アンドロイド・アプリの電力消費振舞い：形式モデルと有界解析

中島 震

電力バグを開発上流工程で除去するモデルベース解析の方法を論じる．電力消費振舞いを重み付き時間オートマトンに基づく形式モデル PCA で表す．消費電力解析は有界区間でのコスト制約問題と考えることができ，さらに，ロジック・モデル検査の問題と定式化することができる．本稿では，SAT/SMT ベースのスコープ有界検証を PCA に適用する方法ならびに，この検証作業によって不具合があるとわかった PCA を対象に欠陥箇所を自動特定する方法を提案する．

1 はじめに

Cyber-Physical System (CPS) （[13] p.5）という言葉が登場して久しい．その間に普及したスマートフォンやタブレットといった携帯端末は日常の生活スタイルを変えた．組込み装置はネットワークによってつながり「スマート」なサービスを提供する基盤となっている．携帯端末や組込み装置の形や機能は千差万別な一方，電力消費が大きな問題となるという共通点がみられる．たとえば，スマートフォンは自身の電池で作動する．電池容量は限定されており，充電しない限り電池は減る一方である．また，意図通りの機能を示すアプリケーション・プログラムが予想外の電池消耗を引き起こすという「電力バグ (e-bugs)」[25] が頻発していることも事実であろう．

モデルベース解析による電力バグ検知の方法がある (例えば [17])．アプリケーション・プログラムからハードウェアまでを含む全体の電力消費振舞いを統一的に表現し解析することで，電力バグを検知する．プログラムが示す離散的な振舞いと同時に，実行時間や消費電力量といった連続値を扱う．CPS の 3 つの特徴 [30] のひとつである「離散と連続の共存」，ハイブリッド性を持つ．実際，線型ハイブリッド・オートマトン [6] のサブクラスを採用する研究がみられる [9] [17] [18] [19].

電力バグを検知する方法は，消費電力値が予め決めた上限を超えないことを確認すれば良いだろう．ところが，消費電力は時間に対して単調増加するので，計測時間を長くとると，いつの時点かでは必ず上限値を超える．計測時間を有界区間に限定することで期待する検査が可能になる．積算消費電力量を一般化して連続値で表せる数値的なコストとみなすと，これは「有界区間でのコスト制約問題」となる．時間有界の到達性解析 [22] や凍結演算子を持つ線型時相論理に対するモデル検査の方法を用いれば良い [19] [21].

このような解析法は，電力バグがあるか否かを自動的に調べ不具合状況を表す反例が出力する．反例を分析して欠陥箇所をつきとめる作業は多くの工数を要するので，欠陥箇所を自動的に発見したい．AI 研究の分野で始まった「モデルベース診断」[26] を基に，SAT ベースの有界モデル検査法 [7] で用いられる「振舞いの論理式表現」を組み合わせる方法がある．欠陥箇所自動発見の方法が VLSI デザイン [27] や手続き型プログラム [12] に適用されて成功している．

本稿では，電力消費振舞いのモデルベース解析法を

Energy Consumption Behavior of Android Applications : Model and Analysis.
Shin Nakajima, 国立情報学研究所, National Institute of Informatics.
総合研究大学院大学 SOKENDAI

紹介する．アンドロイド・フレームワークで生じる電力バグの問題を具体例として，電力消費振舞いの形式モデル，有界解析ならびに欠陥箇所自動発見の方法を述べる．

2 電力バグとモデルベース解析

2.1 アンドロイド多層アーキテクチャ

アンドロイド・フレームワーク（あるいは、アンドロイド）[1] は，リアルタイム・リナックス上に構築された Java ベースのアプリケーション・フレームワークである†1．スマートフォンは，LCD ディスプレイをはじめとして，Wi-Fi，GPS 等のさまざまな機能ハードウェアを持つ．これらのハードウェア資源はリアルタイム・リナックスの管理下におかれている．アプリケーション，いわゆるアプリは，Java で記述されたプログラムであって，アンドロイドが提供する API（オブジェクト群）を通して機能ハードウェア資源を使用する軽い糊付けスクリプトである．

アンドロイドは電池消耗を抑えることを目的として,「積極的な電力削減」ポリシーを採用している．電池消耗に最も大きく関わる機能ハードウェアは LCD ディスプレイである．利用者がしばらくの間，画面をタッチしないでいるとスリープ・モードに移行する．LCD ディスプレイを使わないことで電池の消耗を防ぐ．一方，アプリの機能によってはスリープを抑止したいこともある．たとえば，ストリームデータの配信では，利用者は映像を観ているだけであり，画面に働きかけない．画面タッチがないからといってスリープしていては，映像表示という機能が果たせない．そこで，アンドロイドは，この基本的なポリシーを上書きする電力制御 API (Wake Lock) を提供する．Wake Lock を獲得することで，スリープ・モードへの移行を禁止する．ところが，このような API を誤使用している場合，不必要に長い時間にわたって機能ハードウェアを使用することになり，結果として，電池の消耗が予想外に大きくなる．

