Vol.2014-EMB-34 No.1
2014/9/17

電力消費振る舞いのロジック・モデル検査

中島 震[1,2,a)]

概要：モデルベース電力消費解析では、電力消費に関わる振る舞い仕様と検査性質を表現する形式言語が中心的な役割を果たす。振る舞い仕様表現として重み付き時間オートマトン、検査性質の表現として凍結限量子を持つ重み時相論理が提案されている。本稿では、そのモデル検査の方法を考察する。モデル検査は決定不能であることから、時間有界な範囲での検査に限定し、また、連続時間に対する近似法としてサンプリング抽象を用いる。Realtime Maude 上で解析することを目的として、検査性質を表す時相論理のサブセットを定義した。

1. はじめに

スマートフォンなどの小型軽量システムでは、アプリケーション・プログラムに起因する電力消費が大きな問題となる。機能振る舞いが期待通りであっても、電力消費が想定外であれば継続的に稼働できない。システムの実行時に消費電力を測定する方法が知られている。一方で、設計時の検討不足に起因する不具合も多い。システム開発の上流工程で原因を除去するモデルベース解析手法（例えば [9]）の確立が期待されている。

モデルベース解析では、電力消費に関わる振る舞い仕様と検査性質の表現を厳密に行う方法が中心的な役割を果たす。そのような形式仕様言語として、電力消費オートマトン (PCA) ならびに凍結限量子を持つ時相論理 (fWLTL) が提案された [11]。この時、電力消費解析は、ロジック・モデル検査の問題となる。しかし、PCA は重み付き時間オートマトンであり、fWLTL は TPTL を拡張した時相論理なので、モデル検査問題は決定不能である。自動検証を目指すには、何らかの近似手法を導入せざるを得ない。

本稿では、PCA と fWLTL のロジック・モデル検査の方法を考察する。検証エンジンとして、Realtime Maude[12] を利用することを前提として、時間有界モデル検査にサンプリング抽象を組み合わせる方法を検討する。凍結限量子を除去することを目的として、fWLTL を構文的に制限した $fWLTL^S$ を用いる。以下、第２節で電力消費振る舞いのモデル検査問題を整理する。第３節で Realtime Maude への変換法を提案する。第４節で関連研究と比較し、第５節で今後の研究方向を論じる。また、付録に Realtime Maude の概要を掲載した。

1 国立情報学研究所, National Institute of Informatics
2 総合研究大学院大学, SOKENDAI
a) nkjm@nii.ac.jp

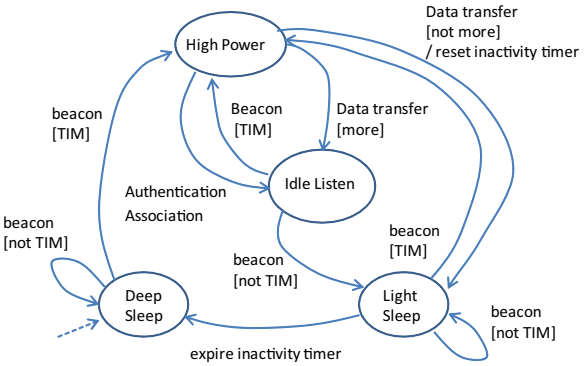

図 1 Wi-Fi STA の振る舞い

2. 電力消費振る舞いの形式化

文献 [11] にしたがって電力消費振る舞いと検査性質の表現の方法を述べる。

2.1 振る舞い仕様の表現

2.1.1 PCA

電力消費オートマトン (Power Consumption Automaton, PCA と略記) のダイアグラム表現例を図 1 に示す。各状態で単位時間あたりの消費電力が数値的に与えられており、その状態での滞留時間がわかれば消費電力を求めることができる。状態間は、信号受信やタイムアウト等のイベントで遷移する。状態の遷移系列を求め、各状態での消費電力を加算することで、その遷移系列実行時の消費電力総量を計算することができる。単位時間あたりの消費電力を重みとすると、PCA を重み付き時間オートマトンとみなすことができる。以下、PCA $\mathcal{A}$ を、アトミックな命題の有限集合 $Prop$ に対して、次のように定義する。

$\langle\ Loc, C, D, \Sigma\cup\{\epsilon\}, Edg, Flow, Inv, Lab\ \rangle$

以下、構成要素を説明する。

(1) Loc は、ロケーションの有限集合。

(2) C はクロック変数の集合、 D は重み変数の集合。C と D は互いに素であり、$C\cap D=\emptyset$。クロック変数 x、自然数 n、オペレータ $\bowtie\in\{<,\leq,=,\geq,>\}$ に対して、$x\bowtie n$、$x_1-x_2\bowtie n$、の形をした制約式の集合をクロック制約 Z(C) と呼ぶ。

(3) Σ は、入力記号の有限集合であり、 ϵ は記号が空の場合を示す。

(4) Edg は遷移を表し、$Loc\times Z(C)\times\Sigma\times 2^C\times Loc$ の部分集合。要素 (l_1, g, a, r, l_2) を、$l_1 \xrightarrow{g,a,r} l_2$ のように表記する。ここで、g はクロック制約からなるガード条件、a は入力記号、r はリセットするクロック変数の集合を示す。また、$Edge$ のすべての要素に対して、ある遷移元からの遷移が、真となるガード条件と入力記号を決めた時、その遷移先が唯一に決定できる場合、決定性 PCA (deterministic PCA) と呼ぶ。

(5) $Flow$ は、各ロケーションにおける重みの変化を表すダイナミックス。$\mathcal{R}_+$ を非負の実数、$\mathcal{R}_+^D = D\to\mathcal{R}_+$ として、$Flow\ :\ Loc\to(\mathcal{R}_+^D\to\mathcal{R}_+^D)$ である。PCA は各ロケーション (ℓ) での単位時間あたりの電力消費が一定 (M^ℓ) なので、重みのバリュエーションを w、消費電力量を表す変数を p とすると、$Flow(\ell)(w)(p) = dp/dt = M^\ell$ となる。

