情報処理学会論文誌 Vol.55 No.8 1866–1875 (Aug. 2014)

推薦論文

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,a)] 稲村 浩[1] 太田 賢[1]

受付日 2013年12月19日, 採録日 2014年5月17日

概要：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．本論文では，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．

キーワード：デバッグ支援，プロファイリング，モバイル，Android OS，スマートフォン

Evaluation of a Model-based Energy Profiler for Android Applications

Takeshi Kamiyama[1,a)] Hiroshi Inamura[1] Ken Ohta[1]

Received: December 19, 2013, Accepted: May 17, 2014

Abstract: This paper describes a model-based energy profiler for Android applications that allows developers to test the energy-efficiency of their applications in real user environments. The profiler offers analysis of the energy consumption using a system-wide power model generated for each type of device. The model, generated by regression analysis, can determine the relationship between power consumption and each behavior of the hardware components. In this paper, we consider two profiler requirements, 1) accurate modeling of up-to-date devices and 2) lightweight online logging on devices for the collection of data. We extend our previous power model to account for the characteristics of multi-core CPUs and 3G/LTE and implement the profiler. Experiments show that it estimates energy consumption with about 10% error in a mixture of applications, while logging incurs a CPU time overhead of only 3.8%, which is superior to other profilers.

Keywords: debugging support, profiling, mobile computing, Android OS, smartphone

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ともない，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 など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．したがって，たと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時間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や電池持ち時間への影響を開発者にフィードバックし，実現される機能そのものや実装の品質に対するコストとして消費電力を意識してもらうことが有効であると考えられる．

[1] 株式会社 NTT ドコモ先進技術研究所
Research Laboratories, NTT DOCOMO, Inc., Yokosuka, Kanagawa 239–8536, Japan
[a)] kamiyamata@nttdocomo.com

本論文の内容は 2013 年 7 月のマルチメディア，分散，協調とモバイル（DICOMO2013）シンポジウム 2013 にて報告され，モバイルコンピューティングとユビキタス通信研究会主査により情報処理学会論文誌ジャーナルへの掲載が推薦された論文である．

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，Android OS でのアプリ消費電力の評価手段を開発者が容易に利用可能な形態で提供することを目的とする．具体的には，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良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，測定器などの機材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ことは，必要な機材の手配による時間や費用が開発コストを押し上げる，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り開発者の利便性は高まるであろう．

本論文の構成を以下に示す．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，3 章では関連研究について議論し本研究の位置付けを明らかにする．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，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，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を評価し，7 章でまとめと今後の課題を述べる．

2. 課題と要件

自然な利用状況におけるそのアプリ消費電力の評価手段をソフトウェアにて実現する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消費電力推定手段を備えること．

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．

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，すべ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 [2] の提案する電力モデルを採用し，スマートフォンのハードウェアコンポーネントごと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高い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ごと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機種ごとの電力モデル作成時のみであり，個別のアプリ・利用シナリオごとの実測は不要である．

上記をふまえ，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．モデルが妥当であっても実アプリの評価時の測定動作がハードウェアコンポーネントの稼動を増やすことは好ましくない．

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 [3] 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，対象が CPU など特定のコンポーネントに限定されていること，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ることから，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 [2]，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，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 RRC（Radio Resource Control）と呼ばれる機構がある [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良く推定している．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．

Yoon らは，実際の端末を構成する主要なコンポーネントを幅広くカバーした精度の高いモデル [7] と，そのモデルに基づくアプリ消費電力の推定手法を評価する手法を提案している [8]．しかし，消費電力の推定に必要なパラメータを得るために，システムコールを記録するなどカーネルレベルでのログ取得をともなう手法であるため，カーネルの改変が前提になる．カーネルの改変やシステムコールの記録はセキュリティの観点から市販の端末では通常は想定されておらず，本研究ではこの前提を受け入れることはできない．

以上より，既存研究では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・要件すべてを満たすことはできない．本研究は，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 [2] 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ごと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ごとの電力は，コンポーネント i ごと

