状態の共通部分を考えることで，その確率分布を選択した結果生じる複数のパスを得ることができる（10–19 行目）．以上のように計算された記号状態集合とその遷移の集合の組を記号状態グラフと呼ぶ．

以下に，procedure 1 の計算量について述べる．procedure 1 では，目的状態から後方探索により，確率線形ハイブリッドオートマトンから記号状態グラフを構成するものであるが，計算量の面から，以下の特徴がある．
(1) まず，13 行のループは無限ループとなっており，アル

Procedure 1 Symbolic Reachability Analysis for PLHA
入力：事後条件付き確率到達可能性問題 $\langle M, l_{target}, \lambda, c\rangle$．ただし，$M = \langle L, l, X, I, \Gamma, \Sigma, T\rangle$
出力：“Yes” or “No”

```
 1: s_init := (l, 0)
 2: σ_target := [[(l_target, I(l_target) ∧ c)]]
 3: for each t ∈ T do
 4:     E_t := ∅
 5: end for
 6: if s_init ∈ σ_target then
 7:     R_0 := 1
 8: else
 9:     R_0 := 0
10: end if
11: Waiting_0 := {σ_target}
12: Visited := {σ_target}
13: for n = 1 to ∞ do
14:     if R_{n-1} > λ then
15:         return “Yes”
16:     end if
17:     if Waiting_{n-1} = ∅ then
18:         return “No”
19:     end if
20:     Waiting_n := ∅
21:     for each τ ∈ Waiting_{n-1} do
22:         for e = (l, g, a, p, u, l') ∈ edges(M), with τ = (l', ·) do
23:             Waiting_n := Waiting_n ∪ ComputePredecessor(τ, e)
24:         end for
25:     end for
26:     Q_n := ConstructSPS(Visited, E, σ_target)
27:     R_n := max_{σ∈Visited∧s_init∈[[tpre(σ)]]} MaxProbReach_{Q_n}(σ, {σ_target})
28: end for
```

Procedure 2 ComputePredecessor
入力：記号状態 τ，PLHA の遷移関係 $e = (t, f)$，ただし，$t = (l, g, a, p), f = (u, l')$
出力：predecessor の集合

```
 1: P := ∅
 2: σ := [[dpre_e(tpre(τ))]]
 3: if σ ≠ ∅ then
 4:     if σ ∉ Visited then
 5:         Visited := Visited ∪ {σ}
 6:         P := P ∪ {σ}
 7:     end if
 8:     E_t := E_t ∪ {(σ, f, τ)}
 9:     Add := ∅
10:     for each (σ', f', τ') ∈ E_t do
11:         if σ ∩ σ' ≠ ∅ ∧ f ≠ f' then
12:             if σ ∩ σ' ∉ Visited then
13:                 Visited := Visited ∪ {σ ∩ σ'}
14:                 P := P ∪ {σ ∩ σ'}
15:             end if
16:             Add := Add ∪ {(σ ∩ σ', f, τ), (σ ∩ σ', f', τ)}
17:         end if
18:     end for
19:     E_t := E_t ∪ Add
20: end if
21: return P
```

ゴリズムの停止性が保証されない．これは記号状態グラフの記号状態が無限となる場合があることによる．

(2) 次に，23 行目で用いる Fourier-Motzkin Elimination の計算量の下界は $2^{2^{\Omega(n)}}$ であることが知られている．ただし，n は量化記号の数である．

定義 11 ConstructSPS（記号状態グラフから SPS への変換）

記号状態グラフ $< V, E >$ および s_{target} が与えられたとき，対応する SPS は，$Q =< V, Steps >$ となる．ここで，$Steps$ は以下のように定義する．

- $(\sigma, \pi) \in Steps$ かつそのときに限り，以下のどちらかの条件を満たす．
  - $\sigma = s_{target} \wedge \pi = \{\sigma \mapsto 1\}$
  - $\exists E_\pi \subseteq E$ s.t.
    - $\forall(\sigma', \cdot, \cdot) \in E_\pi : \sigma = \sigma$
    - $\forall(\cdot, f, \tau), (\cdot, f', \tau') \in E_\pi : \tau \neq \tau' \implies f \neq f'$
    - E_π is maximal
    - $\forall \tau \in V : \pi(\tau) = \sum\{(p(f) \mid (\cdot, f, \tau)\}$

