Vol.2011-SE-172 No.4
Vol.2011-EMB-21 No.4
2011/5/17

電池駆動システムにおけるプログラミング言語が与える消費エネルギーへの影響

横　山　哲　郎[†1]

プログラミング言語の選択による情報通信端末の消費エネルギーの変化を定量的に評価する試みについて報告する．本稿では，アプリケーションの実装プログラミング言語ごとの消費電力への影響を，実情報端末において電力計で実測することにより確かめた．その結果，インタープリタ言語では消費電力の大きな揺れが観測されたが，コンパイル言語と同様の平均消費電力を示し，電池に与える影響の変化は限られた．われわれが実験を行った範囲では，プログラミング言語およびアプリケーションを変化させたときの 1 秒ごとの平均消費電力の変化は高々 20 % 未満であり，その影響は軽微であることが確かめられた．われわれの知る範囲では，プログラミング言語ごとに消費電力/消費エネルギーの比較を行った報告は限られており，本研究で得られた基礎データは低消費エネルギーアルゴリズムや低消費エネルギープログラミングの研究を行う上での参考データとして貴重である．

Programming Language Effects of Energy Consumption on Battery-Powered Systems

Tetsuo Yokoyama[†1]

The trial of finding the programming language effects of energy consumption on a battery-powered system is presented. By means of a digital power meter, we measured the energy consumption of an information device on which different applications implemented in various programming languages run. While the power consumption fluctuated substantially when an interpretive language run on the device, the average power consumption of an interpretive language was not much different from that of a compiler language. The experimental results showed that the effects of what programming languages to use were limited; The change of the power consumption was up to approximately 20 %. To the best of our knowledge the report on power/energy consumption of each programming language implementation has been limited and the result data is useful for studying the theoretical model of energy consumption and exploring energy efficient algorithms and programming.

1. はじめに

情報通信端末における計算資源の爆発的向上を背景として，大量の資源を消費する高機能なプログラミング言語からソフトウェア開発に用いるものを選択することが可能になってきている．一方，電池駆動システムである情報通信端末ではバッテリにおけるエネルギー密度の限界[3] から消費エネルギーを最適化することが重要視されている．われわれは，様々なプログラミング言語が情報端末における消費エネルギーの振舞いにどのような影響を与えるかを明らかにし，ソフトウェア開発におけるプログラミング言語選定のための指標として消費エネルギーを用いることを可能にすることを目指す．

プログラミング言語の実行方法は大きく分けて 2 つに分類される．実行効率が高い機械語にコンパイルで変換する方法と実行時に逐次的に翻訳を行いインタープリタ（解釈系）により実行する方法である．一般に後者では，インタープリタの方が簡易であるが実行効率が悪いことが知られている．情報通信端末の計算資源の爆発的向上により Ruby, Python, Perl といったいわゆるインタープリタ型のスクリプト言語が使われ始めている．しかし，これらの新しい言語が情報通信端末において消費エネルギーにどのような影響を与えるのかは明らかではない．したがって，厳しい消費エネルギー制約が課せられる情報通信端末において，新しく使い始められたスクリプト言語が実行時にどのような消費エネルギーの挙動を示すかについての識見が重要になってきている．

プログラムの計算複雑度として，理論・応用の両面から，古くから研究されているものには実行時間と使用メモリがある．これらは抽象機械を定義しその上で漸近的な振る舞いを議論することが出来る．これに対し消費エネルギーという計算複雑度が注目されだしたのは比較的最近である．さらに，消費エネルギーは実行時間から数乗の影響しか受けず，それほど大きくスケールしない．したがって，既存の計算理論よりもきめ細かな（定数項などを重要視した）研究が必要である．

エネルギーを消費するのはソフトウェアではなくハードウェアである．したがって，一義的には，実存するハードウェアの消費エネルギーを実際に計測することが実践的な研究を行う上で重要である．本研究では，机上のシミュレーションではなく，消費エネルギーの評価

†1 南山大学情報理工学部
Faculty of Information Sciences and Engineering, Nanzan University

Vol.2011-SE-172 No.4
Vol.2011-EMB-21 No.4
2011/5/17

は実際の情報端末とパワーメータにより行う．

本稿の概要は次の通りである．2 章では，他の研究との関連について述べる．3 章では，実験に用いたプログラミング言語，アプリケーション，ターゲット端末，測定環境について説明する．4 章では，実験における実測データとそこから得られた知見を提示する．最後に，5 章で本稿のまとめを述べる．

