電力モデルに基づくアプリ消費電力可視化ツールの評価

神山剛[†1]　稲村浩[†1]　太田賢[†1]

本稿は Android アプリケーションの実利用環境において利用可能なアプリ消費電力の評価手法を提案する．本手法では，スマートフォンを構成する各ハードウェアコンポーネントの特性と消費電力の関係から生成した端末の消費電力モデルを用いることで，アプリ消費電力の推定を可能にする．本稿では，近年の端末の消費電力を妥当な精度で推定できること，推定に必要なログ収集の負荷が低いことを要件とした評価手法を実現するため，マルチコア CPU やモバイル無線インタフェースとその特徴を考慮したモデル拡張を行う．提案本手法において，一般的なアプリ利用のシナリオを対象に 10%前後の誤差で電力推定できること，3.8%程度の低いオーバヘッドでログ収集が実現できることを確認した．

Evaluation of a Model-based Energy Profiler for Android Applications

TAKESHI KAMIYAMA[†1]　HIROSHI INAMURA[†1]　KEN OHTA[†1]

1. はじめに

近年，Android 端末をはじめとするスマートフォンの普及に伴い，利用者から電池持ち時間の改善が要望されるようになった．

スマートフォンが消費する電力は，無線インタフェースや CPU 等のハードウェアリソースの稼動により決定され，その利用率はアプリケーションプログラム（以降，アプリ）の挙動による．従って，電池利用において，例えば必要以上に通信を行うなどの非効率な挙動を示すアプリを改善することで電池持ちの改善が可能である．古庄ら 1)は特定アプリにおいて利用者の平均的な操作パターンに対しアプリの挙動を最適化することでそのアプリが消費する電力（アプリ消費電力）を低減させられることを示した．

この知見を進めるなら，アプリが市場に出てからではなく開発時点でそのコードの消費電力を現実的な精度で開発者にフィードバックし，機能実現に対する電力コストを意識してもらうことが有効であると考えられる．

本研究は，開発者自身によるアプリ消費電力の最適化の実現のために，開発者が容易に利用可能な AndroidOS でのアプリ消費電力の評価手段を提供することを目的とし、アプリ実行時に現実のハードウェアで消費される電力を精度よくソフトウェアのみで推定することによるアプリ消費電力の評価手法およびツールを提案する．本手法では，スマートフォンを構成する各ハードウェアコンポーネントの特性と消費電力の関係から生成した端末の消費電力モデルを用いることで，アプリ消費電力の推定を可能にする．

アプリ消費電力の評価手段をソフトウェアで実現することは、測定器を用いた従来の測定手段に起因する課題を回避するとともに低コスト化，流通の容易性といったメリットがある．機材を用いた従来の電力測定をアプリ開発の現場に持ち込む際には、必要な機材の手配や開発期間の兼ね合いといった制約があり困難が伴う．さらに屋外や移動中など実際のアプリ使用状況において，専用の測定器を使用しながら測定することは自然なアプリ利用を妨げる，といった諸点を考慮しなければならない．昨今のアプリ開発環境は SDK(Software Development Kit)の形態で配布され，実機エミュレータを内包するなどソフトウェアのみで完結しており，アプリ消費電力の評価手段を開発者に提供する際にも，同じアプローチを取ることで安価に広く配布することが可能になる．

本稿の構成を以下に示す．第 2 章では本研究の課題とアプリ消費電力の評価手法に対する要件設定を行い，第 3 章では関連研究について紹介する．第 4 章では，提案手法が取り入れる電力モデルの基本設計と近年のスマートフォンのハードウェア特性に対応するためのモデル拡張について述べ，第 5 章では，拡張した電力モデルを用いたアプリ消費電力可視化ツールを紹介する．最後に，第 6 章では本ツールの電力推定精度と推定に必要なログ収集のオーバヘッド評価し，第 7 章でまとめと今後の課題を述べる．

2. 課題と要件

前述のアプリ消費電力の評価のためには，ソフトウェアにより精度よくハードウェアの消費電力を推定する手段を実現することが必要である．この手段の実現のためには以下の課題がある．

課題 1) ハードウェアの特性をモデル化した精度の高い消費電力推定手段を供えること．

課題 2) 実際のアプリ利用状況をデータ化しこれに基づいた推定が可能であること．

†1 株式会社 NTT ドコモ　先進技術研究所
Research Laboratories, NTT DOCOMO, Inc.

