4A-2

POWER7/VSX 機構向けアラインメント最適化

Alignment Optimization for POWER7/VSX

吉田 美里† 橋本 博幸† 本川 敬子†
Misato Yoshida Hiroyuki Hashimoto Keiko Motokawa

†（株）日立製作所 横浜研究所
† Hitachi, Ltd., Yokohama Research Laboratory

1. はじめに

演算装置において1回の命令実行で複数のデータに対する演算を行う方式を SIMD(Single Instruction Multiple Data)と呼ぶ．近年マルチメディア処理の重要性の高まりに伴い，多量のデータに対して同じ演算を行うのに適した SIMD 命令セットをサポートしたプロセッサが増加している．POWER7 プロセッサ[1]は SIMD 命令セットをサポートした VSX(Vector-Scalar Extension)機構[2]を搭載しており，SIMD 命令の利用により1命令で 16 バイトデータの演算を行うことができる．

本稿ではまず，連続的な配列参照を含むループへの SIMD 命令適用で生成する 16 バイトデータのロード・ストア命令に対して，対象データのアラインメントの違いによる実行性能調査結果を示す．調査の結果，アラインメントの違いによって性能劣化する場合があることがわかった．本研究では，アラインメントの違いによる性能劣化を防ぐため，ループ変換の一方式であるピーリングによるアラインメント最適化方式を提案する．

2. VSX におけるロード・ストア命令の性能

VSX の SIMD ロード・ストア命令では 16 バイトデータを扱うため，対象データの先頭アドレスが 16 バイト境界に合っている場合をアライン，そうでない場合をアンアラインと定義しており，アライン時に最も性能を発揮する[2]．アライン時とアンアライン時の性能差を調べるために，POWER7 のキャッシュラインサイズ 128 バイト中の 16 バイト境界を跨ぐ各データ(図1 (1)-(8))に対して SIMD ロード・ストア命令の性能を調査した．測定には，3.0GHz の POWER7 を搭載する日立 EP8000 を使用した．

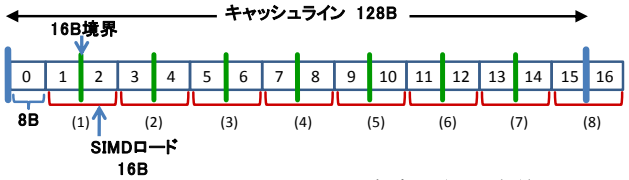

図1 SIMD ロード・ストア命令性能測定箇所

図2の横軸は，キャッシュライン先頭からのオフセットを表し，横軸の数字は図1各データの先頭アドレスのオフセットを示す．縦軸は，アライン(オフセットが 0)時を1とした時の SIMD ロード・ストア実行時間比を表す．図2より，SIMD ロードは先頭アドレスのオフセットが 120 (図1 (8))の時，つまり，128 バイト境界を跨ぐ時に性能劣化し，SIMD ストアは先頭アドレスのオフセットが 24, 56, 88, 120 (図1 (2)(4)(6)(8)) の時，つまり，32 バイト境界を跨ぐ時に性能劣化することがわかる．また，アライン時に比べ，ロードは約 1.7 倍劣化し，ストアは約 2 倍劣化することがわかる．

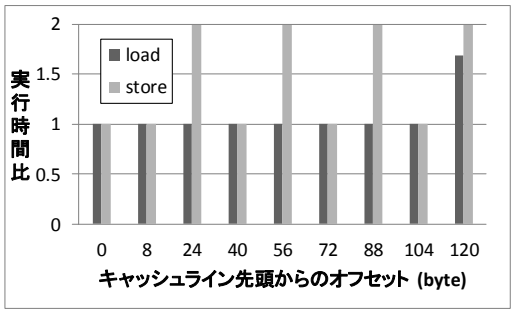

図2 SIMD ロード・ストア命令基本性能

3. アラインメント最適化

本研究では，連続的な配列参照を含むループに対してアンアライン配列参照が増加した時，アラインな配列参照を増やすためのアラインメント最適化として，ループ変換の一方式であるピーリングを行う．本研究のピーリングでは，配列初回参照が 16 バイト境界を満たすように，境界からのオフセット分だけ先頭イタレーションを非 SIMD 演算し，残りのイタレーションを SIMD 演算する．

図3において，配列 a-c は倍精度浮動小数データ(8 バイト)が格納してあり，先頭アドレスはすべて 16 バイト境界に合っているとする．また，配列添字は 0 から始まるとする．図3(a)を SIMD 化した際の配列 b アクセス状況を図4(a)に示す．配列 b の初回参照は b[1]であり，16 バイトずつ(2 要素ずつ)ロードを行うと毎回アンアライン参照となる．この場合，2章の測定結果によれば，8 回に 1 回 128 バイト境界を跨ぐ時に性能劣化する．そこで，配列 a-c の参照がアラインになるようにループイタレーション 1 回分だけ非 SIMD 演算を行い，残りのイタレーションを SIMD 演算することで配列 a-c がアライン参照となる(図3(b))．ここで，図3(b)の a[i:i+1]は 2 要素の一括参照を表し，⊞は SIMD 加算を表す．ピーリング適用後の配列 b アクセス状況を図4(b)に示す．

```
for (i=1; i<n; i++) {
  a[i] = b[i]+c[i];
}
```

(a) 変換前

```
a[1] = b[1]+c[1];        // ピーリング
for (i=2; i<n-1; i+=2) {
  a[i:i+1] = b[i:i+1] ⊞ c[i:i+1];
}                        // SIMD 化
// 余りループ(省略)
```