2. 関 連 研 究

1 次近似的には，消費電力 P を一定と考えることで，消費エネルギー E はアプリケーションの実行時間 t に比例する：

$$E = Pt \tag{1}$$

もしターゲット端末とその CPU のクロック周波数によって P が一意に定まるのであれば，消費エネルギーの予測には，常に式 1 を用いればよい．高 CPU 負荷量で，通信を伴わなず，ファイル IO が少ないアプリケーションをスマートフォンなどの情報通信端末で実行した場合は，各クロック周波数において P は一定値に近い．理論的研究の多くでは，この仮定が用いられている．例えば，デッドライン制約の下でのスケジューリング問題[5] や，消費エネルギー最適化を意識したデータ圧縮[2] に関連するセンサネットワークのエネルギー最適化問題[4),7] などでも用いられてきた．われわれも，整列アルゴリズムの消費エネルギー解析を実施時に本仮定を用いた[6]．実際には，アプリケーションおよびにプログラミング言語ごとに，消費エネルギーは変化する．低消費エネルギーアルゴリズム[1] や低消費エネルギープログラミング[8),9] の研究のためにはこういった基礎データを押さえておきたいが，われわれの知る範囲ではプログラミング言語ごとに消費電力/消費エネルギーを比較を行った信頼性のある報告は限られている．われわれは，こういった研究を進めるときの参照データとして，本稿の取り組みは重要である，と考える．

3. 実 験 環 境

評価を行うプログラミング言語を選定する．言語は大きくいくつかのパラダイムに分かれるが，オブジェクト指向型，関数型，論理型，命令型などから，広く使われている言語を優先的にそれぞれサンプリングしたい．しかし，ベンチマークプログラムやインタープリタが存在しない場合は，その言語を選択することはできない．今回は，プログラミング言語 C, Java, Ruby, PHP のみを対象とせざるを得なかった．ターゲット端末上でこれら以外の言語の動作を実現できなかったためである．C 言語は代表的なコンパイラ型・命令型言語であり，残り 3 者はオブジェクト指向言語である．Java プログラムは，中間言語であるバイトコードに実行前にコンパイルされ，Java 仮想マシンで解釈実行される．Ruby, PHP は，解釈系のプログラミング言語である．

表 1 ベンチマーク
Table 1 Benchmark.

Benchmark	Description
chameneos-redux	Symmetrical thread rendezvous requests
binary-trees	Allocate and deallocate many many binary trees
fannkuch-redux	Indexed-access to tiny integer-sequence
spectral-norm	Eigenvalue using the power method
n-body	Double-precision N-body simulation
regex-dna	Match DNA 8-mers and substitute nucleotides for IUB codes
thread-ring	Switch from thread to thread passing one token
mandelbrot	Generate Mandelbrot set portable bitmap file
reverse-complement	Read DNA sequences - write their reverse-complement
fasta	Generate and write random DNA sequences

表 2 Nokia N900 の仕様.
Table 2 Specification of Nokia N900.

Processor	ARM Cortex-A8 600 MHz
Clock frequency	{600,500,250}
Memory	256 MB RAM, 32 GB storage
WLAN	Wi-Fi 802.11 b/g
OS	Maemo 5

ベンチマークプログラムは，無償で公開されており自由に使用できるものから，複数のプログラミング言語で同一のアルゴリズムを実装したものを入手した．具体的には，`debian.org` の計算機言語ベンチマークゲーム[⋆1]から修正 BSD ライセンスの下で配布されている**表 1** のベンチマークをダウンロードした．このサイトでは，与えられた各問題の解答が様々なプログラミング言語で実装されたものが投稿され，それら特定実装の実計算機での実行時間などが競われている．したがって，各実装はそのプログラミング言語において，自然かつ効率的であることが期待される．われわれが利用したアプリケーションはすべて高 CPU 負荷量で，通信を伴わず，ファイル IO が少ないアプリケーションであった．