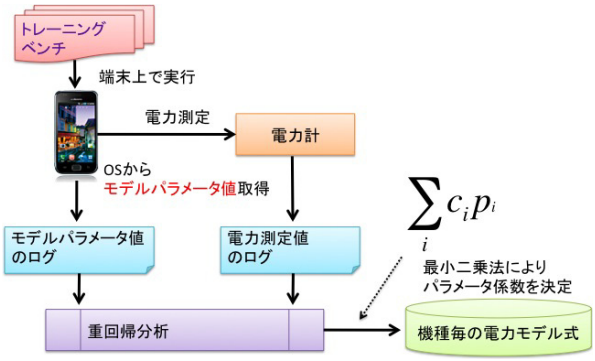


図 1 電力モデルの生成方法
Fig. 1 The process of power characterization.

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．c_{offset} は，モデル式より，コンポーネントの稼動状態に依存しない端末消費電力の基準値であり，端末が Deep Sleep 状態 [9] ではないが，ディスプレイが消灯し，CPU などのその他のコンポーネントが idle 状態であるときに必ず消費する電力と見なすことができる．

モデルの生成方法を図 1 に示す．モデルの生成は従来手法と同様，各コンポーネントを様々な負荷レベルの下で稼動させるように構成され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．

サンプルデータ取得のためのトレーニングベンチは，以下のとおり動作するよう作成した．

CPU ベンチ

1 スレッドで一定の浮動小数点演算を周期的に繰り返す処理．周期はループ処理のうちに，シナリオで指定した期間の Sleep 処理を含めることで，一定の CPU 負荷状態を維持する．また，マルチコア CPU の負荷状態を幅広くサンプル取得するため，任意の数だけ前記スレッド処理を複数同時に実行できるようシナリオ指定できるようにした．

無線通信ベンチ

任意サイズのデータ送信または受信を繰り返す処理．前述の CPU ベンチと同様，シナリオでデータサイズや Sleep 期間を指定することで，一定のスループットを維持しつつ様々なパターンのサンプルを取得する．本ベンチは，3G/LTE や Wi-Fi 無線通信インタフェースにすべてに対して用いる．

ディスプレイベンチ

Android システム上定義されているディスプレイ輝度を任意の値に維持した状態で画面表示を行う処理．

GPS ベンチ

シナリオで指定した期間，GPS を測位状態にする処理．

ディスクベンチ

SD カードなど所定のストレージ領域に対し，任意サイ

ズのデータを読み込みまたは書き込みを繰り返す処理．

GPU ベンチ

シナリオで指定された個数のキューブを指定期間継続して描画する処理．描画処理には OpenGL ES 2.0 [10] を使用．

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 [2] 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．本論文では，拡張箇所のみ 4.2 節に後述し，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 節に後述する．

4.2 モデル拡張

4.2.1 マルチコア CPU のモデル化

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，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，コア数や周波数の切替えも考慮したモデル生成を行う．マルチコア CPU のモデル化の考え方としては，従来研究 [11] において，コアやキャッシュなど CPU を構成するコンポーネントごとに独立にモデル化し，それらを加算する形で定式化する方法が示されている．本提案手法においても，同様の考え方を踏襲してモデル化を行う．

本提案手法における CPU の電力 P_{cpu} を以下に示す．

$$\begin{aligned} P_{cpu} &= P_{core1} + P_{core2} + \cdots + P_{coreN} \\ &= \sum_{i}^{N} c_i p_i = \sum_{i}^{N} c_i f_i u_i \end{aligned}$$

ここで，$P_{core1} \sim P_{coreN}$ はそれぞれ CPU を構成するコアごとの電力であり，P_{cpu} はコアごとの電力を加算したものである．

近年の周波数制御はコアごとになされることから各々独立にモデル化する．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，周波数ごと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．

本提案手法における CPU 電力モデルのパラメータの妥当性を確認するため，4.3 節に後述する端末と 4.1 節に前述したトレーニングベンチを用いて予備検証を行った．トレーニングベンチは，様々な負荷レベルで CPU を稼動させるよう動作し，その動作パターンごと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ごとの消費電力とパラメータとの関係を**図 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．

