AVXを用いた倍々精度疎行列ベクトル積 –転置行列-

菱沼 利彰†, 藤井 昭宏†, 田中 輝雄†, 長谷川 秀彦††
†工学院大学情報学部　††筑波大学図書館情報メディア系

Introduction

Krylov部分空間法などの反復法は，丸め誤差によって収束に影響を受け，改善策として倍々精度の利用が挙げられる．
実問題では倍精度の疎行列が与えられることが多く，倍精度疎行列と倍々精度ベクトルの積の高速化が必要である．
本研究では，SIMD AVXを用いて倍々精度転置疎行列ベクトル積を高速化し，その効果を分析した．

倍々精度演算の概要

- 倍精度浮動小数点数2つを用いて4倍精度演算を実現
- Baileyの”Double-Double”精度アルゴリズム

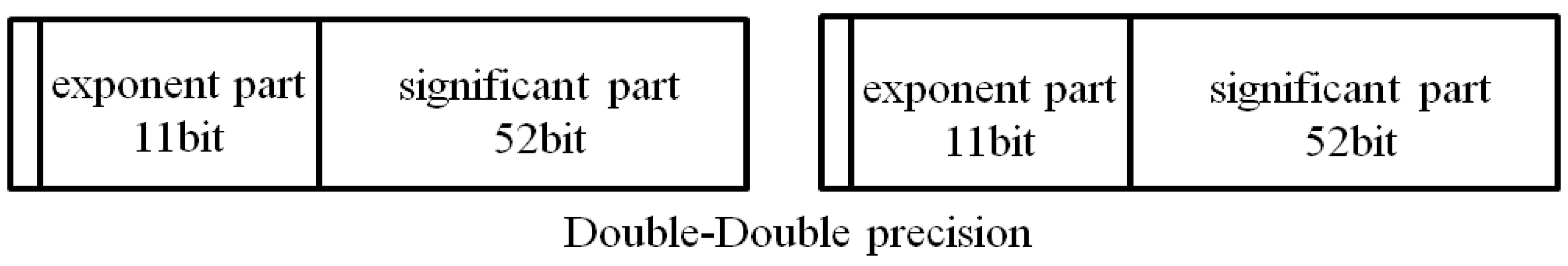

図1 倍々精度のビット長

AVXの概要

- SSE2にかわるSIMD拡張命令
- SSE2　(Streaming SIMD Extensions2) (2000~)
- AVX　(Advanced Vector eXtensions) (2011~)
  - AVXは256bitのデータに対しSIMD演算(倍精度4つ)
  - SSE2は128bitのデータに対しSIMD演算(倍精度2つ)
- AVXはSSE2の2倍の性能が期待できる

A^Txの性能 (Intel Core i7 2600K 3.4GHz)