(6) Inv は、各ロケーションに付されたクロック制約。$Inv\ :\ Loc\to Z(C)$

(7) Lab は、各ロケーションで成り立つアトミックな命題の集合。$Lab\ :\ Loc\to 2^{Prop}$

2.1.2 動作意味

PCA $\mathcal{A}$ の動作意味をラベル付き遷移システム（Labeled Transition System、LTS と略記) で与える。LTS $\langle\ S, T\ \rangle$ を以下のように構成する。

状態空間 S は、ロケーション l、クロック変数のバリュエーション v と重み付き変数のバリュエーション w の組を状態とする集合である。

$\{(l, v, w)\in Loc\times\mathcal{R}_+^C\times\mathcal{R}_+^D\ \mid v\models Inv(l)\ \}$

ロケーションに与えられたクロック制約は常に満たされる。

状態遷移 T は、$T = \{\ \xrightarrow{d,e}\ \}\cup\{\ \xrightarrow{d,\epsilon}\ \}$ であり、通常遷移と滞留遷移からなる。詳細を以下に示す。

- 離散遷移 $(l_1, v_1, w)\xrightarrow{e}(l_2, v_2, w)$

  $\exists(l_1 \xrightarrow{g,a,r} l_2)\in Edg, v_1\models g, v_2 = v_1[r]$

- 時間遷移 $(l, v, w_1)\xrightarrow{d}(l, v+d, w_2)$

  $d\in\mathcal{R}_+, w_1 = f(0), w_2 = f(d)$
  $\forall t\in\,]0, d[\ \mid\ v+t\models Inv(l)\wedge df/dt = Flow(l)$

  遅延 d だけの時間が経過し $(v+d)$、同時に、重み変数も更新される。上記では、一般形として、開区間 $]0, d[$ で微分可能な時間の関数 f を用いて表現した。個別の重み変数 p について考えると、ロケーション ℓ で $dp/dt = M^\ell$ であり、$w_2(p) = M^\ell\times d + w_1(p)$ となる。

- 空遷移 $(l_1, v_1, w)\xrightarrow{\epsilon}(l_2, v_2, w)$

  $\exists(l_1 \xrightarrow{g,\epsilon,r} l_2)\in Edg, v_1\models g\wedge Flow(l)\ =\ \emptyset$

  入力アルファベットが ϵ で重みが不変となるような離散遷移。

- 通常遷移 $(l_1, v_1, w_1)\xrightarrow{d,e}(l_2, v_2, w_2)$
  時間遷移 $(l_1, v_1, w_1)\xrightarrow{d}(l_1, v, w_2)$ の後に、離散遷移 $(l_1, v, w_2)\xrightarrow{e}(l_2, v_2, w_2)$ が生じる。

- 滞留遷移 $(l, v_1, w_1)\xrightarrow{d,\epsilon}(l, v_2, w_2)$
  時間遷移 $(l, v_1, w_1)\xrightarrow{d}(l, v, w_2)$ の後に、空遷移 $(l, v, w_2)\xrightarrow{\epsilon}(l, v_2, w_2)$ が生じる。

この時、遷移列を、通常遷移 $\xrightarrow{d,e}$ からなる有限列であるか、あるいはそのような有限列の後に、0 個以上の滞留遷移 $\xrightarrow{d,\epsilon}$ からなる無限列であると定義する。なお、滞留遷移が定義されているロケーションを受理状態と考える。

さらに、PCA $\mathcal{A}$ の受理列 $\mathcal{L}(\mathcal{A})$ を時間付き状態列 ρ の全体とする。時点 $\tau^j\in\mathcal{R}_+$ の列が時間の経過を表し、$\tau^0 = 0$、時間遷移 $\xrightarrow{d}$ に対して $\tau^{i+1} = \tau^i + d$ とする。状態 $\sigma^j\in S$ に対して、$\rho^j = (\sigma^j, \tau^j)$ として、$\rho = \rho^0\rho^1\cdots$ である。このように構成した ρ に対して、$\rho\in\mathcal{L}(\mathcal{A})$ である。

2.2 検査性質の表現

2.2.1 検査性質

検査したい性質の例として、「状態 p から状態 q までの時間区間 10 以内で消費した電力が与えられた最大値 50 を超えないこと」等がある。「変数 x が自由出現する論理式 ϕ^x が充足する状態での変数値 m を x に束縛する」ことを表す凍結限量子 (freeze quantifier) $\mathcal{F}^m\mathrm{x}\ .\ \phi^x$ を用いて書き表せる。ただし、m はクロック変数かコスト計算に用いる電力消費を表す変数とする。

$\Box\mathcal{F}^\tau x.\mathcal{F}^m u.(p\Rightarrow$
$\quad\Diamond\mathcal{F}^\tau y.\mathcal{F}^m v.\ (q\wedge(y\leq x+10)\wedge(v\leq u+50)))$

2.2.2 fWLTL の構文定義

凍結限量子を持つ重み線型時相論理 (Weighted Linear Temporal Logic with freeze quantifiers、 fWLTL と略記) の構文を示す。

π	$:=\ c$	定数 $\in\mathcal{N}$
	$\mid x\ +\ c$	加算演算
ϕ	$:=\ p$	アトミックな命題 $\in Prop$
	$\mid \pi_1\leq\pi_2$	不等式制約

$\mid \neg\phi$	論理否定
$\mid \phi_1 \wedge \phi_2$	論理積
$\mid \phi_1$ U ϕ_2	Until 演算子
$\mid \mathcal{I}^m x\,.\,\phi^x$	凍結限量子
$\mid \mathcal{I}^\tau x\,.\,\phi^x$	凍結限量子（時点）

2.2.3 fWLTL の意味定義

時点表示の考え方で、fWLTL 式の意味定義 (Pointwise Semantics) を与える。いくつかの記号を導入する。

- 時間付き状態列 $\rho = \rho^0\rho^1\cdots$。ただし、状態 $\sigma^j = (l_j, v_j, w_j)$ と時点 τ^j に対して、$\rho^j{=}(\sigma^j, \tau^j){=}(l_j, v_j, w_j, \tau^j)$ は時間付き状態。
- 変数の可算無限集合 Var
- 凍結限量子で束縛された変数の解釈
  $\Gamma : Var \to \mathcal{R}_+$
- $\Gamma[x := e]$ は変数 x に実数値 e を割当て、x 以外は不変