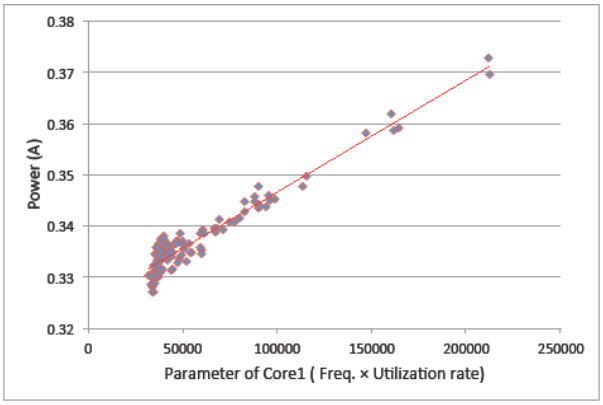


図 2 消費電力と CPU パラメータの関係
Fig. 2 The relation between power consumption and CPU parameter.

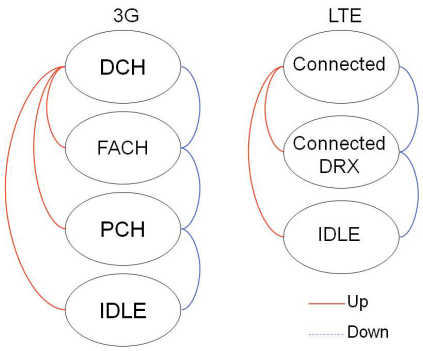


図 3 3G/LTE の RRC State 遷移の例
Fig. 3 3G/LTE RRC State transition.

4.2.2 モバイル無線インタフェースのモデル化

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．

3G/LTE における RRC State 遷移の概略図を**図 3** に示す．たとえば 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

表 1 RRC 推定精度と推定遅延の検証結果
Table 1 Experimental result of RRC estimation accuracy and delay.

Time overlap	Estimation delay of state transition (sec.)			
3G	*to DCH*	*to FACH*	*to PCH*	*to IDLE*
71.3%	0.425	1.679	0.926	71.681
LTE	*to Connected*	*to CDRX*	*to IDLE*	
96.4%	0.532	1.654	1.826	

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 ごと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良く推定することが可能である．加えて，端末上で，直接 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，取得に特権が必要な手段でありセキュリティ上の理由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 Android OS 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を抑えられることから，ログ収集オーバヘッドを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．

本提案手法における推定誤差の影響を確認するため，前述の 1 秒周期でのログ取得による RRC State 推定の精度と推定遅延の予備検証を行った．検証実験では，端末上でランダムにデータ送受信を発生させる試験用プログラムを動作させた状態で，本提案手法による RRC State 推定を行うと同時に，QxDM [12] と呼ばれる試験用ツールを用いて実際の RRC State 遷移ログを取得した．推定結果と QxDM による測定結果との比較により，推定精度と推定遅延の評価した結果を**表 1** に示す．推定精度は，試験プログラムの動作期間のうち，正しく RRC State が推定されていた期間が占める割合とする．LTE の場合，96.4%と精度良く推定できていることが示された．一方，3G の場合は，PCH から IDLE への遷移において大きな推定遅延が生じたことにより，推定精度は 71.3%となった．この推定遅延に対する改善は今度の課題とするが，1）PCH から IDLE 以外の遷移では大きな遅延なく推定できていること，2）PCH および IDLE 状態における実際の消費電力は非常に小さく [13]，両者に大きな差が見られないことから，消費電力の推定精度には影響がないと考えられる．

4.3 モデル生成結果

本論文において用いるモデルの生成結果を示す．Android OS が稼働するスマートフォンのうち，Qualcomm 社の MSM8960 チップセット [14] を搭載した評価端末 A と，Texas Instruments 社の OMAP4460 チップセット [15] を搭載した評価端末 B の 2 機種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を**表 2** に示す．端末全体の消費電力モデルは，コンポーネントごとの消費電力をモデル化し，線形結合したものになる．モデル生成の結果得られたパラメータ係数と，各パラメータに多重共線性の有無を確認するための指標である VIF（Variance Inflation Factor）を**表 3** に示す．一般的に，VIF が 10 以上の値をとると多重共線性の可能性が高いとされているが，表 3 のとおり，すべての係数に係る VIF が 1〜3 程度と十分に低い結果となった．重回帰分析によるモデルの当てはまりの良さを示す指標として調整済み R 二乗があるが，評価端末 A，B ともにそれぞれ 0.9664，0.9875 と高く，モデルの有意性が確認できた．評価端末 B のモバイル無線インタフェースは LTE 非対応のため該当する係数（$i = 10$〜14）は生成していない．