⋆1 http://shootout.alioth.debian.org/

Vol.2011-SE-172 No.4
Vol.2011-EMB-21 No.4
2011/5/17

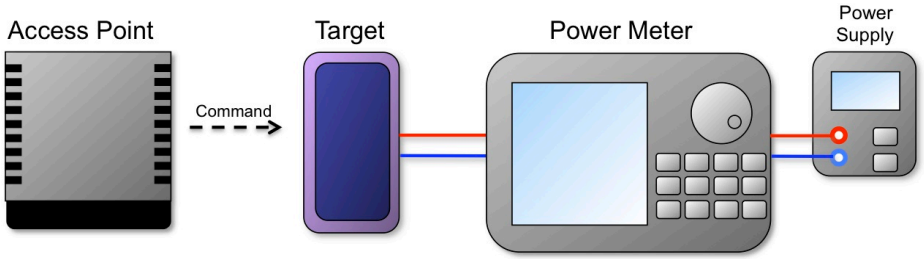

図 1 実験環境
Fig. 1 Experimental setup.

対象端末は，複数のインタープリタが動作する必要があった．本稿では，Linux を搭載した Nokia N900 を対象とした．本稿で利用した Nokia N900 の仕様を**表 2** に示した．プログラミング言語 C, Java, Ruby, PHP が動作するようにした．

実験環境を**図 1** のように整備した．リモートにある計算機からアクセスポイントを通して情報通信端末にコマンドを送信して実験を実施した．情報端末には，ダミーバッテリを装着して，パワーメータ，直流電源と接続した．情報通信端末の電気は直流安定化電源菊水 PMC-18-5A から 4.00 V で供給され, 電力測定器 WT1600 によってその電圧・電流が測定された. 電流の入力範囲を 0.00-1.00 A とした．本稿におけるすべての実験はこの範囲内に収まった．測定電力の確度は読み値誤差 0.1 %，測定レンジ誤差 0.2 % (10 mW) であった．データ更新レートは最速の 50 ms に設定した．なお，附属電池パックには供給電圧が 3.6 V と記載してあったが，情報通信端末はこの電圧では起動しなかった．このため, 完全充電された電池パックの実際の電圧 4 V に設定した．

スクリプトによってパワーメータと情報端末を自動制御して，消費エネルギーの実測を行った．各実験は，長時間連続して繰り返して算術平均をとった．

4. 実験結果

図 2 に，Java 言語で実装された fasta が Nokia N900 におけて消費する電力の推移の一部を示した．この図にプロットされたのは，1 回の試行の結果である．0.8 W で 50 ms ごとに 50 mW ほど上下に揺らいでいる区間と約 1.57 W まで上昇する約 100 ms ほどの区間が観測された．fasta には，バッファに DNA 配列を生成するフェーズとファイルに標準出力

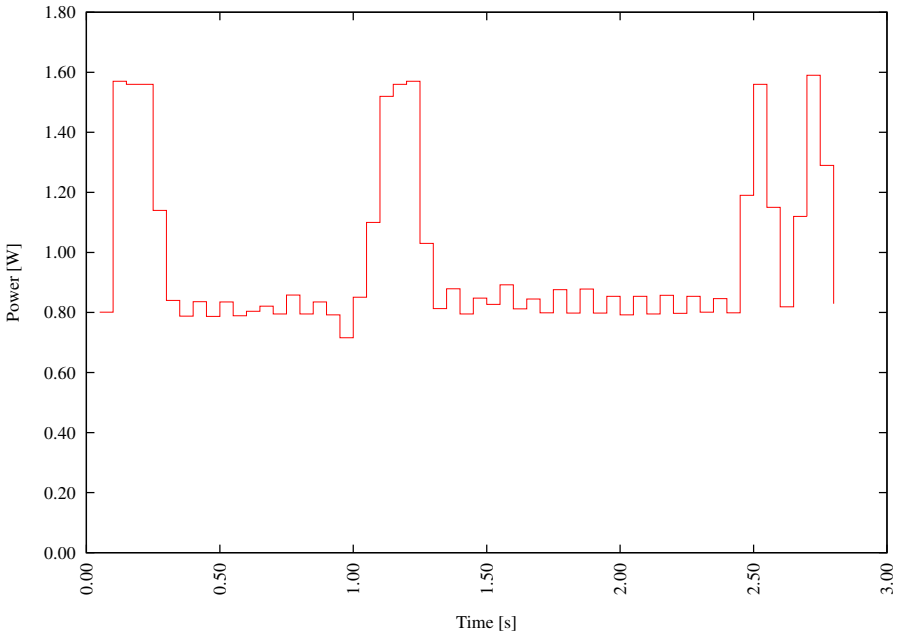

図 2 Java 言語で実装された fasta の消費電力
Fig. 2 Power consumption of fasta implemented in Java.