定理 2 定義 11 に従って構築された確率システム $Q_n =< V_n, Steps_n >$ における最大到達確率は，確率線形ハイブリッドオートマトン $M = \langle L, l, X, I, \Gamma, \Sigma, T\rangle$ の意味 $[\![M]\!] = \langle S, Steps\rangle$ における最大到達確率と等しい．つまり，以下が成り立つ．

任意の状態 s と procedure 1 の 13–25 行の繰返し回数 n に対して，

$$P_n{}^{max}(s \Rightarrow [\![S^T]\!]) = \max_{v \in V_n, s \in [\![tpre(v)]\!]} (v \Rightarrow S^T)$$

ここで，目的状態集合 $S^T \subseteq S$ として，procedure 1 の 13–25 行の繰返し回数 n で構成される確率システムを $Q_n =< V_n, Steps_n >$ とする．なお，右辺の式は procedure 1 の 27 行目の R_n である．

（証明の概要）

以下の証明は，確率時間オートマトン [17] およびコスト付き確率時間オートマトン [3] の最大到達確率計算の正当性の証明とまったく同様であり，紙面の都合上から主要部分のみを説明する．コスト付き確率時間オートマトン [3] は各ロケーションに 1 つのみのコスト変数が記述可能であり，一方，本論文の確率線形ハイブリッドオートマトンは複数のコスト変数が記述可能である．なお，コスト変数とは時間微分係数がクロック変数と異なる変数である．このコスト変数の数の違いは，ロケーションにおける時間前進演算子 $tpre$ においてコスト変数が 1 つか複数かの違いのみであり，これにより確率線形ハイブリッドオートマトンの変数が構成する凸多面体はコスト付き確率時間オートマトンの凸多面体とは異なる．ゆえに，本論文の確率線形ハイブリッドオートマトンの検証 procedure とコスト付き確率時間オートマトンの検証 procedure [3] との違いは時間前進演算子 $tpre$ のみである．

証明に主要部は以下の部分からなる．

まず，以下を証明する．

(a) すべての $n \in N$，$k \in N$，$A_Q \in Adv(Q_k)$，$v \in V_k$ と $s \in [\![tpre(v)]\!]$ に対して，以下のような $A_M \in Adv([\![M]\!])$ が存在する．

$$P_n{}^{A_M}(s \Rightarrow [\![S^T]\!]) \geq P_n{}^{A_Q}(v \Rightarrow S^T)$$

これは時間前進演算子 $tpre$ に注意して，n に関する帰納法で容易に証明できる．

(b) すべての $n \in N$，$A_M \in Adv([\![M]\!])$，$s \in S$ に対して，以下のような $v \in V_n$ と $A_Q \in Adv(Q_n)$ が存在する．

$$P_n{}^{A_Q}(v \Rightarrow S^T) \geq P_n{}^{A_M}(s \Rightarrow [\![S^T]\!])$$

これは時間前進演算子 $tpre$ に注意して，n に関する帰納法で容易に証明できる．

次に，以下を導く．

(c) (a) を使って，すべての $n \in N$，$k \in N$ と以下の

$$b \in \{P_n{}^{A_Q}(v \Rightarrow S^T) | A_Q \in Adv(Q_k)\}$$

より，以下が得られる．

$$sup\{P_n{}^{A_M}(s \Rightarrow [\![S^T]\!]) | A_M \in Adv([\![M]\!])\}$$

(d) (b) を使って，すべての $n \in N$，$s \in S$ と以下の

$$a \in \{P_n{}^{A_M}(s \Rightarrow [\![S^T]\!]) | A_M \in Adv(Q_M)\}$$

より，以下が得られる．

$$sup\{P_n{}^{A_Q}(v \Rightarrow S^T) | A_Q \in Adv(Q_n)\}$$

ここで，sup と max の定義より，以下の等式が得られる．

$$sup\{P_n{}^{A_Q}(v \Rightarrow S^T) | A_Q \in Adv(Q_k)\}$$
$$= \max_{v \in V_n, s \in [\![tpre(v)]\!]} (sup\{P_n{}^{A_Q}(v \Rightarrow S^T) | A_Q \in Adv(Q_k)\})$$