表 2 コンポーネントごとのモデル式
Table 2 Power model of target device.

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	$(c_4 + c_5 p_{sendBps} + c_6 p_{rcvBps}) p_{Dch} + c_7 p_{fach} + c_8 p_{pch} + c_9 p_{idle}$
LTE	$(c_{10} + c_{11} p_{sendBps} + c_{12} p_{rcvBps}) p_{LTEconnected} + c_{13} p_{LTEcdrx} + c_{14} p_{LTEidle}$
Wi-Fi	$(c_{15} + c_{16} p_{sendBps} + c_{17} p_{rcvBps}) p_{flagWiFi}$
GPS	$c_{18} p_{gps}$
Disk	$c_{19} p_{readByte} + c_{20} p_{writeByte}$
GPU	$c_{21} p_{gpuLoad}$

表 3 パラメータ係数
Table 3 Values of Coefficients.

	評価端末 A		評価端末 B	
i	*Ci*	VIF	*Ci*	VIF
0	1.462e-01		1.517e-01	
1	1.375e-07	3.028175	2.482e-07	2.204509
2	1.162e-07	2.938385	1.642e-07	1.741155
3	2.340e-04	1.632474	6.380e-04	1.107095
4	1.563e-01	2.425848	1.531e-01	2.945161
5	3.265e-07	1.716143	5.328e-07	2.216406
6	1.107e-07	1.384222	1.020e-07	1.421324
7	1.179e-01	1.074541	7.726e-02	1.170108
8	1.384e-02	1.074810	2.630e-02	1.005708
9	1.402e-02	1.074463	2.343e-02	1.005638
10	2.054e-01	1.947452		
11	8.000e-08	1.411644		
12	2.048e-08	1.296661		
13	3.892e-02	1.074483		
14	1.723e-02	1.074530		
15	3.210e-02	2.043827	4.648e-02	2.066004
16	1.633e-07	1.410098	9.862e-08	1.449676
17	1.218e-07	1.410647	6.724e-08	1.438688
18	3.712e-02	1.018166	6.119e-02	1.011056
19	3.307e-10	1.061168	6.669e-10	1.043131
20	1.336e-09	1.039160	3.315e-09	1.032754
21	7.797e-10	1.057354	1.515e-01	1.034463

本研究においてモデル化したコンポーネントは，近年の様々なアプリにおいて利用度が高いことと，アプリ動作時に端末全体の消費電力に占める割合が大きいコンポーネントを優先的にモデル化の対象とした．各コンポーネントに対して，検討したパラメータが連続値をとり，電力値との関係に線形性がある場合，あるいはパラメータ自体がコンポーネントの状態を示す値で，各状態に対応した電力値が定まる場合には，同じモデル生成の考え方，方法のもと，新たなコンポーネントを追加したモデル拡張も可能であると考える．たとえばカメラなど，既知の未対応コンポーネントに対しては，モデル生成方法との適合性確認も含め，今後の課題としてモデル拡張を検討する予定である．

一方，本研究においては，アプリから OS にアクセスすることで容易に取得可能なコンポーネントの稼働量を利用することを要件としているため，6.1.2 項で議論するような，OS がアプリに挙動を観測する手段を提供しないコンポーネントについてはモデルとして扱えない．さらに OS が提供する稼働量が実際のコンポーネント稼働を反映しない場合もモデルとして扱えない．たとえば CPU 仮想化により同一端末内に複数の OS が実行される環境には対応できない [16]．

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図 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および表示などを行うデータ分析用ツールから構成される．

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 秒ごとに収集する．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