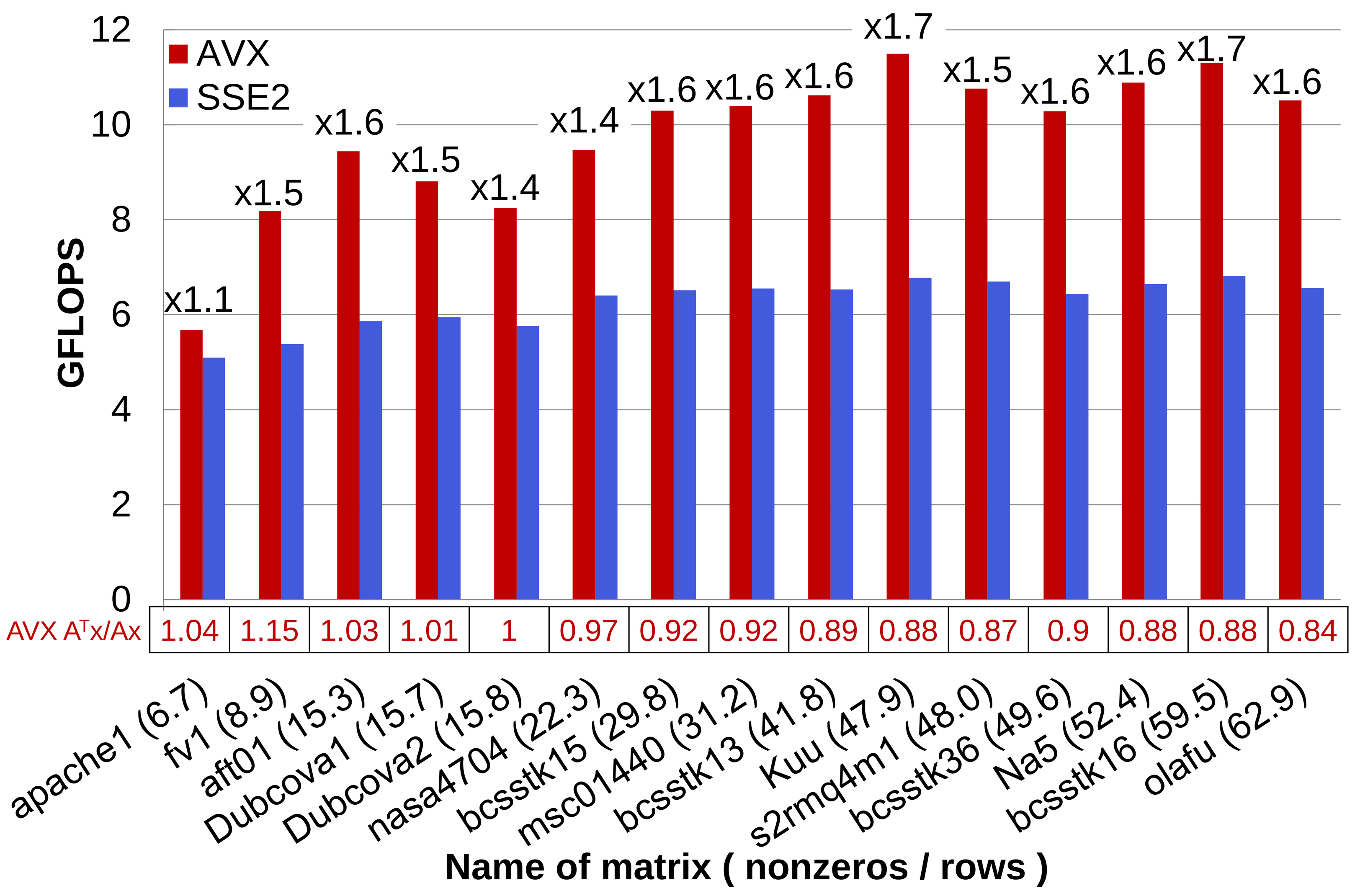

図2 フロリダコレクションの疎行列の性能

- 性能はサイズの影響を受けない
- SSE2と比べAVXは1.1から1.7倍の性能
- A^TxとAxの性能差は小さい
  - SSE2はほぼ同等の性能，AVXは±15%
- AVXはピーク性能の20~42%，SSE2は38~50%

Ax

```
for(i=0 ; i<N ; i++)
    for(j=A_row_ptr[ i ] ; j<A_row_ptr[ i+1 ] ; j++)
        y[ i ] = y[ i ] + A_value[ j ] * x[ A_index[ j ] ]
```

A^Tx

```
for(i=0 ; i<N ; i++)
    for(j=A_row_ptr[ i ] ; j<A_row_ptr[ i+1 ] ; j++)
        y[ A_index[ j ] ] = y[ index[ j ] ] + A_value[ j ] * x[ i ]
```

図3 AxとA^TXのアルゴリズム(擬似コード)

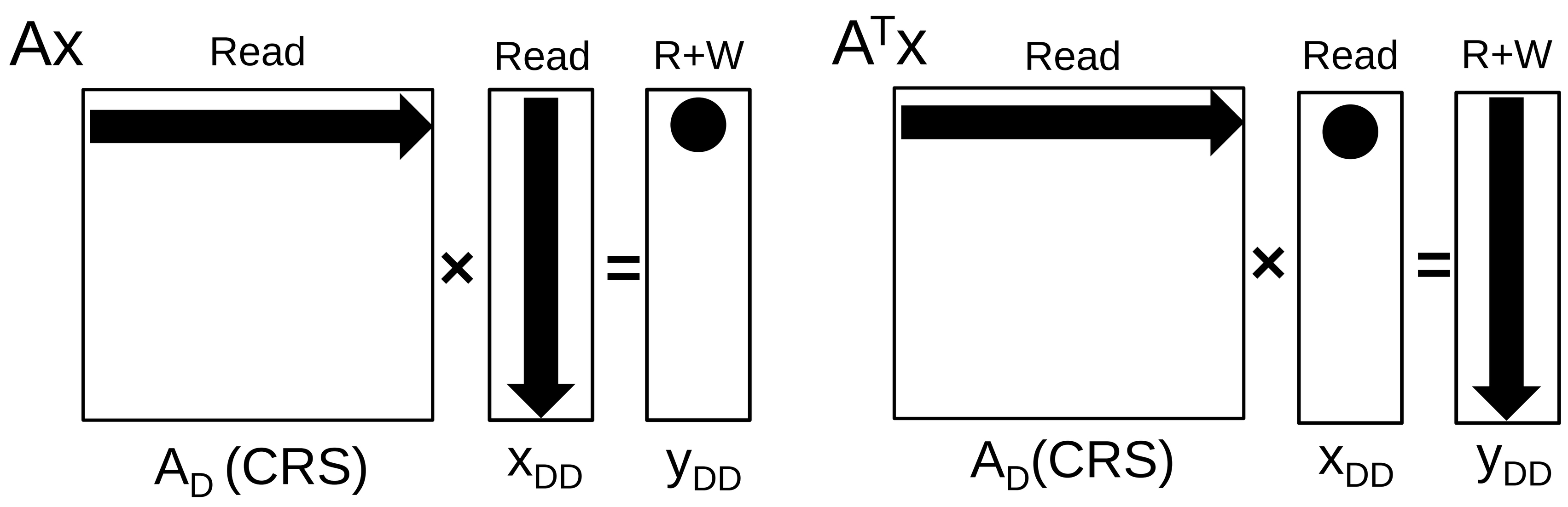

図4 AxとA^TXのデータアクセスの違い

- Axはx_{DD}へのアクセスが性能に影響
- A^Txはy_{DD}のアクセスが性能に影響
- x_{DD}のアクセスはRead，y_{DD}へのアクセスはRead+Write

