SS研
ポストペタアプリ性能WG資料

FUJITSU
shaping tomorrow with you

3.4 電解磁場解析「ADVENTURE_Magnetic」DAXPYのSIMD化について

2015年7月1日
富士通株式会社
千葉　修一

Copyright 2015 FUJITSU LIMITED

サマリー（1）

FUJITSU

■課題

DAXPYコードなどSIMD化されない理由が不明確

■対象コード

■電解磁場解析「ADVENTURE_Magnetic」

■DAXPY：aij_solver_cg.hの367行目のループ

■メッセージ

コンパイラの世代によってメッセージが異なる

① jwd6101s-i "aij_solver_cg.h", line 367: ループ内にSIMD化の制約となる文が存在します。

② jwd6202s-i "aij_solver_cg.h", line 369: ループ中で変数'aq'を定義する順序が逐次実行と異なるため、このループはSIMD化できません。

Copyright 2015 FUJITSU LIMITED

サマリー（2）

FUJITSU

■回答

■SIMD化されない原因

- ポインタアクセスにより、依存関係が無いことが判断できないため
- コンパイラに対する指示によりSIMD化が可能

■メッセージが異なる原因

- メッセージの改善により以下の版から出力内容が変更
- 「京」16版　　(2014年3月)
- FX10 V1.0L30 (2013年9月)

■コンパイラへのフィードバック

- 判りやすいメッセージへの改善（短期課題）
- ポインタのSIMDの強化（長期課題）

Copyright 2015 FUJITSU LIMITED

分析（1）

FUJITSU

■対象コード

```
#define MTYPE double
#define MPLUS( x, a, b )          ( (x) = (a) + (b) )
#define MMULTI( x, a, b )         ( (x) = (a) * (b) )
```

```
MTYPE  *r ;
MTYPE  *aq ;
 r = (MTYPE *)Malloc( sizeof(MTYPE), nf ) ;
aq = (MTYPE *)Malloc( sizeof(MTYPE), nf ) ;
```

```
for( i=0 ; i<nf ; i++ ) {
  MTYPE  dd ;
  MMULTI( dd, alpha, aq[i] ) ;
  MPLUS( r[i], r[i], dd ) ;
}
```

ループ内のデータに依存あり

Copyright 2015 FUJITSU LIMITED

分析（2）

FUJITSU

■データ依存の分析

■展開コード

```
for( i=0 ; i<nf ; i++ ) {
  double  dd ;
  dd = alpha * aq[i];
  r[i] = r[i] + dd;
}
```

■コンパイラの判断

- 変数aq と 変数r はポインターであり依存が不明
- SIMD化するとデータアクセス順序が変化すると判断

Copyright 2015 FUJITSU LIMITED

分析（3）

FUJITSU

■SIMD化によるデータアクセス順序の変化

```
MTYPE aq = &temp[i];
MTYPE r  = &temp[i+1];
```

```
LOOP:
 load  aq[i] -> s;
 store  r[i] <- s;
goto LOOP;
```

SIMD化

```
LOOP:
 vload  aq[i:i+1] -> v;
 vstore  r[i:i+1] <- v;
goto LOOP;
```

※ ストア／ロードのみの簡易コードで評価

データアクセス順序

```
-Iteration 1,2-
 load  temp[i]   -> s;
 store temp[i+1] <- s;
 load  temp[i+1] -> s;
 store temp[i+2] <- s;
```

temp[i+1]に対するアクセス順序が変化

```
-Iteration 1-
 load  temp[i]   -> s;
 load  temp[i+1] -> s;
 store temp[i]   <- s;
 store temp[i+1] <- s;
```


Copyright 2015 FUJITSU LIMITED

対処（1）

FUJITSU

■restrict 修飾子による対処

```
#define MTYPE double
#define MPLUS( x, a, b )          ( (x) = (a) + (b) )
#define MMULTI( x, a, b )         ( (x) = (a) * (b) )
```

```
MTYPE  * restrict r ;
MTYPE  * restrict aq ;
 r = (MTYPE *)Malloc( sizeof(MTYPE), nf ) ;
aq = (MTYPE *)Malloc( sizeof(MTYPE), nf ) ;
```

```
for( i=0 ; i<nf ; i++ ) {
  MTYPE  dd ;
  MMULTI( dd, alpha, aq[i] ) ;
  MPLUS( r[i], r[i], dd ) ;
}
```

[restrict 修飾子]
指定した変数の
ポインタのアクセスが
重ならないことを指示

Copyright 2015 FUJITSU LIMITED

対処（2）

FUJITSU

■最適化指示行 NORECURRENCEによる対処

```
% fccpx -Kocl ...
```

```
#pragma loop norecurrence
for( i=0 ; i<nf ; i++ ) {
  ...
```

[norecurrence 指示子]
ループ内の
ポインタのアクセスが
重ならないことを指示

■翻訳オプション -Krestp による対処

```
% fccpx -Krestp=all ...
```

[-Krestp=all]
ファイル内のポインタのアクセスが重ならないことを指示

[-Krestp=arg]
関数の引数のポインタのアクセスが重ならないことを指示

Copyright 2015 FUJITSU LIMITED

コンパイラへのフィードバック

FUJITSU

■メッセージの改善

メッセージの改善を検討する。

メッセージ例)
jwd*NNNN*s-i "aij_solver_cg.h", line 369: ループ中で変数'aq'と変数'r'の依存が不明確なため、このループはSIMD化できません。

■コンパイラの解析能力の強化

C/C++コンパイラにおいて、ポインタ変数が指す領域の重なりを判断できるように改善します。

→ restrict修飾子なしでもSIMD化を適用

Copyright 2015 FUJITSU LIMITED

FUJITSU

shaping tomorrow with you