組み $\langle \rho, \Gamma \rangle$ が、 fWLTL 式 ϕ を満たすことは、時点表示によって、以下のように帰納的に定義された充足関係 $\models$ に対して、$\langle \rho, \Gamma \rangle \models \phi$ となることである。

$\langle \rho^j, \Gamma \rangle \models c$ iff $c \in \mathcal{R}_+$
$\langle \rho^j, \Gamma \rangle \models x + c$ iff $\Gamma(x) + c \in \mathcal{R}_+$
$\langle \rho^j, \Gamma \rangle \models p$ iff $p \in Lab(l_j)$
$\langle \rho^j, \Gamma \rangle \models \pi_1 \le \pi_2$ iff $\Gamma(\pi_1){\le}\Gamma(\pi_2)$
$\langle \rho^j, \Gamma \rangle \models \neg\phi$ iff $\langle \rho^j, \Gamma \rangle \not\models \phi$
$\langle \rho^j, \Gamma \rangle \models \phi_1 \wedge \phi_2$
　iff $\langle \rho^j, \Gamma \rangle \models \phi_1$, and $\langle \rho^j, \Gamma \rangle \models \phi_2$
$\langle \rho^j, \Gamma \rangle \models \phi_1$ U ϕ_2
　iff $\langle \rho^k, \Gamma \rangle \models \phi_2$ for some $k \ge j$,
　and $\langle \rho^i, \Gamma \rangle \models \phi_1$ for all i $(j{\le}i{<}k)$
$\langle \rho^j, \Gamma \rangle \models \mathcal{I}^m x\,.\,\phi^x$
　iff $\langle \rho^j, \Gamma[x := (v^j \cup w^j)(m)] \rangle \models \phi^x$
$\langle \rho^j, \Gamma \rangle \models \mathcal{I}^\tau x\,.\,\phi^x$
　iff $\langle \rho^j, \Gamma[x := \tau^j] \rangle \models \phi^x$

アトミックな命題 p の否定 $\neg p$ は以下のようになる。

$\langle \rho^j, \Gamma \rangle \models \neg p$ iff $p \in Prop \backslash Lab(l_j)$

これを、$\langle \rho^j, \Gamma \rangle \not\models p$ と書く。

2.3 モデル検査

電力消費オートマトン $\mathcal{A}$ が生成する遷移列の全体を $\mathcal{L}(\mathcal{A})$ とする時、$\mathcal{A}$ に対する閉じた fWLTL 式 ϕ のモデル検査 $\mathcal{A}, \Gamma \models \phi$ は、全ての時間付き状態列 $\rho{\in}\mathcal{L}(\mathcal{A})$ と空の Γ に対して、$\langle \rho^0, \Gamma \rangle \models \phi$ が成り立つかを調べることである。一方、PCA に対する fWLTL のモデル検査は決定不能 [11] なことから、何らかの近似法を導入しなければならない。第 3 節で詳述する。

3. Realtime Maude への変換

3.1 概要

モデル検査の近似手法として、Real-time Maude[12] の時間有界探索とサンプリング抽象の方法を採用する。有限長の時間付き状態列 $(\rho^0 \ldots \rho^N)$ を対象として、与えられた $\mathcal{A}$ が状態変化を引き起こすインデックス $j(0{\le}j{\le}N)$ をカバーするように論理式を検査する点を選ぶ有界モデル検査法である。

本稿では、PCA ならびに fWLTL 式を Realtime Maude に変換する方法を考察する。PCA を重み付き時間オートマトンにより表し、操作的な意味を与えるラベル付き遷移システムを、Realtime Maude に変換する。Realtime Maude が提供する LTL モデル検査アルゴリズムを利用するには、アトミックな命題を定義しなければならない（付録参照)。さらに、凍結限量子を持たないので、これを、どのように表現するかが技術的な問題となる。fWLTL のサブセットを導入し、凍結限量子と等価な情報を命題で表現する方法を考案することで、この問題を解決する。

3.2 fWLTL のサブセット

3.2.1 凍結限量子の取り扱い

fWLTL の表現力が大きいことは、凍結限量子が任意の fWLTL 式 ϕ^x をとることにある。Realtime Maude での取り扱いが難しいことから、ϕ^x の形を制限する。以下、$\mathcal{I}^\tau x.\phi^x$ を対象として考察する。$\mathcal{I}^m x.\phi^x$ の場合も同様に取り扱えば良い。

3.2.1.1 意味定義と補足

- 状態命題論理積 (State-sensitive Conjuncts)
  時点付き状態 ρ^j に対して、当該状態に付与されたアトミックな命題 $p \in Prop$) から作られるリテラル g を少なくとも 1 つ含む論理積。ϕ_S と書く。この時、g を状態ガード命題 (State Guard Proposition) と呼ぶ。
- 凍結限量された論理式 $\mathcal{I}x.\phi_S$ の意味
  論理式 ϕ_S が持つ状態ガード命題を g として、$\phi_S = g \wedge \phi^x$ とおくと以下になる。
  $\langle \rho^j, \Gamma[x := \tau^j] \rangle \models \mathcal{I}^\tau x\,.\,\phi_S$
  　iff $\langle \rho^j, \Gamma \rangle \models g$
  　and $\langle \rho^j, \Gamma[x := \tau^j] \rangle \models \phi^x$
- 論理和
  積項 ϕ_{S1}、ϕ_{S2} の論理和は、分配則を用いる。
  $\mathcal{I}^\tau_x.\ (\phi_{S1} \vee \phi_{S2}) = \mathcal{I}^\tau_x.\ \phi_{S1} \vee \mathcal{I}^\tau_x.\ \phi_{S2}$
- 論理否定
  可能な時点の集合 $\Theta = \{ \tau^j \}$ を考えると、Θ は可算無限集合である。ここで、概念的に、凍結限量子を Θ の全ての要素についての論理和で展開する。
  $\langle \rho^j, \Gamma \rangle \models \mathcal{I}^\tau x\,.\,\phi_S$