課題 1)の解決によってソフトウェアのみによる消費電力の測定手段の構築が可能になり，全ての開発者に測定機材を配備する必要がなくなる．さらに，アプリ消費電力の評価が自然なアプリ利用の妨げになってはならない．課題 2)の解決は，アプリの利用シーンに過度に介入することなく簡易なデータ取得のみによって評価を成立させるために必要である。これにより，実際の様々な利用シナリオ・環境における評価が可能になると考える．

我々はスマートフォンの電力モデルとアプリ実行時のログを用いた，アプリ消費電力可視化ツールを提案する．

課題 1 の解決のため，電力推定手法を以下のとおりとする．石原ら 3)の提案する電力モデルを採用し，ハードウェアコンポーネント毎の稼動量と実測した消費電力をもとに，対象とする端末の特性に適合させるべく重回帰分析によりパラメータ係数を求めることで電力モデルを作成する．近年のスマートフォンにおいても同様の精度で推定するために，マルチコア CPU におけるコア数や周波数の切替えなど近年のハードウェアコンポーネントとその特徴を考慮すべくモデル拡張を行う．

課題 2 の解決のため，端末上でアプリを実行している間に所定のログを収集しておき，そのログからハードウェアコンポーネント毎の稼動量をパラメータとして求め，電力モデルに適用しアプリ消費電力を推定する．本ツールの構成では測定器による実測は電力モデル作成時のみであり，個別のアプリ・利用シナリオ毎の実測は不要である．

上記を踏まえ，以下 2 点の要求条件を設定するものとし，後述の評価における指標とする．

[1]システム全体を対象に電力推定精度が妥当であること

[2]実利用時のログ収集にかかる負荷が低いこと

3. 関連研究

本章では既存の電力モデルに基づく電力推定手法やアプリ電力評価手法についてまとめ，2 章に述べた本研究における課題への適合性について述べる．

電力モデルを用いた電力推定手法は，線形式で特定のハードウェアコンポーネントの消費電力をモデル化しておき，電力推定の際には，モデルパラメータを抽出するためのログを取得し，モデルへの入力とすることで推定する手法が提案されている．

Lee ら 2)は CPU の命令をパラメータとした線形モデルを提案し，高い精度でプロセッサの電力を推定している．一方，CPU など特定のコンポーネントに限定されていることや，あるいは推定のためのパラメータ同定には高負荷なハードウェアエミュレーションが必要であり，アプリの実利用時に用いることは現実的ではない．

石原ら 3)，Kaneda ら 4)の手法は，システムを構成する主要なコンポーネントの消費電力を，CPU 稼働率など OS レベルで容易に取得可能なデータをパラメータとしてモデル化することで，システム全体の消費電力を精度良く推定する手法を提案している．これらの手法は，モデルの設計概念としてシステム全体をカバーしやすく，様々なコンポーネントに対応するログ収集のオーバヘッドが小さいという利点がある．一方で，モデル自体がマルチコア CPU とその周波数制御や，3G/LTE といったモバイル無線インタフェースとその挙動など，近年のスマートフォンのハードウェア構成と動作が考慮されていない．

近年のスマートフォンとそのアプリ評価を対象にした関連研究としては Michigan 大による ARO(Application Resource Optimizer)と呼ばれるツール 5)が提案されている．本ツールでは，モバイル無線インタフェースの電力モデルに特徴がある．3G/LTE の無線通信では，RRC（Radio Resource Control）と呼ばれる端末と無線基地局間における複数種類の無線チャネル割当を制御している 6)．この割当状態（以下，RRC State）をパラメータとした電力モデルを用いることで，モバイル無線通信における電力を精度よく推定している．ただし，RRC State を特定するために必要なログとしてパケットキャプチャを用いており，ログ取得に特権が必要であることと，ログ取得にかかるオーバヘッドが大きいことから，実機におけるアプリの実利用時への適用は困難である．本研究では対象開発者を広く取るためこのような制約は認められない．

以上より，既存研究では，本研究が目指すアプリの実利用におけるアプリ消費電力評価における前述の課題・要件全てを満たすことはできない．本稿では，システム全体をカバーしやすくログ収集オーバヘッドの小さい既存研究 3)のモデルを踏襲しつつ，かつ ARO のように近年のハードウェアコンポーネントの特徴を考慮したモデル拡張を行うことにより，上記要件を満たすアプリ消費電力の評価手法を提案する．