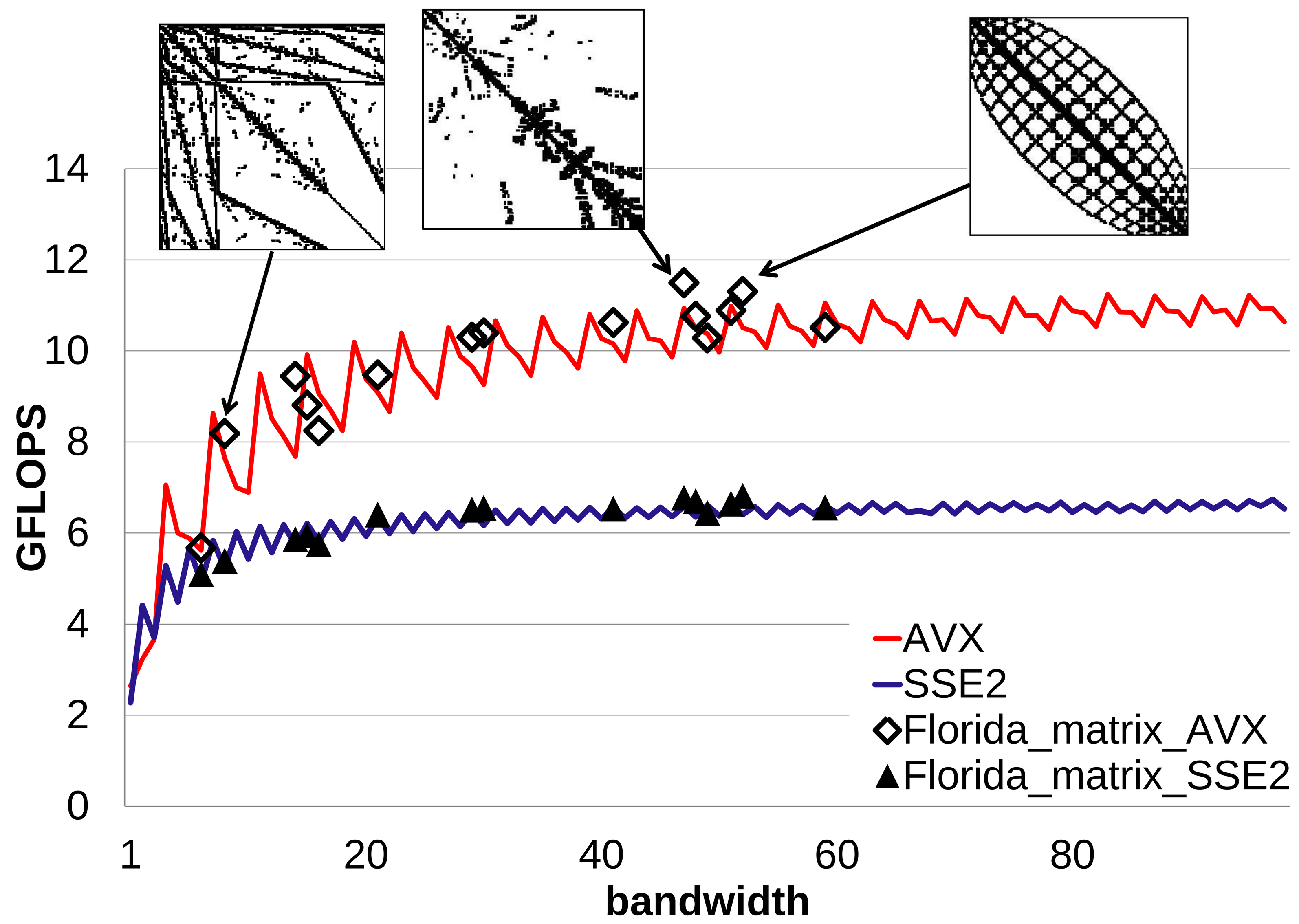

図5 帯幅と性能の関係

- 平均非零要素数が多いほど性能が高い
  - 帯幅80のときピーク性能の41%
  - 帯幅10のときピーク性能の22%
- Ax同様に性能は平均非零要素数に依存
- フロリダコレクションの性能は帯行列の1.05倍から0.91倍

References

- Bailey, D, H.: *High precision Floating-Point Arithmetic in Scientific Computation, computing in Science and Engineering, pp. 54-61 (2005)*
- 反復解法ライブラリLis, http://www.ssisc.org/lis/

Conclusion

- A^TxとAxの性能差は小さく，SSE2ではほぼ同等(ピーク性能の38から50%)，AVXでは±15%(ピーク性能の20から42%)
- A^Txの性能はAxと同様に平均非零要素数に依存し，帯行列と比較したフロリダコレクションの性能は1.05から0.91倍