†1 フレームワークとリアルタイム・リナックスの両方を合わせて，アンドロイド・オペレーティングシステムと呼ぶこともある．

2.2 アクティビティ

アプリはアプリケーション・プログラムであるが，画面を占有し GUI 部品を通して，利用者とインタラクションするという特徴がある．GUI インタラクションはアクティビティと呼ぶコールバック型の処理機構で実現されている．つまり，アプリの GUI を担うメインスレッドはアクティビティ・オブジェクトを実行する．

アンドロイド・フレームワークは，アクティビティの生成から消滅までのライフサイクル・ステージごとに適切なコールバック・メソッドを起動する．ところが，消滅に関わる振舞いに大きな特徴があり，これが，さらに電力バグの原因となることがある．アクティビティが消滅する際，適切な終了処理を実行するのが普通のことであり，この処理を行うコールバック・メソッドが定義されている．多くの計算システムでは，ここで処理漏れがあっても，アプリケーション・プログラムを実行していたプロセスの消滅と共に，計算資源が解放される．ところが，アンドロイドでは事情が異なる．同じアプリが近い将来ふたたび起動される可能性を想定する．引き続く実行開始処理を高速化する目的で，消滅したアクティビティのメモリ・イメージを残しておく．メモリ不足が生じた時に漸くメモリが上書きされる．消滅したアクティビティの残像中に，電力制御に関わるオブジェクトが生き続けていると，予想外に電池が消耗される結果となる．

2.3 電力バグの発見

現在，電力バグの発見には，プログラムを実行して電力消費量を監視するプロファイラ (例えば [25]) が用いられている．ところが，プロファイラはプログラムの動的テストと同様に，与えたテストデータあるいは設定したテスト環境のもとでの振舞いを調べるだけであって，網羅的な検査ができない．さらに，アプリの機能は，アンドロイド・フレームワークとリアルタイム・リナックスを含む全体によって実現される．電力消費するエンティティは，プログラムではなくハードウェアである．一方，プロセッサならびにメモリといった基本ハードウェア以外の機能ハードウェア・コンポーネントは，アプリが提供する機能に依存して使

用・不使用が決まる．つまり，機能ハードウェアの消費電力はそれを使用するアプリの振舞いに依存する．電力バグはアプリの振舞いに起因するが，消費電力に関わる不具合は機能ハードウェアの電力消費状態として認識される．そこで，フレームワークならびにリナックスの内部まで処理の流れを追跡する必要があることから，プロファイラ開発にはアンドロイド・オペレーティングシステムの改造を伴う（例えば [31]）．

次に，アンドロイド端末のプロセッサに目を移す．ハードウェアの点でも省電力を考えることから，DVFS（例えば[11]）による電力制御が可能なプロセッサを採用する．また，マルチコアプロセッサを利用することも多い．実行負荷が小さい時は，少ないコア数で低い周波数によって実行する．負荷が高まると共に，動作周波数を上げる，あるいはコア数を増やす，等で負荷上昇に対応する．実行コア個数も動作周波数もプロセッサの消費電力に影響する．

実際の測定実験によると，負荷が大きくなるにつれてアプリケーション・プログラムが処理完了するまでの実行時間が長くなり，したがって機能ハードウェアを利用している場合に消費電力が増大することがわかる．一方，DVFS によって実行時間をほぼ一定に保つことが可能な負荷領域も存在する．それでも消費電力のゆらぎが約 30%に達する [20]．つまり，電力バグによる影響が消費電力の 30%内であれば，測定のゆらぎに隠れてしまう．プロファイラがアプリ実行の際の実測に依存していることの限界を示している．

以上から，開発の上流工程で電力消費の振舞い解析を行うモデルベース解析方法 [17] への関心が高まってきた．次節以降で，モデルベース解析の方法を紹介する．

3　振舞い表現

モデルベース解析はアプリから機能ハードウェアまでを含む全体の電力消費振舞いを統一的に表現し解析することで，電力バグを検知する方法である．本研究では，電力消費振舞いの形式表現として，線型ハイブリッド・オートマトン [6] を用いることを考える．

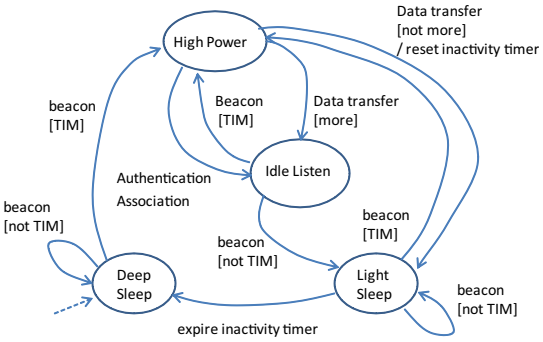

図 1　Wi-Fi STA の振舞い

3.1　Mealy マシン

プログラムが示す離散的な機能振舞いのデザイン表現として，状態遷移システムを用いる方法が知られている（例えば[29]）．オブジェクト指向モデリングで用いる UML ステートダイアグラムなどがある．先に述べたように，アプリのアクティビティはアンドロイド・フレームワークが規定するライフサイクルにしたがって作動する．ライフサイクルは，コールバック・メソッドの実行と完了をイベントとする Mealy マシンで表すことができる [22]．