$\Leftrightarrow\ \langle\ \rho^j,\ \Gamma\ \rangle \models (\bigvee_{\tau\in\Theta}\ (x=\tau))\ \wedge \phi_S$
$\Leftrightarrow\ \ \langle\ \rho^j,\ \Gamma[x:=\tau^j]\ \rangle \models\ \ (x=\tau^j)\wedge\phi_S$

次に、$\neg(\mathcal{I}^\tau x.\ \phi^x)$ を考える。上記と同様に、論理和で展開する。

$\langle\ \rho^j,\ \Gamma\ \rangle \models\ \neg(\mathcal{I}^\tau x\ .\ \phi_S)$
$\Leftrightarrow\ \langle\ \rho^j,\ \Gamma\ \rangle \models \neg((\bigvee_{\tau\in\Theta}\ (x=\tau))\ \wedge\phi_S)$
$\Leftrightarrow\ \langle\ \rho^j,\ \Gamma\ \rangle \models \neg(\bigvee_{\tau\in\Theta}\ (x=\tau))\ \vee\neg\phi_S$
$\Leftrightarrow\ \langle\ \rho^j,\ \Gamma\ \rangle \models (\bigwedge_{\tau\in\Theta}\ (x\neq\tau))\ \vee\neg\phi_S$

ところが、Θ は可算無限集合なので、$x=\tau$ が存在するはずである。したがって、

$\Leftrightarrow\ \langle\ \rho^j,\ \Gamma\ \rangle \models\ \neg\phi_S$
$\Leftrightarrow\ \langle\ \rho^j,\ \Gamma\ \rangle \models\ \mathcal{I}^\tau x.\ \neg\phi_S$

- 凍結命題 F^x

  説明のため、F^x を $\langle\ \rho^j,\ \Gamma[x:=\tau^j]\ \rangle \models F^x$ であるようなアトミック命題として導入する。直感的には、変数 $x=\tau^j$ の等号関係を表す。

  $\langle\ \rho^j,\ \Gamma[x:=\tau^j]\ \rangle \models\ F^x\wedge\phi^x$

  凍結命題は、凍結限量子が付された場所を示すマーカで、変数束縛環境の更新箇所に対応する。

3.2.2 制約式の取り扱い

3.2.2.1 制約命題

fWLTL のモデル検査を複雑にしている 2 つめの理由に、時点情報や重みに関する制約条件の表現力がある。定量的な制約関係 $C^{x,y}$ を構成する変数 x,y は次のような性質がある。

- 凍結限量子で束縛されており、対応する状態ガード命題を満たす時点の値を保持する。
- 凍結限量子は入れ子構造になり、2 つの変数 x と y の間には、出現順にしたがった順序関係が生じる。
- 時点ならびに電力消費を表す重みは単調増加する。

本稿では、$C^{x,y}$ を変数 x を決めた時の変数 y に対する閉じた区間制約と考える。つまり、$C^{x,y}$=$C^x(y)$=$(y\le c(x))$ とする。今、$\mathcal{I}x.(\cdots\Diamond\mathcal{I}y.(\cdots\wedge\ C^{x,y}))$ とする。凍結限量子 $\mathcal{I}x.$ に対応する状態ガード命題が p である時、$\langle\ \rho^j,\ \Gamma\ \rangle \models p$ を満たす時点の集まりを $\Theta_p=\{\tau^i\}$ とする。同様に、$\mathcal{I}y.$ の状態ガード命題 q に対して、$\Theta_q=\{\tau^j\}$ とする。制約条件式 $C^{x,y}$ を評価する x,y は、これらの集合の要素なので、関連する部分式をに対して概念的に論理和に置き換える。ただし、x,y は前後関係に対して指定された時相的な性質を満たす。このことを、$\langle\_,\_\rangle_{LTL}$ と記す。

$$(\bigvee_{\tau^i\in\Theta_p,\tau^j\in\Theta_q}\ \langle x=\tau^i,\ y=\tau^j\rangle_{LTL})\wedge C^{x,y}$$

となる。Θ_p と Θ_q が有限集合であれば良い。サンプリング抽象を用いる方法では、有限に限定することが可能である。

3.2.2.2 式変形の例

fWLTL 式の具体例を用いて、論理式の変形ステップを示す。

$\Box\mathcal{I}^\tau x.\ (\ (p_1\wedge p_2)\Rightarrow\ \Diamond\mathcal{I}^\tau y.\ (q\wedge C^{x,y})\ )$

次の関係式を用いる。

$$\phi_{S1}\Rightarrow\phi_2=\ \neg\phi_{S1}\vee\phi_2\ \ =\neg\phi_{S1}\vee(\phi_{S1}\wedge\phi_2)$$

以下、簡単のため、最外の □ を省略する。

$\mathcal{I}^\tau x.\ (\neg(p_1\wedge p_2)\vee\ \ (p_1\wedge p_2\wedge\Diamond(\mathcal{I}^\tau y.\ (q\wedge C^{x,y}))))$
$\mathcal{I}^\tau x.\ ((\neg p_1\vee\neg p_2)\vee\ \ (p_1\wedge p_2\wedge\Diamond(\mathcal{I}^\tau y.\ (q\wedge C^{x,y}))))$
$\mathcal{I}^\tau x.\neg p_1\vee\mathcal{I}^\tau x.\neg p_2\vee\mathcal{I}^\tau x.\ (p_1\wedge p_2\wedge\Diamond(\mathcal{I}^\tau y.(q\wedge C^{x,y})))$
$(F_1^x\wedge\neg p_1)\vee(F_2^x\wedge\neg p_2)\vee((F_3^x\wedge p_1\wedge p_2)\wedge\Diamond(F^y\wedge q\wedge C^{x,y}))$