上の等式で，$k = n$ とおくと，証明が終わる．

（証明終わり）

確率システムの最大到達確率を求める手法として，政策反復法，値反復法，線形計画法が知られている．本研究の実装では，線形計画法であるシンプレックス法によって最大到達確率を計算している．

確率線形ハイブリッドオートマトンのサブクラスである線形ハイブリッドオートマトンに対する到達可能性問題は，決定不能であることが示されている [10]．したがって，本論文で示した procedure にも停止性は保証できない．ただし，Berendsen らの論文 [3] でも示されているように，この procedure が停止したならば，その結果は正しいものである．

本論文の確率線形ハイブリッドオートマトンのモデル検査に関する停止性について知られていることを述べる．

(1) まず，ハイブリッドオートマトンのモデル検査については，Alur らは停止性が保証されないことを初めて示した [10]．その後，Pnueli らが線形ハイブリッドオートマトンにおいて，ループ内で変数のガード条件を含めば停止性が保証されないことを示した [12]．また，最近では，Pnueli や Asarin らがハイブリッドオートマトンのモデル検査の計算可能性に関して，より深い考察を与えている [13]．一方，Alur らが線形ハイブリッドオートマトンのサブクラスであるコスト付き時間オートマトンの最小コスト到達可能性のモデル検査の決定可能性を示している [14]．

(2) 次に，確率線形ハイブリッドオートマトンのモデル検査に関する停止性については，Kwiatkowska らが確率時間オートマトンの最大確率到達性のモデル検査の決定可能性を示した [6]．同時に，Sproston が確率矩形ハイブリッドオートマトンのモデル検査の決定可能性を示した [15]．その後，Berendsen らがコスト付き確率時間オートマトンのコスト境界確率到達可能性（コスト付き最大確率到達性）のモデル検査は停止性が保証されないことを示して，その検証 procedure を提案した [3]．Berendsen らの検証 procedure は有限の記号状態グラフが構成できる場合または，有限の記号状態グラフで検証結果が判明できる場合には検証結果が得られる．有限の記号状態グラフが得られる条件はまだ知られていない [3]．

本論文の検証 procedure は Berendsen らの検証 procedure をもとにしたものであり，有限の記号状態グラフが得られる条件はまだ知られていない．

5. 実験

事後条件付き確率到達可能性問題を検証する procedure を計算機上に実装した検証器を用いて，確率線形ハイブリッドオートマトンによって記述したモデルの検証実験を行った．

5.1 検証器の実装

プログラミング言語 C++により，検証器を計算機上に実装した．ソースコードの行数は約 3,000 行である．開発・実験環境は以下のとおりである．

- OS：Microsoft Windows 7 Home Premium
- プロセッサ：Intel Core i3 CPU 530 2.93 GHz
- メモリ：2.00 GB RAM
- 開発環境：Microsoft Visual C++ 2010
- 外部ライブラリ：MPIR 2.4.0 [19]

MPIR は多倍長整数（有理数）演算のためのライブラリで，ガード条件の係数や確率計算など検証器全般で使用している．本検証器には，凸多面体上の演算やシンプレックス法の実装が含まれている．

5.2 事例：無線センサネットワーク

無線センサネットワーク（Wireless Sensor Network, WSN）とは，物理量を検知するセンサおよび無線通信機器を搭載した小型デバイス（センサノード）が自律して構成するネットワークのことである．天候・災害などの情報伝達や，住宅街の監視システムなどに応用されることが期待されている．WSN の特徴として，ノードが外部電源を必要としないバッテリ駆動であることがあげられる．

5.2.1 モデル化

短距離無線通信規格の 1 つである ZigBee では，3 種類の通信端末が存在する [22]．

- ZigBee Coordinator — ネットワーク全体を制御する．
- ZigBee Router — データ中継機能を持つため，ネットワークを拡大できる．
- ZigBee End Device — データ中継機能を持たない端末