3.2　Wi-Fi サブシステム

次に，機能ハードウェアの機能振舞いを考える．アンドロイド・タブレットの Wi-Fi サブシステム (Wi-Fi STA あるいは STA) を対象として，Wi-Fi アクセスポイント (AP) と通信している状況での消費電力の変化を測定した結果が報告されている [15]．この測定結果と IEEE 802.11 標準勧告 [2] で示されている機能定義から，STA の振舞いを 4 つの状態間を遷移する状態遷移システム（図 1）として定式化することができる [17]．

Wi-Fi AP は周期的に生成する `beacon` 信号を使って，STA にデータ転送の開始を通知する．STA の初期状態は `DeepSleep` であり，データフレームの送受を行う時には `HighPower` に遷移する．状態 `IdleListen` で引き続くデータフレームの到着を待つ．フレーム中に `no-more-data` フラッグが設定されていると転送終了を示すので `LightSleep` に遷移する．この状態で STA は近い将来に開始される新たなデータ転送に備

え，DeepSleep 状態には遷移しない．その代わりに，タイマーを用い，タイムアウトによって DeepSleep 状態に遷移するように制御する．ここで，DeepSleep は beacon 信号を受信するだけで良いので，使用する電子回路を駆動する小さい電力で良い．一方，HighPower 状態ではデータフレームの中身を調べる必要があることから大きな電力を消費する．つまり，4 つの状態は異なる電力消費率を持つ．

状態 ℓ での単位時間あたりの消費電力量を定数 M^ℓ とする線型近似を導入すると，この状態遷移システムは線型ハイブリッド・オートマトン (LHA) で表すことが可能になる [17]．ある状態に滞留する時間を t とする時，消費電力は $M^\ell \times t$ で表され，連続値（実数）をとる．状態遷移が続く場合を考えると，初期状態から N 回の遷移についての総和をとればよく，$E = \sum_{j=0}^{N} M_j \times t_j$ である．なお，定数 M^ℓ は状態 ℓ での平均値とすれば良くハードウェアの諸元から決まる．

3.3 線型ハイブリッド・オートマトン

上記の消費電力モデルは線型ハイブリッド・オートマトンで表現可能であるが，そのサブクラスを用いる研究が知られている．ストップウォッチ・オートマトンを用いる方法 [9] は状態 ℓ ごとの定数値の違いを考慮しないで全て同じ値とする場合に相当する．異なる M^ℓ 値を表すにはマルチレート時間システム [18] が必要になる．一方，消費電力の表現を考える際には，これよりも簡単な形式モデルである重み付き時間オートマトン (WTA) で良い [19]．消費電力 E は時間経過と共に積算され，検査時に値が必要になるだけであり，状態遷移エッジのガード条件等に現れることはない．つまり，E は観測可能な値であれば良く，積極的に状態遷移の振舞い制御に関わらない．そこで，時間オートマトン (TA) [4] の拡張である WTA [3] を用いて表現する．

以上から，アプリならびに機能ハードウェアの双方を状態遷移マシンで表すことにする．一般に，Mealy マシンは WTA の特殊形になる．つまり，Mealy マシンを，クロック変数なし，重みなしの特殊形と考えれば良い．なお，WTA では，重み値更新式を状態ならびに遷移に付加することができる．本研究での電力消費振舞いでは，機能ハードウェアの状態遷移は瞬時に起こると仮定し，重みの更新式を遷移に与えない．一方，クロック変数や重み変数以外の内部状態変数を導入することで，記述の表現力を向上させることができる．そこで，このような形式モデルを電力消費オートマトン (PCA) と定義した．

検査対象の全体は，アプリならびに機能ハードウェアから構成される．さらに，補助的なオブジェクトも導入する必要があるだろう．そこで，PCA は並行合成できなければならない．WTA と同様に共通するアルファベットに同期して遷移を起こす同期積の方法を採用する [19]．

4 検査する性質

4.1 有界区間のコスト制約問題

次に消費電力量に対する検査性質を考える．単純に考えると，消費電力の総和 E が与えられた上限値 C_{max} を超えないことと表せる．この性質は線型時相論理 (LTL) を用いると $\Box(E \leq C_{max})$ と書ける．しかし，消費電力は積算値であって，機能ハードウェア部品が使用されている時間に比例して増加する．つまり，E の値は時間に対して単調増加するので，C_{max} が有限値であることから，この単純な性質は，ある時点で満たされなくなる．検査する時間区間を限定しなければならない．検査対象の時区間を $[t_s, t_e]$ とすると，$\forall t{\in}[t_s, t_e] \mid E(t) \leq C_{max}$ であろう．一方，検査対象は状態遷移系の一種であることから，対象の振舞いは遷移する状態の列で表される．明示的に時間情報を用いる区間を指定することは難しい．したがって，状態を用いて区間を指定することが望ましい．$\forall \sigma{\in}[\sigma_s, \sigma_e] \mid E(\sigma) \leq C_{max}$ である．なお，指定区間内で新たに生じた消費電力量を検査する場合は，$\forall \sigma{\in}[\sigma_s, \sigma_e] \mid (E(\sigma) - E(\sigma_s)) \leq C_{max}$ である．$E(\sigma_s) = 0$ とすれば上記と同じ形の式になる．