F_1^x と F_2^x に対応する凍結限量子は制約式 $C^{x,y}$ をスコープに含まないので、凍結命題がマークした箇所は結果に影響しない。以下と等価になる。

$$\Box(\neg p_1\vee\neg p_2\vee((F_3^x\wedge p_1\wedge p_2)\wedge\Diamond(F^y\wedge q\wedge C^{x,y})))$$

F_3^x の状態ガード命題は $p_1\wedge p_2$ であり、F^y に対しては q になる。

3.2.3 検査対象論理式の構文

Realtime Maude でモデル検査を行う時に、対象とする論理式の構文を示す。

ϕ	$:=\ p$	アトミックな命題 $p\in Prop$
	$\mid\neg p$	否定命題 $p\in Prop$
	$\mid F^x$	凍結命題 $\in Prop^x$
	$\mid C^{x,y}$	制約命題 $\in Prop^{x,y}$
	$\mid\neg\ \phi$	論理否定
	$\mid\phi_1\wedge\phi_2$	論理積
	$\mid\phi_1\vee\phi_2$	論理和
	$\mid\Box\phi$	Always 演算子
	$\mid\Diamond\phi$	Eventually 演算子

アトミックな命題、凍結命題、制約命題、を、Realtime Maude の枠組みで表現すればよい。ただし、凍結命題は、常に状態ガード命題との論理積で出現する。そこで、検査性質を表す LTL 論理式から凍結命題を省略する。

3.3 変換の概略

3.3.1 ラベル付き遷移システムの変換

3.3.1.1 状態

時間付き状態 (l_j,v_j,w_j,τ^j) はソート記号と生成子を定義する。第 3.4.2 節を参照。

3.3.1.2 遷移

離散遷移 状態遷移 $(l_1,v_1,w)\xrightarrow{e}(l_2,v_2,w)$ を表す瞬時ルールの書き換え規則で表すことができる。遷移元の不変量を書き換え規則の条件とする。

r : $E(A)$ (L1,V,W,T)
$\Longrightarrow$ (L2,reset(L1,V),W,T) **if** inv(L1,V)

$E(A)$ は引数 A を持つイベント（アルファベット Σ）、reset は PCA の遷移に付されたクロックをリセットする処理、inv は遷移元ロケーションに付されたクロック制約条件を評価する処理、を各々表す。ところが、線型不等式を表すには、関数 mte を併用する場合がある。$d_1 \leq R \leq d_2$ は、瞬時ルールと mte を組み合わせる。

r : $E(A)$ (L1,V,W,T) $\Longrightarrow$ (L2,V,W,T) **if** $d_1 \leq T$
d : `mte`((L,V,W,T)) = d_2 `monus` T .

下限 (d_1) は瞬時ルールの条件で判定する。上限 (d_2) は mte で計算し、与えられた時区間でサンプリングする。

時間遷移 状態遷移 $(l, v, w_1) \xrightarrow{d} (l, v + d, w_2)$ を、時間ルールで書き表すことで、時間の経過を表現する。

l : { S }
$\Longrightarrow$ { `delta`(S, D) } **in time** D **if** $D \leq$ `mte`(S)

時間経過に関する振る舞いは、関数 `delta` で定義する。

d : `delta`((L,V,W,T),D)
= (L,V+D,update(L,W,D),T+D)

関数 update は重み値を W+M^{ℓ}×D に更新する。

通常遷移 Realtime Maude は瞬時ルールと時間ルールとをインターリーブするので、通常遷移を実現できる。

3.3.2 検査方法の変換

3.3.2.1 命題の変換

アトミックな命題 $p \in Prop$ はロケーションで定義されている。以下で、関数 labels は Lab を表す。

p : { (L,V,W,T) S } $\models$ p = true **if** p $\in$ labels(L)
p : { (L,V,W,T) S } $\models$ np = true **if** p $\notin$ labels(L)

制約命題は変数環境 G が保持する値を参照する。

p : { G S } $\models$ $C^{X \leq Y}$ = (lookup(G,x) $\leq$ lookup(G,y))

凍結命題は状態ガード命題 p と同一視できるので具体的な表現が不要となる。

3.3.2.2 束縛変数の環境

凍結命題が真になる時、限量変数の環境 Γ を更新する。更新遷移を、$\xrightarrow{\Gamma}$ と表記する。簡単には、状態ガード命題が真になる条件で、変数環境 G を更新すれば良い。

r : (L,V,W,T) G
$\Longrightarrow$ (L,V,W,T) update(G,x,T) **if** p in labels(L)

これは、瞬時ルールであり、Realtime Maude では、離散遷移と共に発火する。つまり、$\xrightarrow{e1}$; $\xrightarrow{\Gamma}$; $\xrightarrow{d1}$ の順に遷移発火する。一方、G は、離散遷移によって到達した安定状態に対して経過時間を計算した後の値を保持しなければならない。つまり、$\xrightarrow{e1}$; $\xrightarrow{d1}$; $\xrightarrow{\Gamma}$ である。この後、次の $\xrightarrow{e2}$ が続くことから、$\xrightarrow{\Gamma,e2}$ のような瞬時ルールとして表現する。

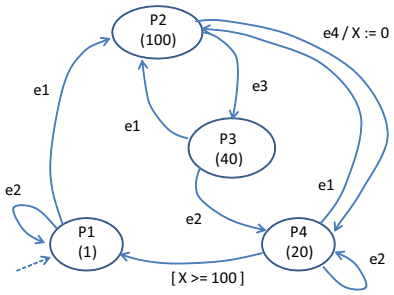

図 2 検査対象例

3.3.2.3 検査 LTL 式

凍結命題 F^x はマーカーであり、対応する状態ガード命題 g と等価 ($F^x \Leftrightarrow g$) なことから検査 LTL 式から省くことができる。第 3.2.2.2 節の例を参照のこと。

3.4 Realtime Maude 記述の具体例

3.4.1 例題

図 2 は図 1 を簡単化した PCA のダイアグラム表現例である。これに対して、論理式

$$\Diamond \mathcal{I}^{\tau} x. \mathcal{I}^{m} u. (\mathrm{aP2} \wedge \Diamond \mathcal{I}^{\tau} y. \mathcal{I}^{m} v. (\mathrm{expire} \wedge \mathrm{cXY} \wedge \mathrm{cUV})))$$

を検査する場合について、Realtime Maude の記述を示す。ただし、

$$\mathrm{cXY} \equiv y \leq x + 200, \quad \mathrm{cUV} \equiv v \leq u + 2000$$

とした。時相的な性質と同時に制約条件を満たす時点ならびに重みを見つければ良い。

3.4.2 補助定義

3.4.2.1 変数環境

凍結限量子の値を管理する変数環境 Γ を変数ごとに値を保持する項で表す。たとえば、$\mathcal{I}^{\tau} x$ に対する Gamma 項 gX は次のようになる。

sort Gamma . **subsort** Gamma < `System` .
op gX : Time → Gamma [ctor] .