ZigBee による WSN の実現を考えたとき，センサノードと呼ばれる端末は，ZigBee End Device となる．ここでは，データの中継は考えずに，**図 3** のようなスター型トポロジのネットワーク構成を考える．センサノードから送信されるデータを受信するための（ZigBee Coordinator に相当する）コーディネータを配置し，その周りにセンサノードを配置する．コーディネータは，センサノードから送信されてくる情報をつねに待ち続ける．センサノードは，送信状態と待ち状態を切り替えながら，コーディネータにデータ送信を試みる．

コーディネータの PLHA によるモデル C を**図 4** に示す．コーディネータの消費電力については考慮しないものとする．図中には，data_i というアクションを持つ遷移が書かれている．これは，周囲に存在するセンサノードの数

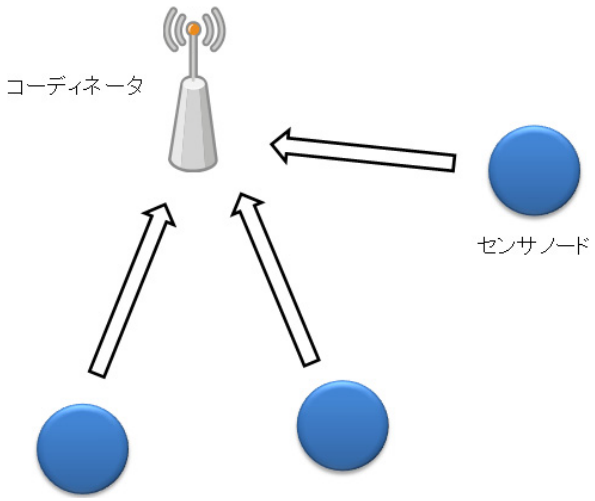

図 3 無線センサネットワーク（スター型トポロジ）
Fig. 3 Wireless sensor network (star topology).

表 1 WSN モデル検証実験の結果
Table 1 Result of verification of WSN models.

n	K	$\|S\|$	$\|Steps\|$	Time [s]	Reachability (max probability)
1	5	2	2	0.011	true (1)
2		7	7	0.042	true (1)
3		40	40	0.808	true (1)
1	6	4	4	0.069	true (0.1)
2		476	637	2068.52	false (0.01)
3		≥ 2908	≥ 3340	-	- (≥ 0)

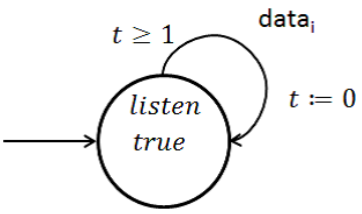

図 4 コーディネータの PLHA モデル C
Fig. 4 PLHA C which models coordinator.

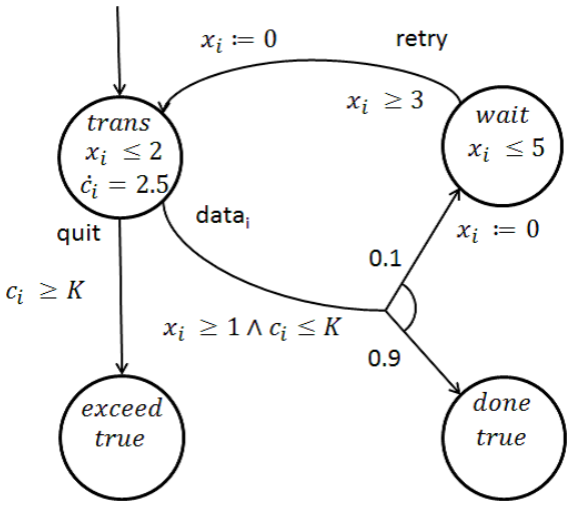

図 5 センサノードの PLHA モデル D_i
Fig. 5 PLHA D_i which models sensor node.