以下では，これを一般化して考える．コスト制約を実数上の線型不等式とする時、「有界区間におけるコスト制約問題」と呼ぶ．コスト制約として，クロック変数や重み変数（消費電力の積算値）に対する数値制約を考えれば良い．

検査性質を具体的に議論するには，有界区間の指定方法とコスト制約の表現方法を考える必要がある．電力消費の振舞いが状態遷移システムで表されることから，区間の開始と終了を状態で印付けすれば良いだろう．今，P_s，P_e を状態命題とする時，LTL を用いて，$P_s \wedge \Diamond P_e$ の形で表せる．一方，コスト制約はクロック変数や重み変数に対する線型不等式である．命題 P_j が成り立つ状態でのクロック変数の値を X_j，重み変数の値を W_j とする時，コスト制約 $C^{\langle C^x;C^w\rangle}(s,e)$ は $(X_e - X_s \leq C^x) \wedge (W_e - W_s \leq C^w)$ のように表現できる．そこで，X_j および W_j の読み出し方法を具体的に考えれば良い．いくつかの方法が考えられる．

4.2 ロジック・モデル検査

第 1 に，凍結限量子を LTL に導入する方法がある [19]．この fWLTL と呼ぶ体系は，TPTL [5] を基に凍結限量する変数値をクロック変数および重み変数とする拡張である．ϕ^x を変数 x が自由出現する LTL 式とする時，TPTL では $\mathcal{I}x.\phi^x$ によって，限量変数 x に ϕ^x が成り立つ状態での時点値を束縛（凍結）する．fWLTL では限量子に注釈を与えるように拡張し，$\mathcal{I}^m x.\phi^x$ によって，m で指定されるクロック変数あるいは重み変数の値を凍結するようにした．なお，LPTL の凍結限量子は時点 τ を明示して $\mathcal{I}^\tau x.\phi^x$ とすれば良い．fWLTL を用いると，先の性質は，

$$\mathcal{I}^X x.\mathcal{I}^W u.(P_s \wedge \Diamond \mathcal{I}^X y.\mathcal{I}^W v.(P_e \wedge C^{\langle C^x;C^w\rangle}(s,e)))$$

のように書く．その結果，消費電力の解析問題を，fWLTL 式の PCA に対するロジック・モデル検査として定式化できる．

ところが，fWLTL は TPTL を含む一方，TPTL は MTL を含むことが知られている．MTL (Metric Temporal Logic) 式の TA に対するロジック・モデル検査は一般の連続時間に対して決定不能である．PCA が TA の拡張であることを考慮すると，fWLTL と PCA に関するロジック・モデル検査も一般には決定不能である．そこで，文献 [21] では，近似手法を導入し，さらに検査 fWLTL 式のパターンを限定して，Real-Time Maude [24] の LTL モデル検査に翻訳する方法を採用した．

4.3 時間有界探索

第 2 に，有界な区間での検査に限定すれば良いことから，fWLTL 式を使う場合に比べて簡便な方法も考えることができる．検査対象 PCA が生成可能な経路の全体から特定の性質を満たす経路だけを選び出し，その選んだ経路についてのみ，コスト制約を評価する．クロック変数および重み変数を凍結する状態を何らかの方法で別途指定する．今，選び出す経路の条件をガイド制約 φ_G で表す時，検査対象 PCA A に対して，$A \models \varphi_G \Rightarrow C^{\langle C^x;C^w\rangle}(s,e)$ を調べることに相当する．

先の fWLTL の場合では，クロック変数および重み変数を凍結する状態の指定方法が明確であった．一方，この方法では，検査対象の PCA 中に明示的に書き表す必要がある．コスト制約として，冒頭の簡単な性質 $\Box(E \leq C_{max})$ に限定する場合に有効な方法である[†2]．文献 [22] では，Real-Time Maude の時間有界横型探索の方法を用いた実験例が報告されている．

5 スコープ有界検証

5.1 有界モデル検査の問題

一般に，ロジック・モデル検査や状態空間探索による自動検証の方法では，いわゆる「状態空間の爆発」という問題が生じる．これに対する解決策として提案されたのが，SAT 法を用いる有界モデル検査 (BMC) である [7]．検査対象システムの実行可能な経路の全体を（命題）論理式 φ_{TF} にエンコードする．この時，遷移列の経路は有限長に限定する．検査性質を表す表明 φ_{AS} に対して，BMC 問題は $\neg(\varphi_{TF} \Rightarrow \varphi_{AS})$ の充足性判定になる．この論理式全体が充足する時，命題変数値の割当て（モデル）は，φ_{TF} が φ_{AS} 満たさない理由（反例）を表す．歴史的な経緯から BMC と呼ばれるが，以下，最近の呼び方にしたがって，スコープ有界検証と呼ぶ．