4. モデルによる電力推定

4.1 モデル概要

電力推定に用いる端末の電力モデルについて述べる．

本電力モデルは，CPU など端末を構成するハードウェアコンポーネント毎にモデル化された電力の総和をとることで，端末全体の電力を表現する．

$$P_{est} = c_{offset} + \sum_{i}^{N} c_i p_i$$

前記コンポーネント毎の電力は，コンポーネント i 毎にその稼働状態を示す値をパラメータ p_i とし，所定のパラメータ係数 c_i およびオフセット定数 c_{offset} からなる数式で求められる．

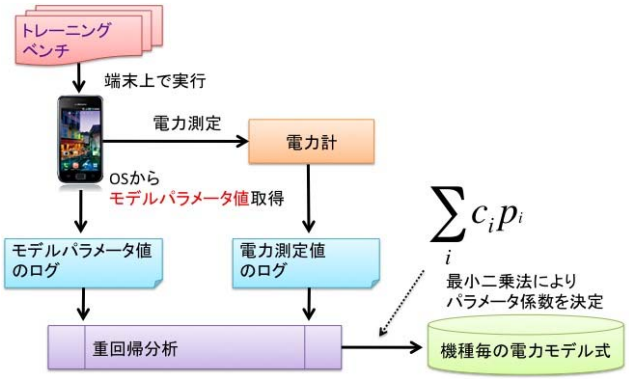

Fig. 1 電力モデルの生成方法

モデルの生成方法を Fig. 1 に示す．モデルの生成は従来手法と同様，コンポーネント毎に着目したトレーニングベンチを端末上で動作させた際の端末全体の消費電力と各モデルパラメータ p_i の実測値をサンプルデータとして上記モデル式に代入し，重回帰分析によりパラメータ係数 c_i を求める手法となる．しかしながら、本研究においては，実際に様々な機種の電力モデルを用いて評価することを想定しているため，モデルの基本構造として，CPU 稼働率など一般性があり抽象度の高いパラメータを用いることで，モデル生成過程や使用時における機種依存性を極力排除できるようにする．

本提案手法におけるモデル生成では，従来研究 3)のモデル設計を基にパラメータを追加し，モデル拡張を行う．なお，本稿では，拡張箇所のみ 4.2 に後述し，それ以外のコンポーネントも含めた全体のモデル式および生成結果については 4.3 に後述する．

4.2 モデル拡張

4.2.1 マルチコア CPU 対応

近年のスマートフォンでは, マルチコア CPU が搭載されており，また，これらは省電力化などの目的でシステム稼働状況に応じて使用するコア数や周波数を切り替える制御がなされていることが一般的である．従来研究 2)は，シングルコア CPU を対象に，CPU 稼働率のみをパラメータにしたモデルであるため，前述の制御による影響を考慮した電力推定を行うことができない．よって，本提案手法による CPU 電力を説明すべきパラメータの拡張を行うことにより，これらの挙動も考慮したモデル生成を行う．

本提案手法における CPU の電力 P_{cpu} を以下に示す．

$$P_{cpu} = P_{core1} + P_{core2} + \cdots + P_{coreN}$$
$$= \sum_i^N c_i p_i = \sum_i^N c_i f_i u_i$$

ここで，$P_{core1} \sim P_{coreN}$ はそれぞれ CPU を構成するコア毎の電力であり，P_{cpu} はコア毎の電力を加算したものである．近年の周波数制御はコア毎になされることから各々独立に

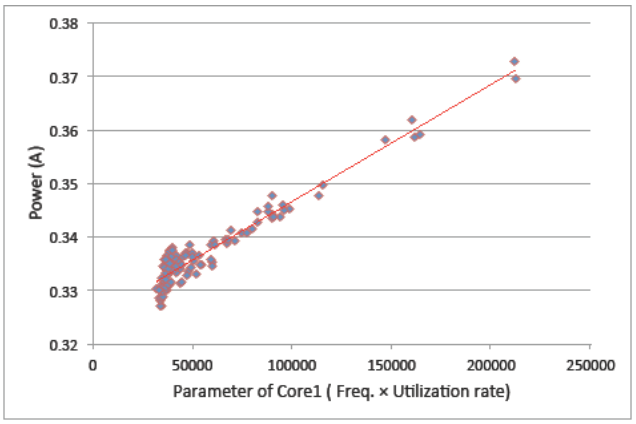

Fig. 2 消費電力と CPU パラメータの関係

モデル化する．次に，コア i の電力 P_{corei} を構成する c_i，p_i，f_i，u_i は，それぞれ，コア i に対するパラメータ係数，パラメータ，動作周波数，CPU 稼働率である．本提案手法においては, 周波数毎の性能差を加味しつつ CPU による仕事量を示す値をパラメータとするため, 動作周波数に CPU 稼働率を乗算した値をパラメータとした．

