A-27　RISCアーキテクチャに適した差分圧縮アルゴリズム

Differential Compression Algorithm available for RISC Architectures

寺薗 浩平†　岡田 佳之†
Kohei Terazono　Yoshiyuki Okada

1．背景

プロセッサの高性能化，メモリの大容量化に伴い，ＰＤＡ，携帯電話，ゲーム機などの組み込み型システムでは，急速に高機能化が進んでいる。しかし，その一方で機能追加や仕様変更などの機会も増えてきており，パーソナルコンピューターと同様に簡単に内蔵ソフトウェアの更新ができる仕組みが強く求められている。

2．目的

最近の組み込み型システムには，外部インタフェースを持つものが増えてきたが，無線などの狭帯域な外部インタフェースが多い。そのような環境下で，効率良くソフトウェアを更新するには，転送するデータ量を極力少なくできる差分圧縮[1][2]を用いるのが効果的である。

本稿では，組み込み型システムのソフトウェア更新を想定して，更新するデータ量の 10%以下に圧縮することを目標に，ＡＲＭなどの RISC プロセッサ[3]の特徴を利用した高効率な差分圧縮アルゴリズムを検討したので報告する。

3．アルゴリズム

3.1　差分の基本アルゴリズム

差分圧縮アルゴリズムの基本概念は，新データの先頭から逐次，同じデータ列が旧データ中に存在するかどうかを検索し，検索した新旧データの差分のみを符号化するものである。

具体的な処理の流れとしては，

(1)　新旧データの先頭からデータを順次比較し，データ列が一致するならば「複写」という表現で，データ列の一致長を符号化する

(2)　一致するデータ列が存在しないならば，「置換」あるいは「新規」という表現で，データ列をそのまま符号化する

となる。図１の例で示すように，新旧同一部分を「複写」とし、短い符号で符号化することで差分圧縮効率が上がる。

しかしながら，命令コードの一部が変更されるようなプログラム更新の場合，「複写」部分が短くなり，「複写」と「置換」を繰り返すことで細切れに符号化されるので，基本アルゴリズムでは，差分圧縮効率が十分ではなかった。

（図３の基本アルゴリズムの処理例を参照）

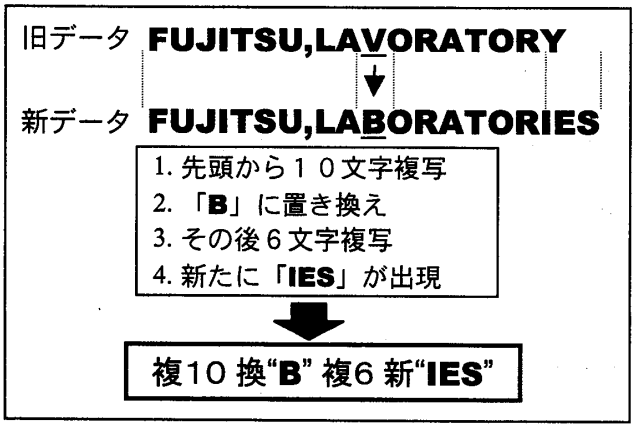

図１．差分の基本アルゴリズム概念図

3.2　ＲＩＳＣアーキテクチャに適したアルゴリズム

基本アルゴリズムの前記欠点を補い，差分効率を上げるため，RISC アーキテクチャの特徴を利用した差分圧縮アルゴリズムを新たに考案した。

RISC アーキテクチャの場合，CISC と比較して

- 同一フォーマットによる固定長の命令サイズ
- ロード／ストア構造から，データ処理の命令セットがレジスタのみで処理

という特徴を持つ。これらの特徴と，更新前後のプログラムコード解析から，

- データ転送命令などの指定レジスタの変更に関与する全ての命令セット
- 分岐命令などのアドレス指定変更に影響のある全ての命令セット

で，命令コード単位で同一の変更パターンが連続的に続くことがわかった。

この性質を捉えて，新たな差分圧縮アルゴリズムを考案した。考案アルゴリズムの処理手順を次に示す。

（１）　新旧データを命令単位で同一命令のレジスタ或いはアドレス部分を比較する
（２）　各命令の指定レジスタ或いは指定アドレスの変更パターンの同一連続性を「値の増減」と「同一パターンの繰返し回数」で符号化する

図２に考案アルゴリズムのブロック構成を示す。考案アルゴリズムは，命令単位の差分抽出部と差分値の同一繰返符号化部に分かれる。基本アルゴリズムと並行して

† （株）富士通研究所, Fujitsu Laboratories, Ltd.

実行し，適応的に切り替えることで効率の良い符号化を実現する。

図３に考案アルゴリズムの処理例を示す。基本アルゴリズムではバイト単位で比較するため，「複写」と「置換」を繰り返すのに対して，考案アルゴリズムでは命令コード単位で比較するため，３つの命令セットを「値の増減」と「同一パターンの繰返し回数」で表現でき，効率の良い差分圧縮が期待できる。