検査式 φ_{AS} を，前節と同様に，$\Box\psi$ に限定する場合について，もう少し詳しく説明する．検査対象 PCA の状態を表現する変数の集合を U とする．U^j を添字 j で区別する集合とする．特に初期状態は U^0 である．

†2 ただし有界区間に限定．

論理式 φ_{TS} は初期状態から K ステップでの遷移全体を表す．初期状態を定義する論理式を I，遷移関係を T，$T^j(U^{j-1},U^j)$ を状態 U^{j-1} から U^j への遷移関係とする．この時，K ステップ遷移は

$\varphi_{TF}{=}I(U^0) \wedge (\bigwedge_{j=1..K} T^j(U^{j-1},U^j))$

となる．次に，$\neg(\varphi_{TF} \Rightarrow \varphi_{AS}){=}\ \varphi_{TF} \wedge \neg\varphi_{AS}$ であることを利用する．$\Box\psi$ の否定は $\Diamond\neg\psi$ であり，K ステップの範囲で，$\Diamond\neg\psi{=}\bigvee_{j=1..K}\neg\psi(U^j)$ である．したがって，スコープ有界検証の方法は，次の論理式 φ_{safety} の充足性判定問題になる．

$$\varphi_{safety}{=}I(U^0) \wedge (\bigwedge_{j=1..K} T^j(U^{j-1},U^j)) \wedge (\bigvee_{j=1..k}\neg\psi(U^j))$$

5.2 PCA の有界区間解析

スコープ有界検証の方法を導入するには，検査対象の φ_{TF}，つまり遷移関係の論理式表現を定義しなければならない．PCA は TA の拡張であることから，TA に対する遷移関係定義 [28] をもとにした方法を適用すれば良い．

PCA は，離散遷移と時間遷移という 2 種類の遷移を持ち，これらの論理式表現を考える [23]．状態を表す変数 U を act（イベント記号），at（ロケーション），x（クロック変数），w（重み変数），v（内部変数）からなるとする．

状態 ℓ から ℓ への遷移を $e = \ell^{a,g,r,u}\!\!\longrightarrow\!\ell'$ とする．ここで，a はイベントを表すアルファベット，g はガード条件，r はリセットクロックの集まり，u は内部状態の更新処理，を表す．この時，遷移 e に対する離散遷移の論理式表現 $T(e)$ は次のように定義できる．

$$T(e) \ = (at = \ell \wedge at' = \ell' \wedge act = a \wedge g \wedge x' = z \wedge w' = w \wedge v' = f(v) \wedge Inv(\ell')(x'))$$

$Inv(\ell)$ は状態 ℓ のクロック不変式，$f{\in}u$ は値更新関数である．

時間遷移は，δ を適切な経過時間を示す正の実数として，論理式表現 $D(\delta,\ell)$ を次のように定義する．ここで，時間遷移を表す特別なアルファベット $delay$ を用いた．

$$D(\delta,\ell) \ = (at' = at \wedge act = delay \wedge x' = x + \delta \wedge w' = w + M^{\ell}{\times}\delta \wedge v' = v \wedge Inv(\ell')(x'))$$

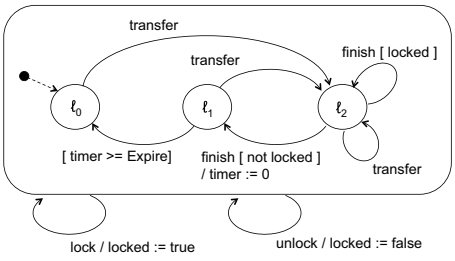

図 2　PCA の例

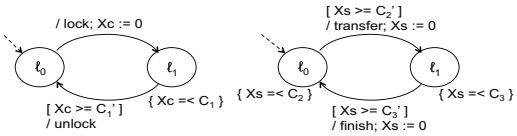

図 3　Client（左側）と Wi-Fi AP（右側）

本稿で行う解析は，有界区間でのコスト制約問題であり，ガイド制約 φ_G を用いる．これは，$\varphi_{TF}{\wedge}\neg(\varphi_G{\Rightarrow}\varphi_{AS})$ であり，$(\varphi_{TF}{\wedge}\varphi_G){\wedge}\neg\varphi_{AS}$ の形のスコープ有界検証問題である．

5.3 例

図 2 に示した簡単な例を考える．これは，図 1 を単純化した W-Fi STA の PCA であり，3 つの状態からなる．状態 ℓ_0 は deep-sleep を表し，ここでの単位時間あたりの消費電力 (M^0) は小さい．状態 ℓ_1 の light-sleep では，その値は M_1 は M^0 よりも大きい．一方，状態 ℓ_2 ではデータフレームのデコード処理を行うので最も大きな値 M_2) を持つ．状態遷移の図から，Lock されていない時，データフレーム転送終了と共に、状態 ℓ_2 から ℓ_1 へ遷移する．この時，タイムアウトで ℓ_1 から ℓ_0 へ遷移するようにタイマーを設定する．逆に，Lock されている時，転送が終了しても，状態 ℓ_2 に留まる．