このパラメータの妥当性を確認するため，4.3 に後述する端末を用いて予備検証を行った．同一タスクに対して動作周波数を計測範囲で任意に設定する試験用プログラムを端末で動作させ, その動作パターン毎に端末の消費電力と，パラメータ導出の元になる動作周波数および CPU 稼働率を測定した．上記試験により測定したパターン毎の消費電力とパラメータとの関係を Fig. 2 に示す．本提案手法において設定したパラメータと消費電力の間に線形性が見られることから，CPU 電力モデルのパラメータとして妥当であると考えられる．

4.2.2 モバイル無線インタフェース対応

3G/LTE 無線通信では，RRC と呼ばれる端末と無線基地局間における複数種類の無線チャネル割当制御を行う仕組みがある．RRC は，多数の端末が在圏する基地局配下における無線リソースの効率利用のため，各端末からの通信要求に応じて，端末と無線基地局間で複数種類のチャネル割当状態（以下，RRC State）を切り替えるなどの制御を行っている．

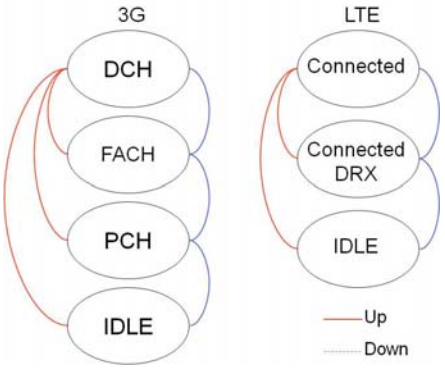

Fig. 3 3G/LTE の RRC State 遷移の例

3G/LTE における RRC State 遷移の概略図を Fig. 3 に示す．例えば 3G においては，アプリ利用時など高スループットでのデータ送受信を行うために用いる個別チャネル（DCH），低スループットでデータ送受信が可能な共通チャネル（FACH），無通信状態で待機的な状態として一定期間割り当てられるチャネル（PCH），データ送受信のための無線リソースの解放状態を指す IDLE など，複数の RRC State が定義されており，データ送受信の発生により上位の DCH に，一定期間の無通信状態を検知することにより下位の State に遷移するなど，State 間を遷移するための条件が規定されている．端末における消費電力という点では，State 毎に消費電力が大きく異なり，上位の State であるほど大きな消費電力を消費することが知られている．

ARO ではこの割当状態（以下，RRC State）をパラメータとしたモデルを用いることで，モバイル無線通信における電力を精度よく推定することが可能である．加えて，端末上で，直接的に RRC State を取得することは一般にできないため，ARO ではパケットキャプチャを端末上で収集し，データ送受信のタイミングや量から RRC State を推定する手法を提案している．しかし，3 章に述べたように，この手法では端末上でログ収集オーバヘッドが大きいことに加え，セキュリティ上の理由から特権が必要な手段であることから，広く一般の開発者が利用するアプリ評価手段を実現する上で適切さに欠ける．

本提案手法では，ARO と同様の RRC State 推定ロジックを踏襲するが，前述の問題を回避するため，パケットキャプチャ以外で AndroidOS で容易に取得可能なデータをもとにするよう改良した．具体的には，一定周期でデータ送受信量の統計値を取得し，周期ごとに送受信データの有無や量を確認し，これをもとに推定を行う方法をとる．パケットキャプチャと比較し，収集するログのデータ量に少なくなることから，ログ収集オーバヘッドを大幅に削減できると考えられる．

パケットキャプチャはデータ送受信をトレースしたデータであり，ARO はトラフィック発生タイミングや状況を正確に把握し，RRC State 遷移推定に反映できるが，本提案手法では一定周期のサンプリングデータから遷移推定を行うという仕組み上，推定結果に 1 周期分の遅延が生じるという欠点がある．本提案手法の実装にあたっては，RRC State や消費電力推定に大きな誤差を起こさない範囲で，オーバヘッドを削減することを優先するため，ログ収集の周期を 1 秒とする．