に書き出すフェーズがある．なお，本実験では標準出力はスペシャルファイル `/dev/null` にリダイレクトで行った．電力が上昇した区間ではこれらの両者が同時に行われた可能性も考えられるが，はっきりとした理由は現在のところ不明である．

電池の遅いレスポンスのため 1 Hz 以上の周波数成分の電池への影響は軽微であり，簡単のため，われわれはこの影響を無視する[3]．この場合，図 2 の電力の揺らぎは無視でき，一定値と仮定できる．

この消費電力の揺らぎの幅は，言語によって異なった．**図 3** に，配列要素に逐次的にアクセスし，この値を用いた簡単な算術演算の結果を別の配列に逐次的に格納したときの消費電力の推移を示した．プログラミング言語は Ruby および C を用いた．C 言語では 50 ms ごとに消費電力が揺らいでいたが，823 mW から 902 mW の区間に収まる帯域で安定していた．一方，Ruby は 0.20 W から 1.42 W までの広い帯域を 0.2 s から 0.5 s くらい

Vol.2011-SE-172 No.4
Vol.2011-EMB-21 No.4
2011/5/17

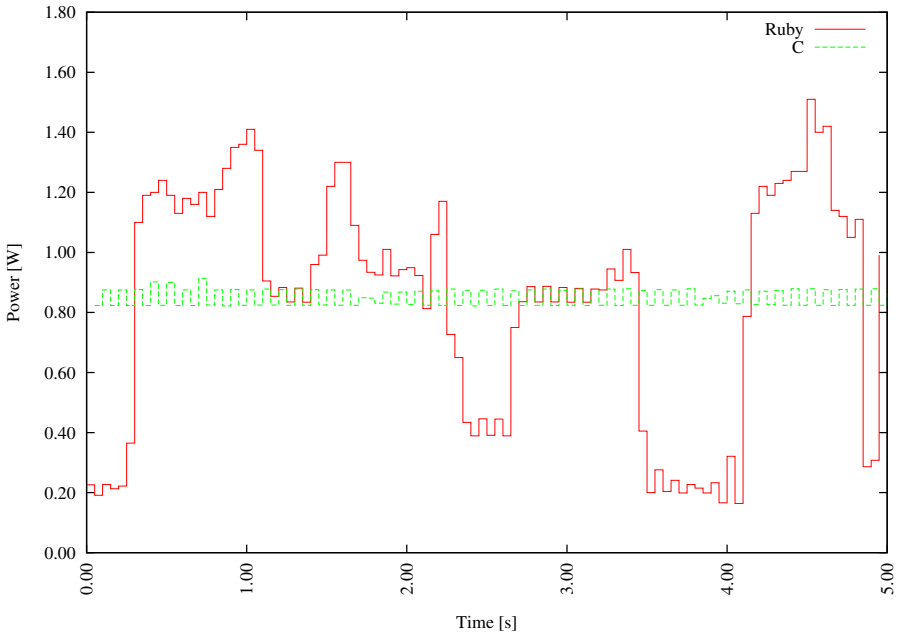

図 3 Ruby および C 言語で実装された配列のコピーの消費電力
Fig. 3 Power consumption of copying arrays in Ruby and C.