次に，図 3 に示した Client と Wi-Fi AP からなる全系のシナリオを考える．文献 [22] の例題を簡単化したものに相当する．消費電力の総和 E が最大値 C_{max} 以下であるという簡単な検査性質を考える．先に論じた $\forall\sigma{\in}[\sigma_s,\sigma_e] \mid E(\sigma) \leq C_{max}$ である．LTL の論理式 $\varphi_{AS} = \Box(E \leq C_{max})$ で表現できる．また，検査対象として，`lock` で始まり，`finish` を経て， `transfer`

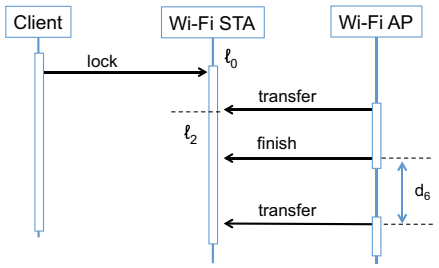

図 4　反例のメッセージ列

で終わる区間を考え，これをガイド制約 φ_G として表す．つまり，φ_G =lock∧◇(finish∧◇transfer) で，状態 σ_s として lock 実行後を考え，$E(\sigma_s) = 0$ としていることに相当する．この時，スコープ有界検証の問題は，$\neg\varphi_{AS}$=◇$(E > C_{max})$ であるから，$\varphi_{TF} \wedge \varphi_G \wedge \Diamond(E > C_{max})$ になる．ここで，論理式 φ_{TF} は 3 つのオートマトンを並行結合した全系が生成する遷移系列の全体を表す．

実際の実験では，クロック制約 (C_1, C_2, and C_3), ガード条件 (C'_1, C'_2, and C'_3), および検査性質 (C_{max}) 等の定数に具体的な値を与える．特に，C_2 と C'_2 を大きな値にすることで，Wi-Fi STA (図 2) が Lock されている時に検査性質が満たされなくなる状況を作った．

具体的な値 K を選んでスコープ有界検証を行う†3と反例が生成されて，(lock, δ_2,transfer, δ_4, finish, δ_6,transfer) を得る．

図 4 はメッセージ系列図として表現した．3 つのオートマトンは並行実行されるので，このメッセージ列は，検査性質を満たさないインターリーブのひとつになる．Wi-Fi STA は初期状態 ℓ_0 にいるが，transfer によって状態 ℓ_2 に遷移する．その後，finish があるにも関わらず，Lock されているので，同じ状態に留まる．d_6 時間経過後，transfer があり，$d_6 \geq C'_2$ を満たす．C'_2 の値を大きくとったことから，この時点での電力消費の総和 E は検査条件を満たさない．E は $M_0 \times d_2 + M_2 \times d_4 + M_2 \times d_6$ である．

6　欠陥箇所特定

6.1　モデルベース診断

先に述べたようにスコープ有界検証は論理式 φ_{BMC} の充足性判定問題に帰着される．ここで，$\varphi_{BMC} = (\varphi_{TF} \wedge \varphi_G) \wedge \neg\varphi_{AS}$ である．反例が生成される場合，その反例，あるいは反例に含まれる情報であって φ_{AS} を満たさないことを表す最小の情報をエンコードした論理式を $\overline{\varphi_{TF}}$ とする時，充足不能となる論理式 $\overline{\varphi_{TF}} \wedge \varphi_{AS}$ を φ_{FL} とする†4．論理式 φ_{FL} が充足不能な時，その充足不能コアに欠陥箇所が含まれる．

欠陥箇所自動特定の系統的な枠組みとしては「モデルベース診断 (MBD)」[26] がある．MBD では充足不能な論理式 φ_{FL} の充足不能コアを conflicts と呼ぶ．一方，φ_{FL} の構成部分式の一部を除去することで，φ_{FL} が充足させることができる．この除去する部分が欠陥箇所であり，diagnoses と呼ぶ．形式的には conflicts は φ_{FL} の MUSes であり，diagnoses は MCSes である．ここで，MUSes は MUS(Minimal Unsatisfied Subset) の集合，MCSes は MCS (Minimal Correction Subset) の集合（例えば [8]）．また，MCSes と MUSes はヒッティングセット関係で結ばれている．MCSes あるいは MUSes を求めれば良い．本稿では，MCSes を MaxSAT 問題の解として求める方法を採用する [14]．

論理式 φ_{FL} が充足不能な時，MaxSAT 問題は，充足する部分式の個数が最大になるように除去する部分式を求める．また，論理式 φ_{FL} から MCSes を求める際，φ_{AS} は除去部分式の対象にしたくない．そこで，Partial MaxSAT (pMaxSAT) 問題を用いる．部分式ごとに，*soft* あるいは *hard* を指定する．この時，pMaxSAT 問題は，すべての *hard* 部分式を満たすと同時に，満たす *soft* 部分式の個数が最大になるように，つまり，除去する *soft* 部分式の個数が最小になるように選ぶ．この方法を採用すると，φ_{AS} を *hard* にする．さらに，削除の候補となる φ_{TF} についても，欠陥候補としたくない部分式を *hard* にし，候