4.3 モデル生成結果

本稿において用いるモデルの生成結果を示す．今回は Qualcomm 社の MSM8960 チップセット 7)を搭載した Android OS の稼働する評価端末を対象にモデルを生成した．モデル化の対象とするコンポーネントと，パラメータを含むモデル式 Table 1 に示す．

Table 1 コンポーネント毎のモデル式

コンポーネント	モデル式
CPU	$c_1 f_{core1} u_{core1} + c_2 f_{core2} u_{core2}$
Display	$c_3 p_{brightness}$
3G	$\left(c_4 + c_5 p_{sendBps} + c_6 p_{rcvBps}\right) p_{pch} + c_7 p_{fach} + c_8 p_{pch} + c_9 p_{idle}$
LTE	$\left(c_{10} + c_{11} p_{sendBps} + c_{12} p_{rcvBps}\right) p_{LTEconnected} + c_{13} p_{LTEcdrx} + c_{14} p_{LTEidle}$
Wi-Fi	$\left(c_{15} + c_{16} p_{sendBps} + c_{17} p_{rcvBps}\right) p_{flagWiFi}$
GPS	$c_{18} p_{gps}$
Disk	$c_{19} p_{readByte} + c_{20} p_{writeByte}$
GPU	$c_{21} p_{gpuLoad}$

Table 2 パラメータ係数

i	パラメータ係数	i	パラメータ係数
0	1.462e-01	11	8.000e-08
1	1.375e-09	12	2.048e-08
2	1.162e-07	13	3.892e-02
3	2.340e-04	14	1.723e-02
4	1.563e-01	15	3.210e-02
5	3.265e-07	16	1.633e-07
6	1.107e-07	17	1.218e-07
7	1.179e-01	18	3.712e-02
8	1.384e-02	19	3.307e-10
9	1.402e-02	20	1.336e-09
10	2.054e-01	21	7.797e-10

5. アプリ消費電力可視化ツール

前記電力モデルを搭載したアプリ消費電力可視化ツールについて述べる．

本ツールの機能構成を Fig. 4 に示す．本ツールは，端末側で電力評価に必要なログを収集するログ収集アプリと，PC 側で前記ログを解析し，アプリ実行時の電力推定及び表示などを行うデータ分析用ツールから構成される．

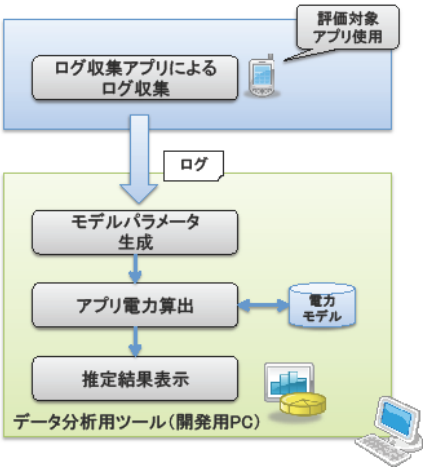

Fig. 4 アプリ消費電力可視化ツールの機能構成

ログ収集アプリは，端末上で評価対象のアプリが動作する間，電力モデルのパラメータ値生成に必要なログを 1 秒毎に収集する．データ分析用ツールは，特定の端末機種に対してあらかじめ作成した電力モデルを備えており，ログ収集アプリから得たログから生成したモデルパラメータから，その端末の 1 秒毎の平均電力を算出する．本ツールにより，アプリ利用による消費電力を，6.1 に後述するような時系列データとして可視化することができる．また，ハードウェアコンポーネント毎の内訳も可視化できることにより，何が原因で電力消費されているか，アプリ仕様や動作と対応づけて開発者が分析しやすくなると考える．

6. 評価

2 章に掲げた要件に対してそれぞれ評価を行った．要件[1]に対して電力推定精度を示す．要件[2]に対してログ収集にかかるオーバヘッドの影響を確認する．所定の評価シナリオに従い，端末上でアプリが動作する際の電力を測定器により実測し，推定結果と比較することで電力推定精度を評価する．ログ収集にかかるオーバヘッドは，ログ収集アプリによるログ収集を行った際の CPU 稼働率を測定し，ログ収集なしの場合と比較した増分を評価する．