この時、lookup(G,x) は、書き換え規則の左辺に出現する gX(M) 項から引数 M を抜き出すことである。また、update(G,x,T) は、書き換え規則左辺の gX(M) 項を無視して、右辺に新たな gX(T) 項を生成することになる。

3.4.2.2 ロケーション

ソート Loc はロケーションの有限集合を表す。各状態は Loc の定数項になる。

sort Loc . **ops** P1 P2 P3 P4 : → Loc [ctor] .

3.4.2.3 イベント

アルファベット Σ はソート Event で表す。引数を持た

ないイベントはロケーションと同様に定数項で表現する。なお、Event を System のサブソートとすることで、大域的な状態のコンポーネントとする。

sort Event . **subsort** Event < System .
ops e1 e2 e3 e4 : → Event [ctor] .

3.4.3 PCA

3.4.3.1 共通部分

基本的な Mealy 型オートマトンを定義する。ソート Machine を System のサブソートとして導入する。

sort Machine . **subsort** Machine < System .

System は複数のコンポーネント (Machine、Event、Gamma) を持つことから多重セットの生成子 (_ _) を導入する。関数 delta と mte が多重セットに適用可能とするように一般的な分解処理を定義する。

op _ _ : System System → System [ctor assoc comm] .
eq delta(S1 S2, R) = delta(S1, R) delta(S2, R) .
eq mte(S1 S2) = min(mte(S1), mte(S2)) .

時間的に変化しない Event 項について関数 delta と mte を定義する。Gamma 項についても同様である。

eq delta(E, R) = E . **eq** mte(E) = INF .

時間ルールは Realtime Maude の基本的な形で実現できる。

crl [tick] : { S } ⇒ { delta(S, R) }
in time R **if** R ≤ mte(S) [nonexec] .

3.4.3.2 振る舞いの定義

具体的な Machine 項と書き換え規則によって、与えられた PCA の具体的な機能振る舞いを定義する。図 2 の PCA はロケーション (Loc 項)、タイマー (Time 項)、エネルギー (Rat 項)、時点 (Time 項)、タイマー有効フラッグ（Bool 項）の 5 つのコンポーネントからなる。例として、ロケーション P4 からの状態遷移に関係する瞬時ルールを示した。

op pca : Loc Time Rat Time Bool → Machine [ctor] .
crl [p4a] : pca(P4,V,W,R,B)
⇒ TimeOut pca(P1,V,W,R,B) **if** V ≥ 100 .
rl [p4b] : pca(P4,V,W,R,B) e1 ⇒ pca(P2,V,W,R,B) .

ルール [p4a] はタイマー V のタイムアウト発生時の振る舞いを示す。ここで、タイムアウト発生を示すイベント TimeOut を導入した。

op TimeOut : → Event .

また、下記の mte を定義しておく。また、ルール [p4b] はイベント e1 による離散遷移である。

eq mte(pca(P4,V,W,R,B)) = 100 monus V .

消費エネルギーはロケーション滞留時間に比例するので関数 delta で定義する。ただし、フラッグがオフの時は、タイマーを更新しない。

eq delta(pca(P2,V,W,R,true), T)
= pca(P2,(V plus T), ((100 * T) + W), (R plus T),true) .
eq delta(pca(P2,V,W,R,false), T)
= pca(P2,V, ((100 * T) + W), (R plus T),false) .

3.4.3.3 アトミックな命題の定義

ロケーション P2 で真となるアトミックな命題 aP2 を Prop 項として定義する。

op aP2 : → Prop .
eq { pca(P2,V,W,R,B) S } ⊨ aP2 = true .

タイムアウト状態を表す命題 expire を定義する。

op expire : → Prop .
eq { TimeOut S } ⊨ expire = true .

3.4.3.4 制約命題の定義

アトミックな命題である Prop 項として定義する。変数環境で保持している変数値を参照する。

op cXY : → Prop .
eq { S gX(X) gY(Y) } ⊨ cXY = Y ≤ X+200 .

3.4.3.5 検査性質の定義

先に示した検査式について、凍結限量子を凍結命題で置き換える。

$$\Diamond(F^x \wedge F^u \wedge \text{aP2} \wedge \Diamond(F^y \wedge F^v \wedge \text{expire} \wedge \text{cXY} \wedge \text{cUV})))$$

凍結命題を除去して、Realtime Maude の LTL 論理式に書き直す。

$$\varphi \equiv \Diamond(\text{aP2} \wedge \Diamond(\text{expire} \wedge \text{cXY} \wedge \text{cUV})))$$

なお、凍結命題 F^x と F^u の状態ガード命題は aP2、F^y と F^v に対しては expire である。

3.4.3.6 変数環境の更新

状態ガード命題が aP2 であるような凍結命題 F^x が束縛変数 x に対応する時 ($\mathcal{J}^\tau x$)、Gamma 項 gX を命題 aP2 が成り立つ条件で更新する。図 2 ではロケーション P2 からの遷移が 2 つ定義されている。そのひとつを示す。

rl e3 pca(P2,V,W,R,B) ⇒ pca(P3,V,W,R,B) .