†3　Yices-1 [10] を利用．
†4　ここで，$\overline{\varphi_{TF}}$ は反例なので φ_G を考慮済みである．

補としたい部分式を *soft* とすれば良い．何を *soft* とするかは，想定している「故障モデル」に依存する．

6.2 PCA の欠陥箇所特定方式

PCA を対象として欠陥箇所特定を自動化するには，検査対象の振舞いを表す論理式表現，検査性質に加えて，故障モデルを決める必要がある [23]．故障モデルは検査したい性質に合わせて考えるので，本稿で扱っている消費電力の問題を振り返る．

$\sum_{j=0}^{N} M_j \times t_j$ で表される総消費電力 E に対して，検査式を $\Box(E \leq C_{max})$ とした．M_j は定数なので，結局，この性質は経過時間 t_j に依存する．t_{i-1} から t_i のような経過時間の区切りは離散遷移が起こす．

一方，PCA の定義から，状態遷移列の時間経過に影響を与えるのは，状態に付された不変式，遷移エッジ上のガード条件およりリセット条件である．つまり，ある遷移列に対して，上記の検査式が満たされるか否かは，その遷移列上のクロッック変数に関わる制約条件との関係で決まる．そこで，本稿の故障モデルを，これらクロック変数に関連する制約条件の出現箇所から構成する．以下，p^H という表記は，p が *hard* であることを示す．同様に p^S は p が *soft* である．

$$
\begin{aligned}
T(e) \ &= ((at = \ell)^H \wedge (at' = \ell')^H \wedge (act = a)^H \\
&\quad \wedge (g)^S \wedge (x' = z)^S \wedge (w' = w)^H \\
&\quad \wedge (v' = f(v))^H \wedge (Inv(\ell')(x)')^S) \\
D(\delta, \ell) \ &= ((at = at')^H \wedge (act = delay)^H \\
&\quad \wedge (x' = x + \delta)^H \wedge (w' = w + M^{\ell} \times \delta)^H \\
&\quad \wedge (v' = v)^H \wedge (Inv(\ell')(x'))^S)
\end{aligned}
$$

今，$\overline{\varphi}_{TF}$ を決めた時，つまり反例を決めた選んだ時，この論理式がエンコードしている遷移の列 $\{\rightarrow_j\}$ から，クロック変数に関わる部分式を集めて φ_{CLK} を作るとする．この時，欠陥箇所特定の問題は，$\varphi_{CLK} \wedge \varphi_{AS}$ のクロック制約問題になる．

6.3 例

具体例として前節のスコープ有界検証の例について欠陥除去の方法を説明する（図 4 参照のこと）．

$\overline{\varphi}_{TF}$ を反例から構成し，$\overline{\varphi}_{TF} \wedge \Box(E \leq C_{max})$ の論理式 φ_{FL} に対して，pMaxSAT ソルバー Yices-1 によって MCS を網羅的に求める．Wi-Fi AP (図 3 右側) の要素に関連した 2 つの MCS からなる MCSes $\{\{[Xs \geq C'_2]\ on\ \ell_0 \rightarrow \ell_1\}, \{(Xs := 0)\ on\ \ell_1 \rightarrow \ell_0\}\}$ を得た．

最初の MCS $\{[Xs \geq C'_2]\ on\ \ell_0 \rightarrow \ell_1\}$ は `finish` とその後の `transfer` の間の時間 C'_2 が大きいことを示す．Wi-Fi STA は，`finish` 後に状態 ℓ_2 で `transfer` を待っており，この経過時間が大きいと多くの電力を消費する．つまり，C'_2 を小さくすれば良い．

2 つめの MCS $\{(Xs := 0)\ on\ \ell_1 \rightarrow \ell_0\}$ を参照すると，クロック変数 X_s を 0 よりも大きな値に初期化すれば状態 ℓ_0 から ℓ_1 へのエッジ上のガード条件が，より短い経過時間で発火するようにできることを示唆している．ところが，PCA の定義から，クロック変数に任意の値を設定することはできない．単に，リセットするだけである．つまり，この 2 つめの MCS は見かけの欠陥箇所であって，除去可能な原因ではない．

7 むすび

アンドロイド・アプリの消費電力振舞いを対象としたモデルベース解析の方法を紹介した．PCA と呼ぶ重み付き時間オートマトンが中心的な形式モデルである．この時，消費電力の解析は有界区間でのコスト制約の問題である．この問題を解くのに，ロジック・モデル検査や時間有界探索の方法があった．本稿では，SAT/SMT ベースのスコープ有界検証の方法を導入し，さらに，欠陥箇所自動特定の方法を述べた．PCA 等の形式的な定義の詳細については，関連文献 [19] [23] を参照されたし．

現状，小さい例題を用いて提案方法の妥当性を確認した段階である．PCA は線型ハイブリッド・オートマトンのサブクラスでありながら，興味深い応用問題に適用できる．サブクラスといっても TA よりも表現力が大きいことから，一般には，自動解析アルゴリズムの複雑さという点で難しい問題が残る．ただし，電力消費は有界区間での解析という制限された問題なので，このことを利用して，複雑さを軽減した解析方法が利用可能といえる．

謝辞 本研究は JSPS 科研費 24300010，26330095

の助成を受けたものです.