6.1 電力推定精度の評価

6.1.1 評価方法

電力推定精度の評価方法について述べる．評価用アプリを用い，各々以下のような操作によるアプリ利用を評価シナリオとする．

＜メールアプリ＞
アプリ起動→新規メール作成→電話帳より宛先選択→メール件名・本文の入力→送信→アプリ終了
＜地図アプリ＞
アプリ起動→現在地取得→地図表示・閲覧→アプリ終了
＜カレンダーアプリ＞
アプリ起動→スケジュール新規作成→件名など入力→登録→アプリ終了
＜動画プレーヤアプリ＞
アプリ起動→コンテンツ一覧から動画選択→動画再生→再生終了後，アプリ終了
＜電話帳アプリ＞
アプリ起動→新規登録→氏名，電話番号など入力→登録→アプリ終了

上記のシナリオに従ったアプリ利用中に，本ツールのログ収集アプリを動作させログ収集を行う．また，同時に端末全体の消費電力を実測する．なお，測定器は Agilent 社製 N6705B を用いて，測定器からの DC 電源出力にて端末を動作させている．

推定精度は，ツールによる推定値と測定器による実測値との比較により評価する．

6.1.2 評価結果

本評価における測定結果として，メールアプリ利用における時系列に推定/実測結果を示したものを Fig. 5 に，各アプリの測定期間での平均誤差を Fig. 6 に示す．

Fig. 5 に示す通り，アプリ使用区間全体を通して，実測値の変動に対応付けて電力を推定できている．なお，平均誤差は約 10%である．また，その他のアプリについても，Fig. 6 に示す通り，全体的に 10%前後程度の誤差で推定できていることから，本ツールは一定の精度でアプリ電力の推定が可能であると確認できた．

一方，Fig. 5 の 20 秒付近，40～90 秒，110 秒付近では，最大で約 0.1A 程度の誤差が出ている．これは，ディスプレイ電力の推定誤差であると考えられる．評価端末は有機 EL ディスプレイを搭載しているが，今回のディスプレイ電力モデルは，元々LCD ディスプレイを対象に，Android OS 上で規定されているディスプレイ輝度値のみをパラメータにしており，有機 EL の特徴をパラメータ化したモデル生成を行っていない．従来研究 8)によれば，有機 EL ディスプレイの消費電力は画素毎の表示色に依存し，RGB 値により説明できるとされている．前述の区間では，ユーザ操作を伴うメールの宛先検索や文字入力など多くの画面遷移を伴い，様々な色調の変化が起こっていることが誤差要因であると考えられる．有機 EL ディスプレイ対応については今後の課題とする．