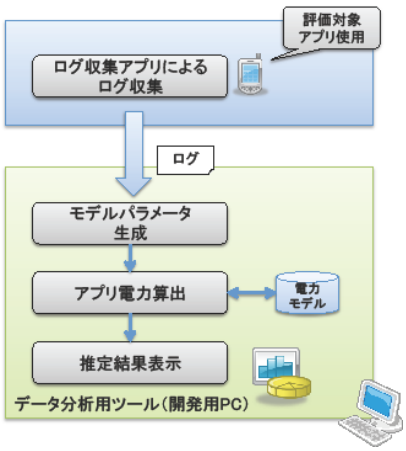


図 4 アプリ消費電力可視化ツールの機能構成
Fig. 4 Overview – Functional design of energy profiler.

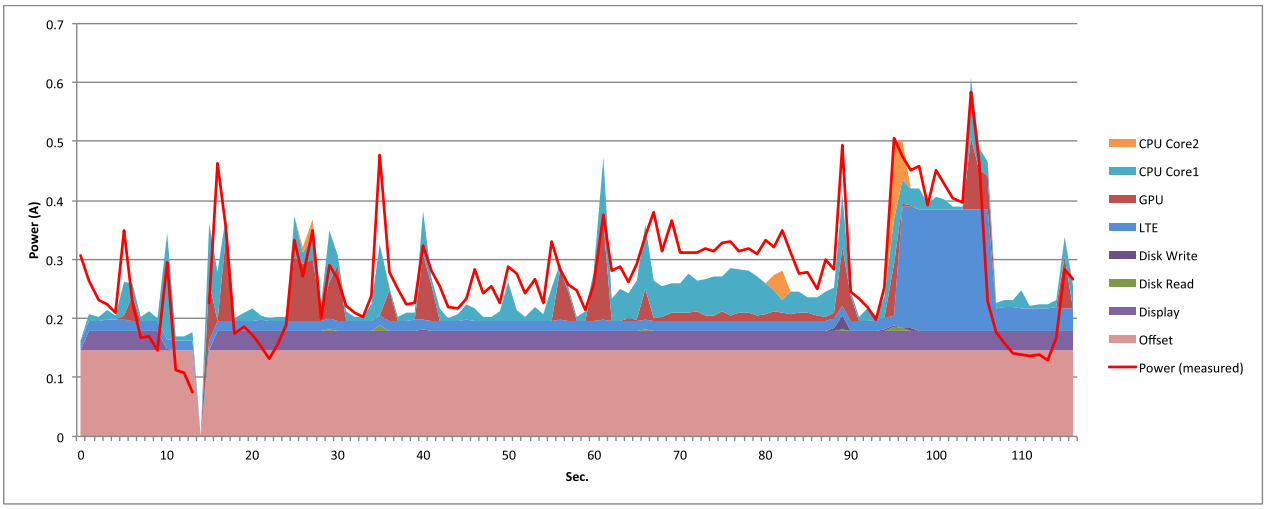


図 5 メールアプリ使用時の消費電力推定値および実測値
Fig. 5 Estimated and measured power consumption for the mail application scenario.

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 秒ごと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，**図 5** 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．ハードウェアコンポーネントごと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．評価用アプリには，実ユーザが一般的に利用するアプリ（メール，地図，カレンダー，動画プレーヤ，電話帳）と，CPU/GPU バウンドなベンチマークアプリとして AnTuTu [17] 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

AnTuTu CPU 負荷ベンチ

アプリ起動 →CPU 負荷ベンチ選択・実行 → アプリ終了

AnTuTu GPU 負荷ベンチ

アプリ起動 →GPU 負荷ベンチ選択・実行 → 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い，同時に端末全体の消費電力を実測する．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．各シナリオの実施にあたっては，端末のディスプレイを消灯し，端末をきわめて消費電力の低い Deep Sleep 状態にした後，シナリオを開始する．時系列に精度評価を行うためには，両データ間でシナリオの開始タイミングを同期させる必要があるが，端末・測定器どうしの時刻同期ができなかったため，代わりに，前述の操作の結果，データから観測される消費電力の下降を開始タイミングと見なすようにした．（図 5 の 14 秒目付近）