この瞬時ルールは、pca がロケーション P2 で時間経過した後に発火する。つまり、$\xrightarrow{d2}$ である。この瞬時ルールに変数環境更新の機能を追記すれば良い。

crl gX(T1) gU(M1) e3 pca(P2,V,W,R,B)
⇒ gX(R) gU(W) pca(P3,V,W,R,B) **if** (T1=0) and (M1=0).
crl gX(T) gU(M) e3 pca(P2,V,W,R,B)
⇒ gX(T) gU(M) pca(P3,V,W,R,B) **if** (T≠0) and (M≠0).

3.4.3.7 検査の外部環境

モデル検査は「閉じたシステム」に対して行う。そこで、検査対象 PCA(図 2) に加えて、e1 等のイベントを発生する「外部環境」が必要となる。検査対象は WiFi-STA の簡易記述であることから、アクセスポイント (WiFi Access Point) を時間オートマトン env として表現した。

3.4.3.8 モデル検査の実行

適切な初期 GlobalSystem 項 ({ S } の形をした項) を与えて、モデル検査コマンドを実行する。以下の例では、検査範囲の時間を 1000 以内に限定した。

$$\mathbf{mc}\ init \models^t \varphi\ \mathbf{in\ time} \leq 1000\ .$$

初期状態 $init$ は、以下のような項からなる。

{ env(A1,3,0) pca(P1,0,0,0,false) gX(0) gU(0) gY(0) gV(0) }

4. 関連研究

PCA は線型ハイブリッドオートマトン (LHA)[3] であり、ストップウォッチ・クロック変数を持つ n レート時間システム (n-RTS) で表現できる [9]。電力消費は観測変数で表現できるので、これを重みとすれば、重み付き時間オートマトン (WTA)[4] あるいはプライス付き時間オートマトン (LPTA)[5] で表現できる。ただし、PCA では重みは状態にのみ定義されており、遷移エッジには重みを持たない。性質検証の基本となる到達性解析については、nRTS は決定不能である。WTA と LPTA は等価であり、時間オートマトン (TA) の記号的な方法に帰着できるので決定可能である。

TA に対して、有界な時区間での到達性解析 (duration-bounded reachability)[1] がある。文献 [4][5] は、この問題を拡張して、重み（プライス）を尺度とする最適経路や最小経路を含む到達性解析の方法を示した。fWLTL は、有界な時区間での到達性は表現することが可能であるが、最適経路等の評価関数が必要な問題を扱うことはできない。多様な性質を柔軟に表現することを重視し、時相論理を採用した。

凍結限量子は LPTL[2] が提案した。LPTL は、TA が生成する時間付き文字列に対して定義されており、束縛変数は時間点 (now) を「凍結」する。一方、fWLTL は、重みによって表現した電力消費値も参照できるように拡張した。LPTL の凍結限量子 $x.\phi^x$ は、fWLTL では、$\mathcal{J}^\tau x.\phi^x$ と表現する。LPTL はサブセットに Metric Temporal Logic (MTL) を含む。ところが、MTL の TA に対するモデル検査は決定不能である。本稿では、fWLTL の PCA に対するモデル検査法として、Realtime Maude の明示的な時点表現と有界時間探索ならびにサンプリング抽象の方法を採用した。特に、Realtime Maude への変換に際して、fWLTL をサブセットに制限した。

5. おわりに

本稿では、先に提案したスマートフォン・アプリケーションの電力消費に関するモデルベース解析 [9] で中心的な役割を果たすモデル検査の方法を考察した。文献 [11] で電力消費振る舞いの形式表現 PCA と検査性質の表現 fWLTL を提案したが、本稿では、Realtime Maude を用いるモデル検査の方法を具体的に検討した。PCA に対する fWLTL のモデル検査は、一般的には、決定不能であることから、近似手法を導入せざるを得ない。今後、検査する制約条件の範囲を広げることを検討する。また、具体例を対象とした実験を行うことで、近似手法が解析結果に及ぼす影響を具体的に考えていく。

参考文献

[1] R. Alur, C. Courcoubetis, and T.A. Henzinger. Computing Accumulated Delays in Real-Time System, In *Proc. CAV 1993*, pp.181-193, 1993.
[2] R. Alur and T.A. Henzinger. A Really Temporal Logic, *J. Accoc. Comp. Machin.*, Vol.41, No. 1, pp.181-204, 1994.
[3] R. Alur, C. Courcoubetis, N. Halbwachs, T.A. Henzinger, P.-H. Ho, X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine. The Algorithmic Analysis of Hybrid Systems, *Theor, Comp. Sci.*, No.138, pp.3-24, 1995.
[4] R. Alur, S. La Torre, and G.J. Pappas. Optimal Paths in Weighted Timed Automata, In *Proc. HSCC 2001*, pp.49-62, 2001.
[5] G. Behrmann, A. Fehnker, T. Hune, K. Larsen, P. Pettersson, J. Romjin, and F. Vaandrager. Minimum-Cost Reachability for Priced Timed Automata, In *Proc. HSCC 2001*, pp.147-161, 2001.
[6] P. Bouyer, U. Fahrenberg, K.G. Larsen, and N. Markey. Timed Automata with Observers under Energy Constraints, *Proc. HSCC 2010*, 2010.
[7] M. Clavel, F. Duran, S. Eker, P. Lincoln, N. Marti-Oliet, J. Meseguer, and C. Talcott. *All About Maude - A High-Performance Logical Framework*, Springer 2007.
[8] J. Meseguer. Conditional Rewriting Logic as a Unified Model of Concurrency, *ENTCS*, no.96, pp.73-155, 1992.
[9] 中島震. スマートフォン・アプリ電力消費のモデルベース解析, 情報処理学会組込みシステム研究会, 2013.
[10] S. Nakajima. Everlasting Challenges with the OBJ Language Family, In *Proc. SAS 2014*, pp.478-493, 2014.
[11] 中島震. スマホ・アプリの電力消費振る舞いと検査性質の表現, 日本ソフトウェア科学会第 31 回大会, 2014.
[12] P.C. Olveczky and J. Meseguer. Semantics and Pragmatics of Real-Time Maude, *Higher-Order and Symbolic Computation*, vol.20, no.1-2, pp.161-196, 2007.
[13] P.C. Olveczky and J. Meseguer. Abstraction and Completeness for Real-Time Maude, *ENTCS*, no.176-4, pp.5-27, 2007.
[14] J. Ouaknine and J. Worrell. Some Recent Results in Metric Temporal Logic, In *Proc. FORMATS 2008*, pp.1-13, 2008.