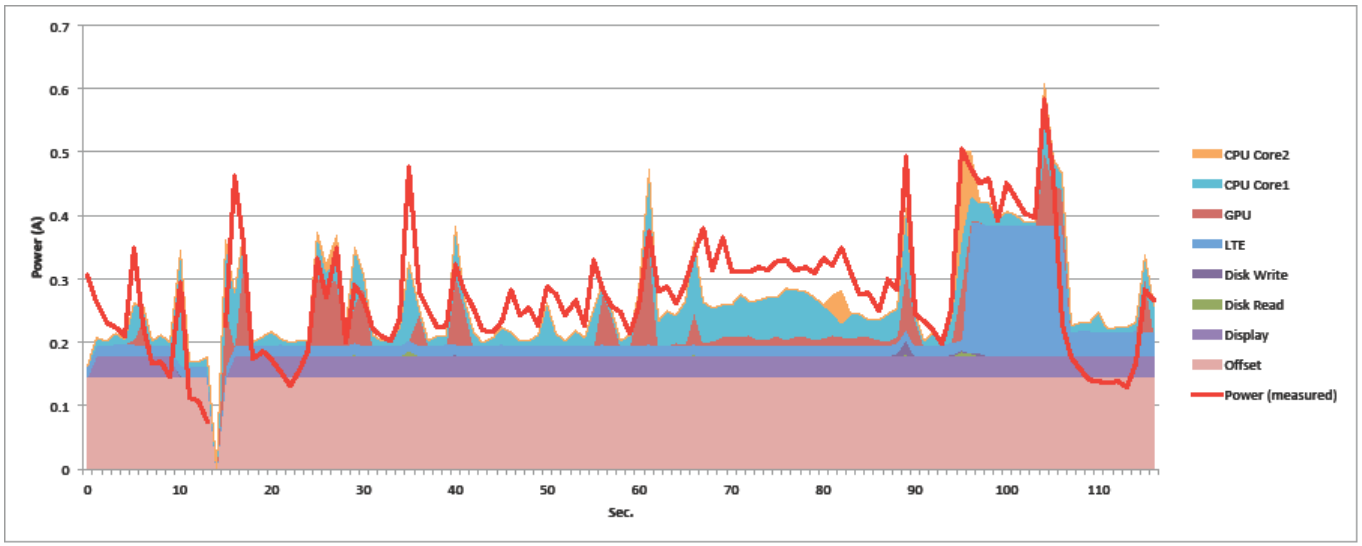

Fig. 5 メールアプリ使用時の消費電力推定値と実測値

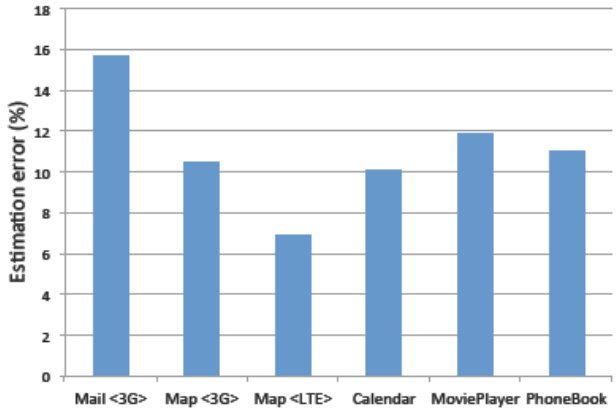

Fig. 6 アプリ毎の推定誤差

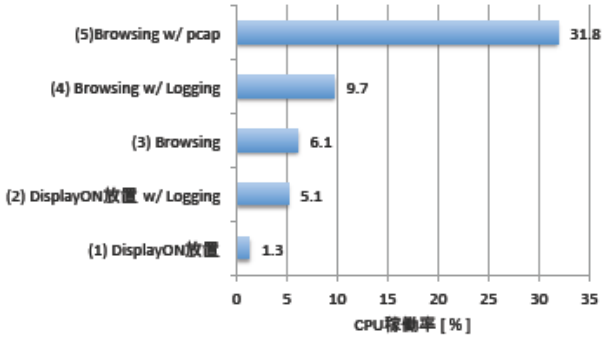

Fig. 7 ログ収集によるオーバヘッド比較

6.2 ログ収集オーバヘッドの評価

本ツールのログ収集アプリのログ収集に伴うオーバヘッド評価について述べる．本提案手法においては，前述の通り，アプリの実利用を妨げとならないようログ収集の負荷が低いことを要件[2]としている．また，3G および LTE の無線インタフェースの消費電力推定精度を高めるため，本提案手法は ARO で提案されている無線インタフェースの電力モデルの考え方を踏襲しているが，前述の要件を満たすため ARO よりもログ収集の負荷が小さい方式である点が特徴の一つである．以上より，本項では，ARO との比較により，本提案手法による前述のオーバヘッド低減の効果を示す．

ログ収集ログ収集アプリは，評価対象アプリの動作中，CPU 稼働率などモデルパラメータの生成に必要なログを 1 秒毎に収集する．

まず，このログ収集に伴うオーバヘッド評価を行うため，以下の試験パターンにて端末全体の CPU 稼働率の測定を行った．

（1） ログ収集動作無しに DisplayON、操作なしにて放置
（2） ログ収集動作ありに DisplayON、操作なしにて放置

測定結果は, Fig. 7 の通り, (1) ログ収集無において 1.3%, (2) ログ収集有において 5.1%となり，ログ収集に要するオーバヘッドは 3.8%程度であることが示された．

次に，ARO におけるログ収集と比較し，本手法のログ収集に伴うオーバヘッドが低いことを示す．以下（3）～（5）の試験パターンで CPU 稼働率の測定を行った．

なお, ARO で用いる収集ログはアプリ動作時のパケットキャプチャデータ（pcap ファイル）であり，測定時に用いる試験用アプリはデータ送受信を伴う必要があるため，複数の Web ページを順次自動でダウンロード・表示する簡易ブラウザを試験用アプリとして用いた．