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図 5 に，各アプリの測定期間での平均誤差を**図 6** に，各アプリの測定期間での消費エネルギーを**図 7** に示す．

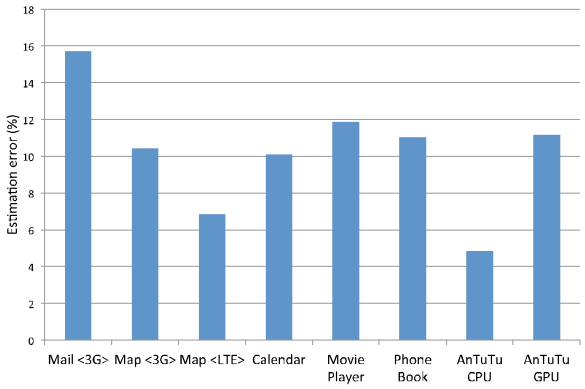


図 6 アプリごとの電力推定誤差
Fig. 6 Power estimation error in each scenario.

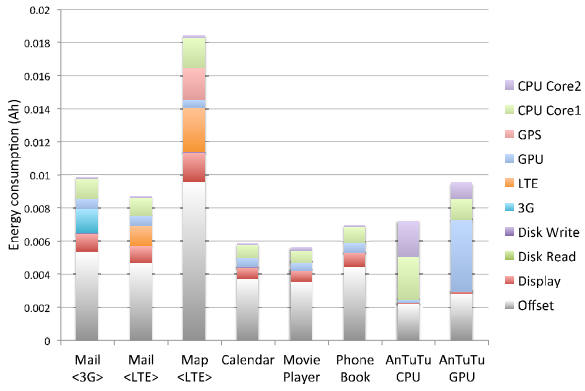


図 7 アプリごとの消費エネルギー推定値
Fig. 7 Energy consumption estimated in each scenario.

図 5 に示すとおり，アプリ使用区間全体をとおして，実測値の変動に対応付けて電力を推定できており，平均誤差は約 10%である．他のアプリについても，図 6 に示すとおり，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，図 5 の 20 秒付近，40〜90 秒，110 秒付近では，最大で約 0.1 A 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 [18] によれば，有機 EL ディスプレイの消費電力は画素ごとの表示色に依存し，RGB 値により説明できるとされている．前述の区間では，ユーザ操作をともなうメールの宛先検索や文字入力など多くの画面遷移をともない，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．

図 5 に示す評価では，全体の消費エネルギーに対して Offset 分が大きく占める結果となった．4.1 節に前述したとおり，Offset 分は端末が Deep Sleep していない Idle 状態において必ず消費してしまう電力であるが，それゆえに，アプリの改善により実行時間を短縮させることも，効果的に省電力化を図る一事例であることが分かる [19]．特に，ユーザに見えない周期的なバックグラウンドタスクの実行をともなうアプリに対する評価観点の 1 つとして，本ツールの使用が有効であると考える．

一方で，Offset 分は重回帰分析の結果得られた端末消費電力の静的成分に過ぎず，ツールを用いて Offset 分に対する詳細な分析を行うことはできない．Offset 分は，1）上述のとおり CPU を含むチップセット全体での待機電力であること，2）システムが制御しアプリから挙動が観測できないようなコンポーネントの動作，たとえば加速度や照度などの何らかのセンサが OS から観測不能な形で動作することがあった場合には，これらのコンポーネントの動作分の消費電力も含むことになる．したがって Offset は機種ごとのモデルの予測誤差となり得る．上記 2）に該当するコンポーネントの ON/OFF などの制御手段なり動作条件が与えられていればモデル化における精度低下を回避できると考える．

6.2 ログ収集におけるオーバヘッドの評価

本ツールのログ収集アプリにおけるログ収集動作によるオーバヘッドを評価する．本提案手法においては前述のとおり，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．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 1 つである．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 秒ごとに収集する．したがって CPU 稼働率が本来のアプリ由来のものより大きくなることが課題である．