付　　録

A.1 Realtime Maude の概要

Realtime Maude (RT-Maude) [12] は Maude [7] を拡張して、実時間システムやハイブリッドシステムを表現可能にした形式仕様言語である。理論的には、Maude の書き換え論理 [8] を基礎とする。クロック変数に関する制約条件に記号表現を用いない。時点を明示的に表す。

A.1.1 書き換え規則

Realtime Maude は、瞬時ルールと時間ルールという 2 種類の書き換え規則を持つ。瞬時ルールは、Maude の書き換え規則そのものであって、離散的な遷移を表現する。時間の遅延がなく、瞬間的に並行実行される。今、ルールに付与するラベルを r とする時、引数 A を持つ項 $T_1(A_1)$ を対象とする条件付き瞬時ルールは、次のようになる。

$$r : T_1(A_1) \Longrightarrow T_2(A_2) \textbf{ if } C \ .$$

条件 C が成り立つ時、左辺の項 $(T_1(A_1))$ が、右辺の項 $(T_2(A_2))$ に、書き換えられることを表す。

時間ルールは、時間経過を伴う振る舞いを表現する。一般に、実時間システムやハイブリッドシステムでは、瞬時ルールと時間ルールが混在する。この時、瞬時ルールと時間ルールが同時に発火可能となる場合があることから、書き換え順序に関する何らかの戦略が必要になる。Realtime Maude は、まず、ある時点で発火可能な全ての瞬時ルールを適用して、システム全体を「安定」な状態に導く。その後、発火可能な時間ルールを適用して時間を進める。この時、瞬時ルールの発火が終了することは仕様作成者の責任である。

時間は対象システムと独立にかつ均一に進行する。時間ルールは、システム全体を書き換えの対象とする。システム全体を表す項 T_1 は、ある時点でのスナップショットを表す。具体的には、Realtime Maude で定義されたソート `System` 項である。時間ルールは、$\{\ T_1\ \}$ の形をした `GlobalSystem` 項に対して定義する。

$$\{\ \_\ \} : \texttt{System} \rightarrow \texttt{GlobalSystem}$$
$$\mathrm{l} : \{\ T_1\ \} \Longrightarrow \{\ T_2\ \} \textbf{ in time } \tau_l \textbf{ if } \ C$$

左辺項 T_1 を右辺項 T_2 に書き換える際に、時間 τ_l が経過することを表す。ここで、式 C は経過時間 τ_l に対する条件を持つことができる。この時、ルール発火に伴う経過時間は条件 C を満たせば良く、そのような τ_l が非決定的に選択される。一方、時間は一般には連続であり、値を決めることが難しい。時点を明示的に表現する Realtime Maude では、何らかの近似法が必要になる。

Realtime Maude は、このような時間非決定的システムに対して、サンプリング抽象の方法を導入する。特に、最大経過時間 (Maximum Time Elapse, mte) の考え方を採用した。項 T_1 に対して、2 つの関数 δ と mte を定義する。関数 δ は時間 τ_l が経過した時点での新しい項 T_2 を返す。関数 mte は項が変更されない最大の経過時間を表す。全く変化しない場合、最大経過時間は無限大となる。

$$\delta : \texttt{System Time} \rightarrow \texttt{System}$$
$$mte : \texttt{System} \rightarrow \texttt{TimeInf}$$

時間ルールは $\delta(T,\tau)$ で項を計算し、条件式から $mte(T)$ を参照する形式になる。

$$\mathrm{l} : \{\ T\ \} \Longrightarrow \{\ \delta(T,\tau_l)\ \} \textbf{ in time } \tau_l \textbf{ if } \ \tau_l \leq mte(T)$$

ここで、$mte(T)$ は経過可能な時間の上限に相当するので、その時間内で時間非決定遷移が起きることを示す。つまり、$mte(T)$ の時間間隔で、少なくとも 1 度、項 T の計算を行うことを表している。τ_l の選び方によっては、項 T に変化が生じない $(\delta(T,\tau_l) = T)$ 場合もある。

A.1.2 モデル検査

Realtime Maude は Maude と同様に、LTL ロジック・モデル検査機構を内蔵している。一般には時間は連続かつ単調増加なので、明示的な時点を採用する場合は、探索空間を限定する工夫が必要となる。Realtime Maude は、先に述べたサンプリング抽象と時間有界モデル検査を組み合わせる。検査性質を線型時相論理 (Linear Temporal Logic, LTL) の論理式 φ で表現する時、以下のコマンドで、モデル検査エンジンを実行する。

$$\textbf{mc } initState \models^t \varphi \textbf{ in time} \leq B$$

探索する時間の範囲を B に制限することを示す。LTL 式 φ が参照するアトミックな命題 p は、別途、定義しておく。Maude 組込みのオペレータ $\models$ を使う。

$$\_ \models \_ : \texttt{System Prop} \rightarrow \texttt{Bool}$$

ここで、`Prop` はアトミック命題を表す組込みのソート記号である。

$$\mathrm{p} : \rightarrow \texttt{Prop}$$
$$\mathrm{a} : T \models \mathrm{p} \ = \ \mathrm{true} \textbf{ if } C \ .$$

Realtime Maude は、状態命題と時点命題を定義する枠組みを提供する。以下、本稿で用いた状態命題を説明する。サブソート関係 `GlobalState<ClockedSystem<State` を導入した。状態命題は `GlobalSystem` 項に対して定義する。

$$\_ \models \_ : \texttt{GlobalSystem Prop} \rightarrow \texttt{Bool}$$
$$\mathrm{a} : \{\ T\ \} \models \mathrm{p} \ = \ \mathrm{true} \textbf{ if } C \ .$$