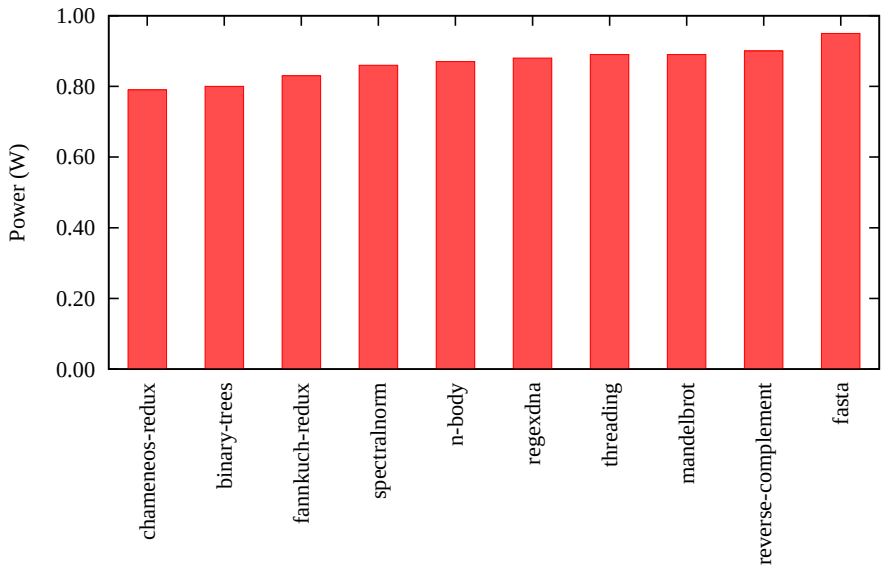

図 4 Java 言語のアプリケーションの消費電力
Fig. 4 Power comsumption of Java applications.

の電力が安定した区間に分割された．これはコンパイル言語である C では配列はメモリ空間に連続的に格納され使用メモリが小さく CPU 負荷量が安定しているのに対し，Ruby はインタープリタ言語であり仮想マシンにおける解釈実行のオーバヘッドやさらにはガーベジコレクションなどの複雑な機構が導入されているため消費電力の大きな揺らぎが観測されたと考えられる．しかし，Ruby の平均消費電力は 844 mW であり，C 言語の 850 mW とほぼ同じ結果となった．1 Hz 以上の周波数成分を無視すれば，この 2 者が電池に与える影響はほぼ同じと考えられる．ただし，Ruby による実装の方が C 言語のものよりも実行時間が長いため，消費エネルギーも高くなるのはいうまでもない．

図 4 に，Java 言語で実装された各アプリケーションの Nokia N900 における消費電力を示した．アプリケーションは消費電力をキーとして昇順に並べた．電力の計測区間は，アプリケーションを実行し 250 mW を上回った時点からアプリケーションの実行が終了し 250 mW を下回った時点とした．消費電力が最大の fasta と最小の chameneos-redux では約 17 % の値の差が観測された．同一言語で高 CPU 負荷なアプリケーションでもこれだけの誤差があり得るということが分かった．

図 5 に，Nokia N900 においてアプリケーション fasta および fannkuch-redux をプログラミング言語 PHP, Ruby, Java でそれぞれ実装したものを実行したときの消費電力を示した．実装する言語により消費電力が異なった．さらに，消費電力の順位は Ruby と Java において逆転していた．このことから，Ruby と Java を考えたとき，消費電力最適化を目標にした場合，fasta を実装するのは Ruby の方が良く，fannkuch-redux を実装するのは Java の方が良いと結論が得られる[⋆1]．もっとも，この揺らぎは高々 15 % であったので，こ