まず，このログ収集にともな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) ログ収集動作なしに 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，**図 8** のとおり，(1) ログ収集なしにおいて 1.3%，(2) ログ収集ありにおいて 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ともな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．

ARO で用いる収集ログはアプリ動作時のパケットキャ

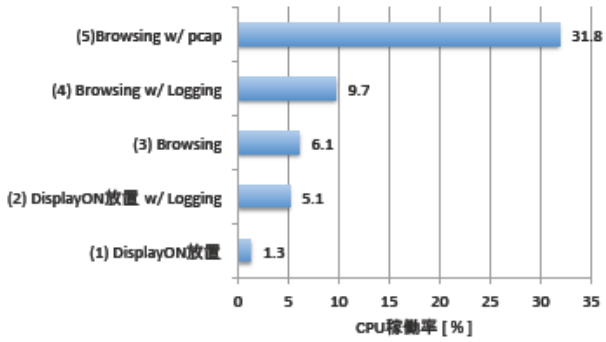


図 8 ログ収集によるオーバヘッド比較
Fig. 8 Comparison of CPU time overhead for logging.

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ともなう必要があるため，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（ログ収集なし）
(4) 簡易ブラウザ（提案手法のログ収集あり）
(5) 簡易ブラウザ（パケットキャプチャあり）

測定結果を図 8 (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ともなうものが多く，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，低いオーバヘッドでログ収集が可能である．さらに本提案手法では，パケットキャプチャを用いずに 3G/LTE の RRC State 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

本研究は，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 を考慮するよう既存研究に見られる電力モデルを拡張した．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%程度と小さなオーバヘッドかつ特権を必要としない 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，容易に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．

今後は新たなデバイスである有機 EL ディスプレイへの対応など，誤差の要因に対して推定するアプリ消費電力のさらなる高精度化を行う予定である．

謝辞 本論文の執筆にあたり，ご助言をいただきました京都大学石原 亨准教授，高瀬英希助教に心から感謝申し上げます．

参考文献

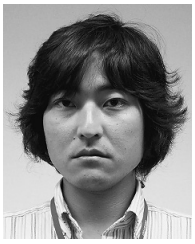
[1] 古庄裕貴，久住憲嗣，神山 剛，稲村 浩，中西恒夫，福田晃：Android アプリケーションの運用時消費電力分析，信学会ソフトウェアサイエンス（SS）研究会，Vol.112, No.373, SS2012-58, pp.73–78 (2013).
[2] 石原 亨，奥平拓見，久住憲嗣，神山 剛，関根和寿，片桐雅二：OS から解析可能な無線通信端末の消費電力モデルとその生成手法，信学技報，CPSY2008-92 (Mar. 2009).
[3] Lee, D., Ishihara, T., Muroyama, M., Yasuura, H. and Fallah, F.: An Energy Characterization Framework for Software-Based Embedded Systems, *Proc. 2006 IEEE/ACM/IFIP Workshop on EStIMedia*, pp.59–64 (Oct. 2006).
[4] Kaneda, Y., Okuhira, T., Ishihara, T., Hisazumi, K., Kamiyama, T. and Katagiri, M.: A Run-Time Power Analysis Method using OS-Observable Parameters for Mobile Terminals, *Proc. Int'l Conference on Embedded Systems and Intelligent Technology*, pp.39–44 (Feb. 2010).
[5] Qian, F., Wang, Z., Gerber, A., Mao, Z.M., Sen, S. and Spatscheck, O.: Profiling Resource Usage for Mobile Applications: A Cross-layer Approach, *Proc. MobiSys 2011*, pp.321–334 (2011).
[6] 3G Specifications, 3GPP Website (online), available from ⟨http://www.3gpp.org/3GPP-specifications⟩ (accessed 2014-04-10).
[7] Jung, W., Kang, C., Yoon, C., Kim, D. and Cha, H.: DevScope: A Nonintrusive and Online Power Analysis Tool for Smartphone Hardware Components, *Proc. 8th IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis* (*CODES+ISSS'12*), pp.353–362 (2012).
[8] Yoon, C., Kim, D., Jung, W., Kang, C. and Cha, H.: AppScope: Application Energy Metering Framework for Android Smartphone using Kernel Activity Monitoring, *Proc. 2012 USENIX Conference on Annual Technical Conference*, pp.36–36 (2012).
[9] ARM Information Center: Cortex-M3 テクニカルリファレンスマニュアル（オンライン），入手先 ⟨http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.ddi0337gj/Cihjbbge.html⟩（参照 2014-04-