を n 個とすると，data_1〜data_n のアクションをそれぞれ持つ遷移があることを意味する．

センサノードの PLHA モデル D_i を**図 5** に示す．センサノードは，送信状態では経過時間に比例して電力を消費する．一定確率でデータ送信に失敗し，待機状態に遷移する．その後，再び送信状態に遷移する．もし，累積消費電力が K に達したら，それ以上通信ができない状態 exceed に遷移する．ここで，K はバッテリ残量を表す設計パラメータである．WSN は，これらの PLHA の並列合成でモデル化できる．つまり，検証対象となる PLHA は $M = C \parallel D_1 \parallel \cdots \parallel D_n$ である．

検証性質

安井らの論文 [16] では，コスト付き確率時間オートマトンにより無線センサネットワークをモデル化し，「“すべてのノードが通信を成功させる前に”，バッテリを使い果たすかどうか」を検証している．ただし，これはすべてのノードの合計コストに関する検証である．コスト付き確率時間オートマトンでは，コストを 1 つしか扱えないため，個々のノードのパラメータに対する検証を行うことができないからである．本研究で定義した確率線形ハイブリッドオートマトンによるモデルでは，センサノードにそれぞれコストを持たせることが可能であるため，検証問題として，「“すべてのノードが”，通信を終える前にバッテリを使い果たすかどうか」を考える．つまり，すべてのセンサノードが，ロケーション exceed に到達するかどうかを検証する．検証確率は 3%，事後条件は *true* とする．

5.2.2 実験結果

センサノードの数 n および設計パラメータ K を変えていくつかの検証実験を行った．その結果を**表 1** に示す．$|S|$，$|Steps|$ は，検証終了時の状態数および確率分布の数をそれぞれ表している．$K = 5$ のケースでは，n が増えても比較的はやく結果が得られている．これは，少ない離散遷移数で目的状態に到達可能だからである．$K = 6$ のケースでは，探索する状態空間が，$n = 2$ 以降で爆発的に増えている．消費可能なコスト K が増えたことにより，可能な動作（trans → wait → trans のループ）が増えることが原因と考えられる．特に，目的状態に到達しない場合は，すべての動作を探索する必要があるため，検証コストが増大する．$n = 2$，$K = 6$ のケースにおいて，検証確率を 0 とした場合は，状態数 79，確率分布数 87，検証時間 16.677 [s] で *true* を出力した．$n = 3$，$K = 6$ のケースについては，さらに状態数が増加し，検証が終了していない．

なお，この例題に関連する検証 procedure が停止するための条件は分かっていない．

より多くのケースを実験するためには，検証器の性能の向上が必要である．コスト付き確率時間オートマトンのモデル検査器 Fortuna [4] では，いくつかの最適化手法が提案・実装されている．別のアプローチとしては，述語抽象化による近似解法 [23] があげられる．

6. おわりに

本研究では，システム記述言語である確率線形ハイブリッドオートマトンに対する確率到達可能性問題の検証手法を提案した．コスト付き確率時間オートマトンに対する

到達可能性検証手法を，確率線形ハイブリッドオートマトンに適用できるようにprocedureの拡張を行い，検証ツールを開発した．そして，確率線形ハイブリッドオートマトンの構成要素であるリアルタイム性，物理量，確率的動作を含むシステムとして，無線センサネットワークを例に，システムの記述から検証までを行い，計算機上での自動検証が可能であることを示して，提案手法の有効性を実証した．具体的には，従来のコスト付き確率時間オートマトンでは，コストを1つしか扱えないため，個々のノードのパラメータに対する検証を行うことができなかったが，本研究で定義した確率線形ハイブリッドオートマトンによるモデルでは，センサノードにそれぞれコストを持たせることが可能となった．

今後の課題としては，procedureの最適化による検証器の性能向上や，状態数を削減するアプローチとして，述語抽象化による近似手法の導入が考えられる．

参考文献

[1] Bianco, A. and de Alfaro, L.: Model Checking of Probabilistic and Nondeterministic Systems, LNCS 1026, pp.499–513 (1995).

[2] Bengtsson, J. and Yi, W.: Timied Automata: Semantics, Algorithms and Tools, LNCS 3098, pp.87–124 (2003).

[3] Berendsen, J., Jansen, D.N. and Katoen, J.-P.: Probably on Time and within Budget: On Reachability in Priced Probabilistic Timed Automata, *QEST '06*, pp.311–322, IEEE Computer Society (2006).

[4] Berendsen, J., Jansen, D.N. and Vaandrager, F.W.: Fortuna: Model Checking Priced Probabilistic Timied Automata, Technical Report, ICIS, Radboud University Nijmegen (2009).