⋆1 消費エネルギー最適化を目標にした場合，実行時間が最短である Java を選択するのが最良であった．

Vol.2011-SE-172 No.4
Vol.2011-EMB-21 No.4
2011/5/17

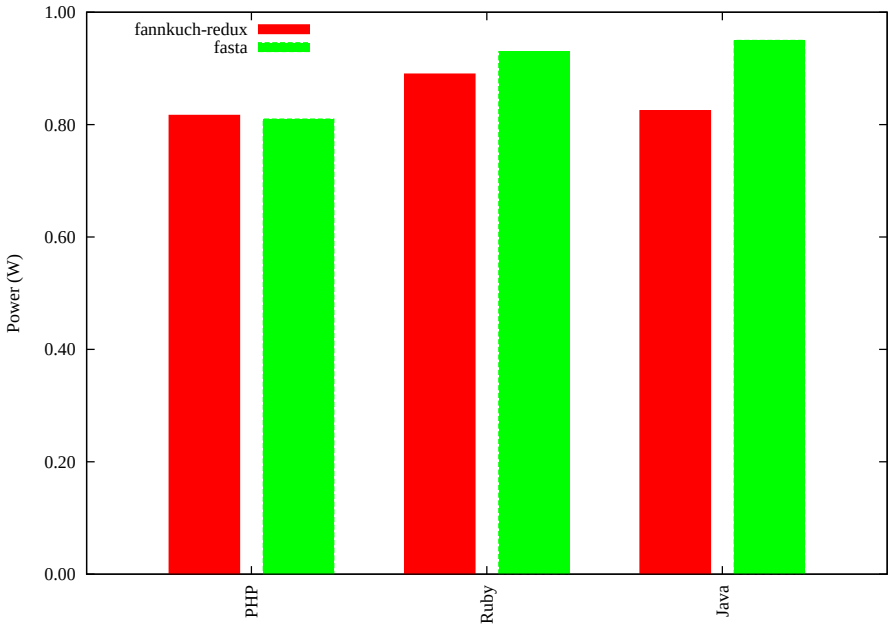

図 5 アプリケーションの消費電力
Fig. 5 Power comsumption of applications.

の実験においてプログラミング言語が消費電力に与える影響は軽微であると言えよう．

5. おわりに

プログラミング言語ごとに情報端末の消費エネルギーが変化する量を定量的に計測した．本稿では，消費電力はアーキテクチャやチップセットの依存度が高く，実行時間を除くプログラムの振る舞いが消費電力に与える影響は軽微であることが確認された．その影響は，われわれの用いた端末において Java アプリケーションでは高々 17 % の消費電力の変化が観測されたにとどまった．配列コピーの実験では，スクリプト言語では消費電力の大きな揺らぎが観測されたが，平均をとるとコンパイル言語が消費した電力値と変わらなかった．もっとも，プログラミング言語ごとに局所的に大きな揺らぎが観測されたのは事実であり，特殊なアプリケーションでは，消費電力の傾向が大きくことなる可能性は排除しきれない．

これらの事実から，プログラミング言語ごとの消費電力の優劣は限られており，実行時間の短縮が消費エネルギーの最適化に繋がることが確認された．

ターゲット端末上ですべてのアプリケーションが動作しなかったため，網羅的なデータを収集することができなかった．また，他の情報端末でもプログラミング言語が与える消費エネルギーの影響を調べたい．これらは今後の課題である．

謝辞 本研究の一部は，公益財団法人 堀科学芸術振興財団（旧 財団法人 堀情報科学振興財団）の研究助成を受け実施された．ここに深く謝意を表する．

参 考 文 献

1) Albers, S.: Energy-efficient algorithms, *Commun. ACM*, Vol.53, pp.86–96 (2010).
2) Barr, K.C. and Asanović, K.: Energy-aware lossless data compression, *ACM Trans. Comput. Syst.*, Vol.24, No.3, pp.250–291.
3) Martin, T. L.: Balancing Batteries, Power, and Performance: System Issues in CPU Speed-Setting for Mobile Computing, PhD Thesis, Carnegie Mellon University (1999).
4) Tavli, B., Bagci, I. E. and Ceylan, O.: Optimal data compression and forwarding in wireless sensor networks, *Comm. Letters.*, Vol. 14, pp. 408–410 (online), DOI:http://dx.doi.org/10.1109/LCOMM.2010.05.092372 (2010).
5) Yao, F., Demers, A. and Shenker, S.: A scheduling model for reduced CPU energy, *FOCS '95: Proceedings of the 36th Annual Symposium on Foundations of Computer Science (FOCS'95)*, Washington, DC, USA, IEEE Computer Society, p.374 (1995).
6) Yokoyama, T., Zeng, G., Tomiyama, H. and Takada, H.: Analyzing and Optimizing Energy Efficiency of Algorithms on DVS Systems: A First Step towards Algorithmic Energy Minimization, *Asia and South Pacific Design Automation Conference. Proceedings*, Yokohama, Japan, pp.727–732 (2009).
7) Yu, Y., Krishnamachari, B. and Prasanna, V.K.: Data Gathering with Tunable Compression in Sensor Networks, *IEEE Trans. Parallel Distrib. Syst.*, Vol.19, pp. 276–287.
8) 横山哲郎, 今井敬吾, 曾剛, 冨山宏之, 高田広章, 結縁祥治, 動的電圧制御システムにおける評価戦略選択に基づく高効率消費エネルギー関数型プログラミング, 情報処理学会論文誌：プログラミング, Vol.2, No.2, pp.54–69 (2009). j
9) 横山哲郎, 藤井勝之, 神山剛, 冨山宏之, 高田広章：消費エネルギーを意識した可逆圧縮データ受信, Vol. 2011-EMB-20, No.1, pp.1–6, 2011.