参 考 文 献

[1] Android. `http://developer.android.com`.

[2] IEEE Standard 802.11, Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications , 1999.

[3] R. Alur, C. Courcoubetis, and T.A. Henzinger. Computing Accumulated Delays in Real-Time System, In *Proc. CAV 1993*, pp.181-193, 1993.

[4] R. Alur and D.L. Dill. A Theory of Timed Automata, *TCS*, 126, pp.183-235, 1994.

[5] R. Alur and T.A. Henzinger. A Really Temporal Logic, *J. Assoc. Comp. Machin.*, Vol.41, No. 1, pp.181-204, 1994.

[6] 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.

[7] A. Biere, A. Cimatti, E.M. Clarke, and Y. Zhu. Symbolic Model Checking without BDDs, In *Proc. TACAS 1999*, pp.193-207, 1999.

[8] A. Biere, M. Heule, H. Van Maaren, and T.Walsh (eds.). *Handbook of Satisfiability*, IOS Press, 2009.

[9] A. Brekling, M.R. Hansen, and J. Madsen. MoVES – A Framework for Modeling and Verifying Embedded Systems, In *Proc. ICM2009*, pp.149-152, 2009.

[10] B. Dutertre and L. de Moura. The Yices SMT Solver. http://yices.csl.sri.com

[11] J.L. Hennessy and D.A. Patterson. *Computer Architecture : A Quantitative Approach (5ed.)*, Morgan Kaufmann 2011.

[12] S. Lamraoui and S. Nakajima. A Formula-based Approach for Automatic Fault Localization of Imperative Programs. In *Proc. ICFEM'14*, pp.251-266, 2014.

[13] E.A. Lee and S.A. Seshia. *Introduction to Embedded Systems - A Cyber-Physical Systems Approach*, `LeeSeshia.org`, 2011.

[14] M.H. Liffiton and K.A. Sakallah. Algorithms for Computing Minimal Unsatisfiable Subsets of Constraints. *Automated Reasoning*, 40(1), pp.1-33, 2008.

[15] J. Manweiler and R.R. Choudhury. Avoiding the Rush Hours: WiFi Energy Management via Traffic Isolation, In *Proc. MobiSys'11*, pp.253-266, 2011.

[16] A. Morgado, M. Liffiton, and J. Marques-Silva. MaxSAT-Based MCS Enumeration. In *Proc. HVC'12*, pp.86-101, 2013.

[17] S. Nakajima. Model-based Power Consumption Analysis of Smartphone Applications, In *Proc. ACES-MB 2013*, pp.5:1-5:10, 2013.

[18] S. Nakajima. Everlasting Challenges with the OBJ Language Family, In *Proc. SAS 2014*, pp.478-493, 2014.

[19] S. Nakajima. Model Checking of Energy Consumption Behavior, In *Proc. 1st CSDM Asia*, pp.3-14, 2014.

[20] S. Nakajima and M. Toyoshima. Behavioral Contracts for Energy Consumption, *Ada User Journal*, vol.35, no.4, pp.266-271, 2014.

[21] S. Nakajima. Using Real-Time Maude to Model Check Energy Consumption Behavior, In *Proc. FM 2015*, pp.378-394, 2015.

[22] S. Nakajima. Formal Analysis of Android Application Behavior with Real-Time Maude, In *Proc. CPSNA 2015*, pp.7-12, 2015.

[23] S. Nakajima and S. Lamraoui. Fault Localization of Energy Consumption Behavior using Maximum Satisifiability, In *Proc. CyPhy 2015*, (to appear) 2015.

[24] P.C. Ölveczky and J. Meseguer. Semantics and Pragmatics of Real-Time Maude, *Higher-Order and Symbolic Computation*, vol.20, no.1-2, pp.161-196, 2007.

[25] A. Pathak, Y.C. Hu, and M. Zhang. Bootstrapping Energy Debugging on Smartphones: A First Look at Energy Bugs in Mobile Devices, In *Proc. Hotnets'11*, pp.5:1-5:6, 2011.

[26] R. Reiter. A Theory of Diagnosis from First Principles. *Artificial Intelligence*, 32(1), pp.57-95, 1987.

[27] S. Safarpour, H. Mangassarian, A. Veneris, M.H. Liffiton, and K.A. Sakallah. Improved Design Debugging using Maximum Satisfiability, In *Proc. FM-CAD'07*, pp.13-19, 2007.

[28] M. Sorea. Bounded Model Checking for Timed Automata, *ENTCS*, 68(5), pp.116-134, 2002.

[29] R.J. Wieringa. *Design Methods for Reactive Systems*, Morgan Kaufmann Publishers 2003.

[30] J.M. Wing. Cyber-Physical Systems, *Computer Research News*, 21(1), p.4, 2009.

[31] L. Zhang, M.S. Gordon, R.P. Dick, Z.M. Mao, P. Dinda, and L. Yang. ADEL : An Automatic Detector of Energy Leaks for Smartphone Applications, In *Proc. CODES+ISSS'12*, pp.363-372, 2012.