（3） 簡易ブラウザのみ（ログ収集無）
（4） 簡易ブラウザ（提案手法のログ収集有）
（5） 簡易ブラウザ（パケットキャプチャ有）

測定結果を Fig. 7（3）～（5）に示す．前述と同様に，（3）と（4）との比較により，提案手法のログ収集オーバヘッドは 3.6%程度の増加となる．一方，ARO を想定したパケットキャプチャ収集では，（3）と（5）の比較により，25.7%の CPU 稼働率増加が観測された．

4 章に示したように, CPU 稼働率は CPU 電力のモデルパラメータであり，ログ収集処理による稼働率の増加分に比例して端末で電力が消費されることになるため，パケットキャプチャにより収集したログを用いた消費電力の推定結果は, 実際のアプリ利用による端末の消費電力と乖離する．本評価の測定によって乖離の一例が示された．パケットキャプチャ取得に要する処理負荷は，アプリや使い方によるトラフィックパターンや量に依存して増減するため，その補正は単純ではない．さらに，スマートフォン向けアプリはデータ送受信を伴うものが多く，パケットキャプチャ動作による消費電力の増加は一般的なアプリ利用の妨げになる可能性もある．

一方，本提案手法は，一定の周期で所定のログを収集する仕組みを用いているため，アプリの種別や利用シナリオを問わず，一定の低いオーバヘッドでログ収集が可能である．また，本提案手法では，パケットキャプチャを用いずに 3G/LTE の RRC 状態遷移を推定する仕組みを実現したことで，ログ収集のオーバヘッドを抑えつつも，ARO と同様に無線電力の推定精度を向上させることが可能になった．

7. おわりに

本稿では, 実際のアプリ使用において容易に利用可能なアプリ消費電力の評価を実現するため，電力モデルに基づく消費電力の推定によるアプリ消費電力の評価手法およびツールを提案した．モデルに基づく電力推定手法において，昨今の端末のハードウェア構成に対応すべく，マルチコア CPU のコア数および周波数の変動や，3G/LTE の RRC State を考慮するようモデルを拡張した．電力推定精度を評価した結果，一般的なアプリで 10%前後程度の誤差で推定できることが示された．さらに，従来手法（ARO）と比較し，3.8%程度のオーバヘッドでかつ特権の必要としない RRC State 遷移の推定手段を用いたことで，一定の精度を保ちつつ，特別な制約なく利用可能なアプリ消費電力の評価手段を実現することができた．今後の課題は，前述したように有機 EL ディスプレイへの対応など，推定誤差の要因に対してさらなるモデル拡張を行う予定である．

謝辞

本稿の執筆にあたり，ご助言をいただきました京都大学石原准教授，高瀬助教に心から感謝申し上げます．

参考文献

1) 古庄裕貴, 久住憲嗣, 神山剛, 稲村浩, 中西恒夫, 福田晃 : Android アプリケーションの運用時消費電力分析，信学会　ソフトウェアサイエンス（SS）研究会，　Vol.112, No.373, SS2012-58，pp.73-78, 2013-01-10-11
2) D. Lee, T. Ishihara, M. Muroyama, H. Yasuura and F. Fallah, “An Energy Characterization Framework for Software-Based Embedded Systems”, Proc. of the 2006 IEEE/ACM/IFIP Workshop on EStIMedia, pp.59-64, Oct. 2006.
3) 石原 亨，奥平 拓見，久住 憲嗣，神山 剛，関根 和寿，片桐 雅二，”OS から解析可能な無線通信端末の消費電力モデルとその生成手法”，信学技報，CPSY2008-92，Mar.2009.
4) Y. Kaneda, T. Okuhira, T. Ishihara, K. Hisazumi, T. Kamiyama and M. Katagiri, “A Run-Time Power Analysis Method using OS-Observable Parameters for Mobile Terminals”, Proc. of Int’l Conference on Embedded Systems and Intelligent Technology, pp.39-44, Feb.2010.
5) F. Qian, Z. Wang, A. Gerber, Z. Morley Mao, S. Sen and O. Spatscheck, “Profiling Resource Usage for Mobile Applications: A Cross-layer Approach”, Proc. of MobiSys 2011, pp.321-334, 2011.
6) 3G Specifications, 3GPP Website, <http://www.3gpp.org/3GPP-specifications>
7) Qualcomm Inc.，<http://www.qualcomm.com/snapdragon>
8) M. Dong and L.Zhong, “Chameleon: A Color-Adaptive Web Browser for Mobile OLED Displays”, Proc. of MobiSys 2011, pp.85-98, 2011.
9)