(b) 変換後

図3 ピーリングの例

†(株)日立製作所 横浜研究所
〒244-0817 神奈川県横浜市戸塚区吉田町 292 番地

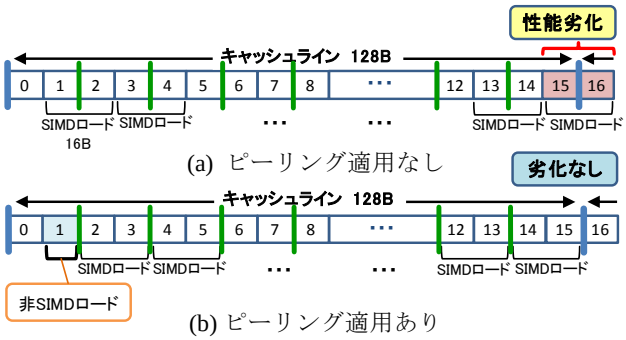

図４　ピーリング適用前後の配列 b アクセス状況

配列の先頭アドレスや制御変数の下限値が静的に不明な場合は，配列の初回参照アドレスが静的に求められないため，動的判定コードを挿入する必要がある．動的判定コードを含めたピーリングコードを図５に示す．初回参照が 16 バイト境界に合っていない場合には，ループイタレーション１回分だけ非 SIMD 演算を行った後 SIMD 演算を行い，初回参照が 16 バイト境界に合っている場合は SIMD 演算のみを行う．

```
for (i=1; i<n; i++) {
  a[i] = b[i]+c[i];
}
```

(a) 変換前

```
i=1;
if (&b[1]%16 != 0) {  // 動的判定
  a[1] = b[1]+c[1];     // ピーリング
  i=i+1;
}
for (; i<n-1; i+=2) {
  a[i:i+1] = b[i:i+1] + c[i:i+1];
}                              // SIMD 化
// 余りループ(省略)
```

(b) 変換後

図５　動的判定付きピーリングコードの例

ピーリング適用基準は次の通りとする．(1)すべての初回参照アドレスが静的に求まり，アンアライン参照の配列が過半数以上の場合，ピーリングを実施する．(2) 動的判定が必要な場合，同アラインメントの数が最大のものから１つを選択し判定対象とする．

４．性能評価

図６のループに対してピーリングの評価を行った．配列 a-e は倍精度浮動小数データで，前提は３章と同様とする．b の配列参照形式を b[i+1]に変更した場合，初回参照は b[1]であり， SIMD ロードで 16 バイトずつ処理を行うと，常に 16 バイト境界を跨ぎアンアライン参照となる(図１，図４(a)参照)． b,c,d,e,a の配列参照形式を上記のように変更し，アンアラインな参照を増やした時の実行時間増加の様子とピーリングの効果を図７に示す．図７は，すべての参照がアライン(図６の状態)で SIMD 化を行った時の実行時間を１とした時の実行時間比を示す．実行は１プロセッサ上で行い，データ領域は L2 キャッシュ(256KB)を対象とした．

```
for (i=0; i<n; i++) {
  a[i] = b[i]+c[i] +d[i] +e[i];
}
```

図６　測定対象ループ

アンアライン参照が過半数以下の時にはピーリングを行わないため，SIMD 化のみを適用した場合と比較して性能の差はない．アンアライン参照が過半数を超えた時には，ピーリングによりアンアライン参照をアライン参照に変更するため，SIMD 化のみを適用した場合と比較して実行時間が大きく削減できている．

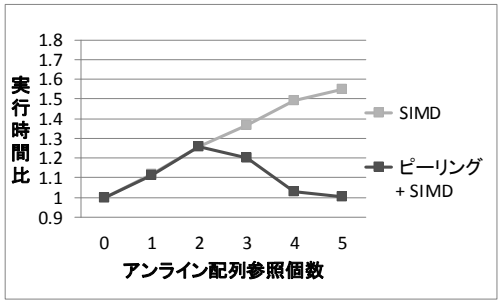

図７　アンアライン参照の増加とピーリングの効果

配列の初回参照アドレスが静的に不明な場合の動的アドレス判定付きピーリングの効果を図８に示す．アンアライン参照が過半数以下の時には，判定コード分だけロスが生じるが，アンアライン参照数が過半数以上の時には，SIMD ロード・ストアの性能劣化が大きいため，動的アドレス判定オーバーヘッドを含んでもピーリングを適用した方が有利である．

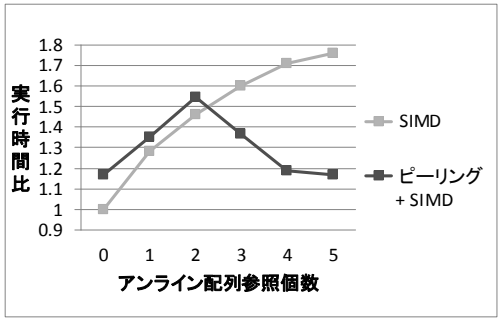

図８　動的判定付きピーリングの効果

５．おわりに

POWER7 プロセッサは SIMD 演算が行える VSX 機構を搭載しており 16 バイトずつデータの処理が行えるが，アンアラインな配列参照が増えると性能が劣化する．

アンアラインな SIMD ロード・ストア実行を削減するために，ループの初回参照がアンアラインな配列参照に対して，先頭イタレーションをピーリングしアラインを調整した後 SIMD 化を行う方式を適用し，性能劣化が防げることを確認した．

参考文献

[1] IBM Corporation：Press releases(08 Feb 2010),http://www-03.ibm.com/press/us/en/pressrelease/29315.wss, (2010).

[2] IBM Corporation, “Power ISA Version 2.06 Revision B”, (July 23, 2010).