14).
[10] Khronos Group: OpenGL (online), available from ⟨http://www.khronos.org/opengles/⟩ (accessed 2013-04-10).
[11] Robert Basmadjian and Hermann de Meer: Evaluating and modeling power consumption of multi-core processors, *Proc. 3rd International Conference on Future Energy Systems: Where Energy, Computing and Communication Meet* (*e-Energy '12*), pp.1–10 (2012).
[12] QxDM Professional: QUALCOMM eXtensible Diagnostic Monitor (online), available from ⟨http://www.qualcomm.com/solutions/testing/diagnostics-software⟩ (accessed 2014-04-10).
[13] Puttonen, J., Virtej, E., Keskitalo, I. and Malkamaki, E.: On LTE Performance Trade-off Between Connected and Idle States with Always-on Type Applications, *Proc. 2012 IEEE 23rd International Symposium on Personal, Indoor and Mobile Radio Communications* (*PIMRC*), pp.981–985 (2012).
[14] Qualcomm Inc.: Snapdragon (online), available from ⟨http://www.qualcomm.com/snapdragon⟩ (accessed 2014-04-10).
[15] Texas Instruments: OMAP4460 (online), available from ⟨http://www.ti.com/product/omap4460⟩ (accessed 2014-04-10).
[16] VMware and Trango: Mobile Virtualization Platform (MVP) (online), available from ⟨http://www.vmware.com/jp/company/acquisitions/trango⟩ (accessed 2014-04-14).
[17] Google: AnTuTu Benchmark (online), available from ⟨https://play.google.com/store/apps/details?id=com.antutu.ABenchMark&hl=ja⟩ (accessed 2014-04-10).
[18] Dong, M. and Zhong, L.: Chameleon: A Color-Adaptive Web Browser for Mobile OLED Displays, *Proc. MobiSys 2011*, pp.85–98 (2011).
[19] 小西哲平，稲村　浩，川崎仁嗣，神山　剛，大久保信三，太田　賢：画面オフ状態におけるバックグラウンドタスク同時実行による Android 端末の省電力化，情報処理学会論文誌，Vol.55, No.2, pp.587–597 (2014).

推薦文

アプリ開発時に消費電力を推定できるため，省電力を目的とした開発支援ツールとして高い有用性がみとめられる．よって，ここに研究会推薦論文として推薦する．

（モバイルコンピューティングとユビキタス通信研究会
主査　渡邊　晃）



神山 剛　（正会員）

NTT ドコモ先進技術研究所勤務．平成 18 年東京大学大学院新領域創成科学研究科修士課程修了．平成 16 年（株）イージス設立および代表取締役．平成 18 年より NTT ドコモ入社．モバイルコンピューティング，ソフトウェア省電力化，分散システムに関する研究に従事．

稲村 浩　（正会員）

NTT ドコモ先進技術研究所勤務．平成 2 年慶應義塾大学大学院理工学研究科修士課程修了．同年日本電信電話（株）入社．平成 6 年から 7 年にカーネギーメロン大学計算機科学科にて訪問研究員．平成 10 年より NTT ドコモ．モバイル環境におけるシステムソフトウェア，トランスポートプロトコルに関する研究開発に従事．電子情報通信学会，ACM，IEEE 各会員．博士（工学）．

太田 賢　（正会員）

NTT ドコモ先進技術研究所勤務．平成 10 年静岡大学大学院博士課程修了．博士（工学）．平成 11 年 NTT 移動通信網（株）入社．現在，NTT ドコモ先進技術研究所勤務．モバイルコンピューティング，端末セキュリティ，分散システムに関する研究に従事．訳書『コンピュータネットワーク第 5 版』等．電子情報通信学会会員．