[5] Ferrante, J. and Rackoff, C.: A Decision Procedure for the First-Order Theory of Real Addition with Order, *SIAM Journal on Computing*, Vol.4, No.1, pp.69–76 (1975).

[6] Kwiatkowska, M., Norman, G., Segala, R. and Sproston, J.: Automatic verification of real-time systems with discrete probability distributions, *Theoretical Computer Science*, Vol.282, pp.101–150 (2002).

[7] Kwiatkowska, M., Norman, G., Sproston, J. and Wang, F.: Symbolic model checking for probabilistic timed automata, *Information and Computation*, Vol.205, pp.1027–1077 (2007).

[8] Kwiatkowska, M., Norman, G. and Parker, D.: PRISM 4.0: Verification of Probabilistic Real-Time Systems, *Proc. 23rd International Conference on Computer Aided Verification* (*CAV'11*), LNCS 6806, pp.585–591 (2011).

[9] Alur, R., Henzinger, T.A. and Ho, P.-H.: Automatic symbolic verification of embedded systems, IEEE Trans. Softw. Eng., Vol.22, No.3, pp.181–201 (1996).

[10] Alur, R.: The Algorithmic Analysis of Hybrid Systems, *Theoretical Computer Science*, Vol.138, pp.3–34 (1995).

[11] Mutsuda, Y., Kato, T. and Yamane, S.: Symbolic Reachability Analysis of Probabilistic Linear Hybrid Automata, *IEICE Trans. Fundamentals*, Vol.E88-A, No.11, pp.2972–2981 (2005).

[12] Kesten, Y., Pnueli, A., Sifakis, J. and Yovine, S.: Decidable Integration Graphs, *Information and Computation*, Vol.150, No.2, pp.209–243 (1999).

[13] Asarin, E., Mysore, V., Pnueli, A. and Schneider, G.: Low dimensional hybrid systems – Decidable, undecidable, don't know, *Information and Computation*, Vol.211, pp.138–159 (2012).

[14] Alur, R., La Torre, S. and Pappas, G.J.: Optimal Paths in Weighted Timed Automata, LNCS 2034, pp.49–62 (2001).

[15] Sproston, J.: Decidable model checking of probabilistic hybrid automata, LNCS 1926, pp.31–45 (2000).

[16] 安井雅俊，山根 智：コスト付き確率時間オートマトンの抽象化精錬を用いた到達可能性解析手法，京都大学数理解析研究所講究録 1691，pp.15–21 (2010).

[17] Kwiatkowska, M., Norman, G. and Sproston, J.: Symbolic Model Checking for Probabilistic Timed Automata, Technical report CSR-03-10, School of Computer Science, University of Birmingham (Oct. 2003).

[18] Niels Lauritzen. Lectures on Convex Sets (2010), available from ⟨http://home.imf.au.dk/niels/lecconset.pdf⟩.

[19] MPIR (Multiple Precision Integers and Rationals), available from ⟨http://mpir.org/⟩.

[20] PRISM - Probabilistic Symbolic Model Checker, available from ⟨http://www.prismmodelchecker.org/⟩.

[21] UPPAAL, available from ⟨http://www.uppaal.com/⟩.

[22] ZigBee Alliance, available from ⟨http://www.zigbee.org/⟩.

[23] 清水隆也，森下 篤，山根 智：確率時間 CEGAR の開発とその実証実験，情報処理学会論文誌 プログラミング，Vol.5, No.2, pp.43–66 (2012).

畠中 克也 （正会員）

2012年3月金沢大学大学院自然科学研究科電子情報科学専攻修了．同年キヤノンイメージングシステムズ株式会社入社．在学中はセンサネットワークの仕様記述とモデル検査の研究に従事．特に，確率時間オートマトンのモデル検査の研究に従事．

山根 智 （正会員）

1984年3月京都大学大学院修士課程修了．現在，金沢大学理工研究域電子情報学系教授．博士（京都大学）．組込みシステムおよびクラウドコンピュータ等の仕様記述，解析，モデル検査等の研究に従事．EATCS，ACM，IEEE，LA，日本ソフトウェア科学会各会員．