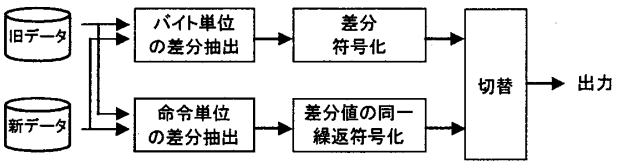

図２．考案アルゴリズムのブロック構成

4. 性能評価

考案アルゴリズムの差分圧縮の性能評価を行った。

サンプルデータとしてはRISCプロセッサ（ARM）で処理されるプログラムで変更を行った新旧データを３組用意した。比較的変更の多い１種類（サンプル 1）と少ない２種類（サンプル 2，3）を選択した。これらのサンプルデータを用いて，基本及び考案アルゴリズムで差分抽出を行い，生成された差分圧縮のデータサイズを比較した。

図４に差分圧縮の性能評価結果を示す。RISC アーキテクチャに最適化したアルゴリズムは，基本アルゴリズムと比較して高い差分圧縮効率を示しており，変更の多いサンプル 1 でも目標の 10%以下を十分達成している。変更が行われた全ての命令コードのうち「置換」の変更がなされた命令コードの割合（置換変更率）が高いもの（サンプル 3）ほど差分圧縮効果が顕著に現れていることを検証した。

また，バイナリコードにおいて小さな差分圧縮データを生成できると言われ、広く使用されているフリーソフトウェアの「WSP version 1.50」[4]の差分圧縮結果を図４に付加した。WSP との比較においても、最適化アルゴリズムの方が差分圧縮効率で優れていることを確認した。

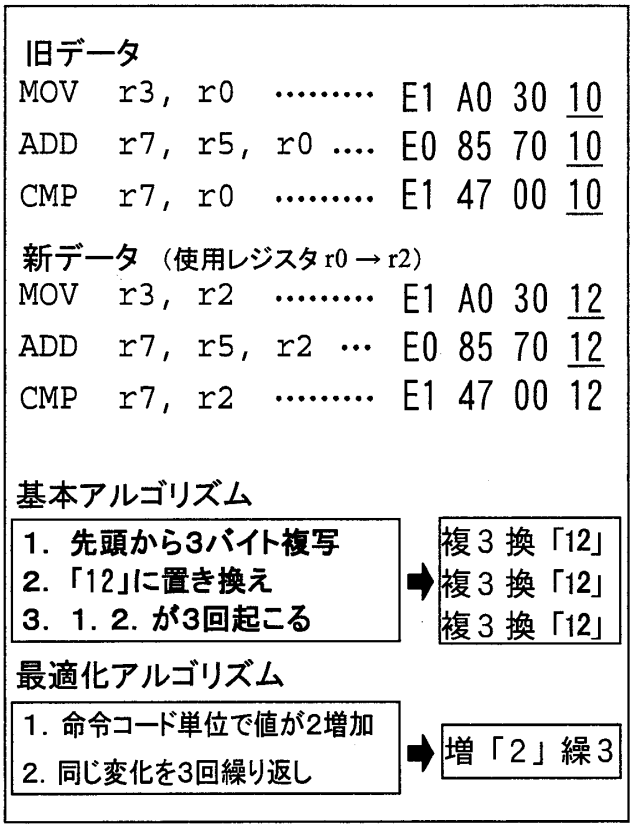

図３．考案アルゴリズムの処理例

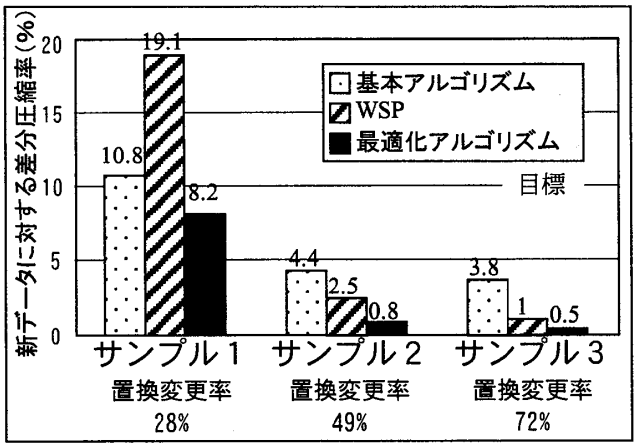

図４．考案アルゴリズムの差分圧縮性能

5. まとめ

本稿では、組み込み型システムのソフト更新を対象に，システムで多く使用されている RISC プロセッサの命令コード変化の冗長性を捉えた新しい差分圧縮アルゴリズムを考案した。考案アルゴリズムは，基本アルゴリズム或いは従来の差分圧縮ソフトと比較して高い差分圧縮効率を示し，更新データの 10%以下という目標を達成した。

組み込み型システムを主体にした情報・携帯機器は，外部インタフェース或いはネットワークを介した機器間の連携が，今後強まると予想される。そのような状況下で常にシステムを最新の状態にグレードアップするには，内蔵ソフトウェアを効率良く更新していく機能が増々必要になってくる。今後，このような分野への差分圧縮技術の応用を図っていきたい。

〔参考文献〕

[1] James J. Hunt, Kiem-Phong Vo and Walter F. Tichy "Delta Algorithms: An Empirical Analysis" ,ACM Transactions on Software Engineering and Methodology, 7(2), pp.192-214, 1998

[2] Andrew Tridgell and Paul Mackerras "The rsync algorithm" ,Ottawa Linux Symposium ,2000

[3] Steve Furber , "改訂／ARM プロセッサ～32 ビット RISC のシステム・アーキテクチャ～",CQ 出版, 2001

[4] 奥村晴彦, "binary diff, delta compression" http://www.matsusaka-u.ac.jp/~okumura/compression/diff.html