疎行列ベクトル積に対するOpenMPスケジューリング方式の分析

菱沼 利彰†，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

P1-5

Introduction

倍精度疎行列と倍々精度ベクトルの積において，疎行列の構造により性能が低い場合がある
性能が低い問題では，各スレッドの担当する非零要素の数に偏りがあり，スケジューリング方式によって改善できる可能性がある
実行時に疎行列の特性と最適なスケジューリング方式を評価するために，各行の要素数を標本とした分散を基に分析を行った

実行環境

1. **Sandy Bridge**
   core i7 2600K 3.4GHz 4core 16GB
   intel C/C++ compiler (-O3 –openmp –xAVX -fp-model precise)
2. **FX10 スーパコンピュータシステム (1node)**
   1.848GHz 16core 32GB
   Fujitsu C compiler (-Kfast -Kopenmp)

対象：実数疎行列1699問題 (U.Florida Sparse Matrix Collection)

評価方法

倍精度疎行列(CRS)と倍精度，倍々精度ベクトルの積：$\boldsymbol{y}=A\boldsymbol{x}$
各スレッドの担当要素数の偏りの指標として分散を求めた

OpenMP Scheduling
static, dynamic, guided, balanced*

*事前に担当する要素数を均等化する

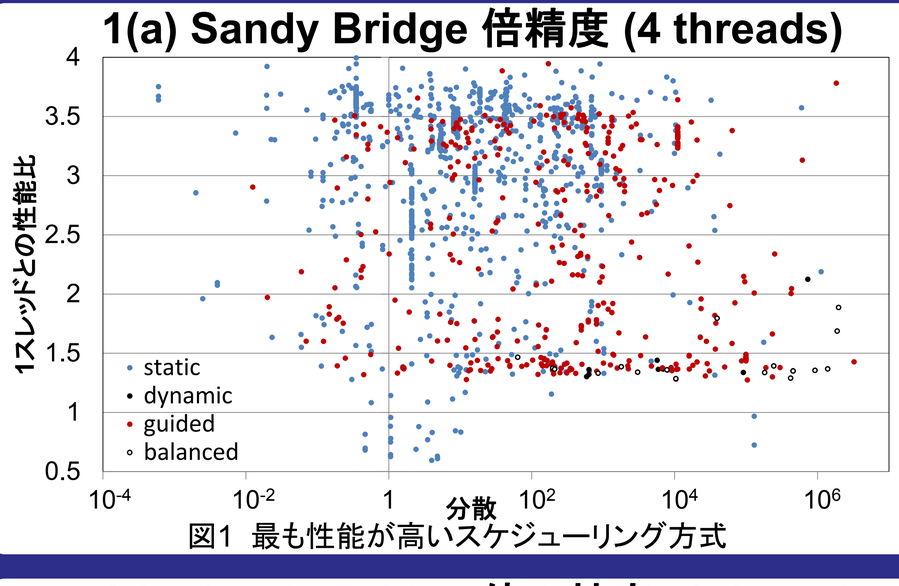

図1　最も性能が高いスケジューリング方式

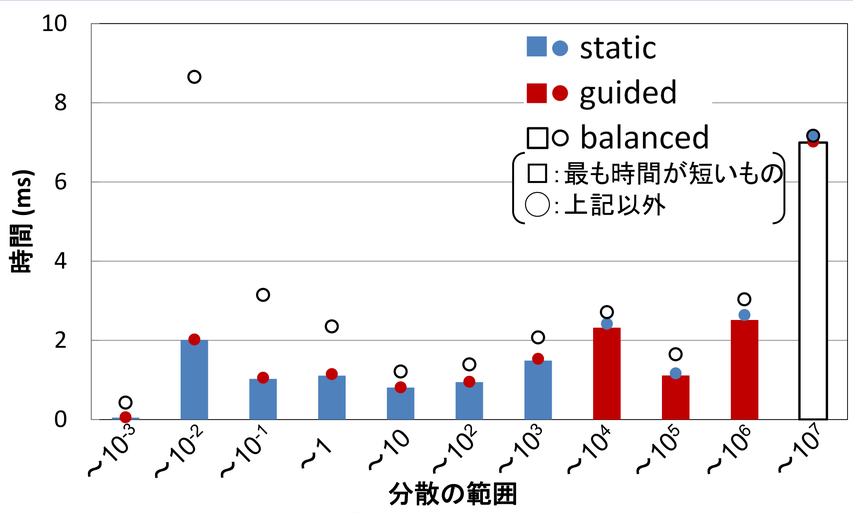

図2　各スケジューリング方式の平均時間 (dynamicは除く)

mix方式

10^7 ≦分散	: balanced
10^3 ≦分散 <10^7	: guided
otherwise	: static

合計時間 (秒)

static	2.02*
dynamic	4.28
guided	2.03
balanced	3.05
mix	2.0
最適値	1.98

-1%

*最も速いスケジューリング方式

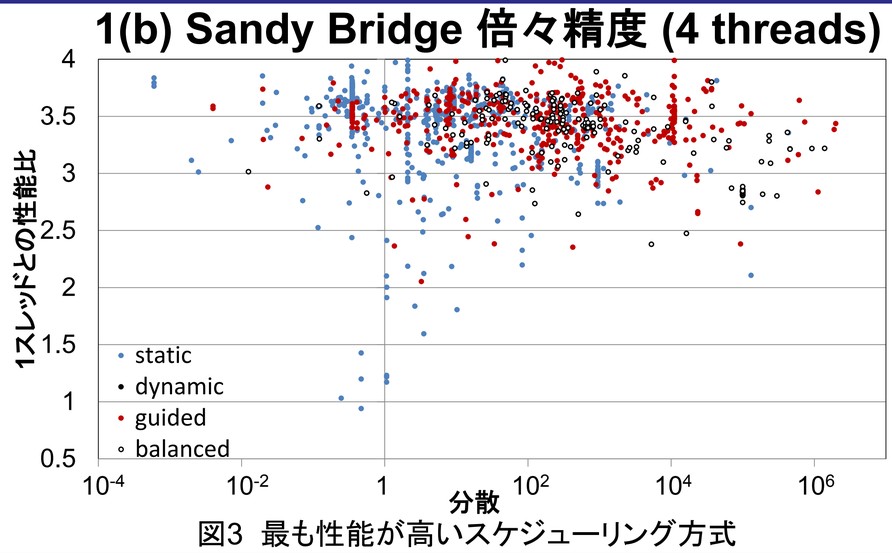

図3　最も性能が高いスケジューリング方式

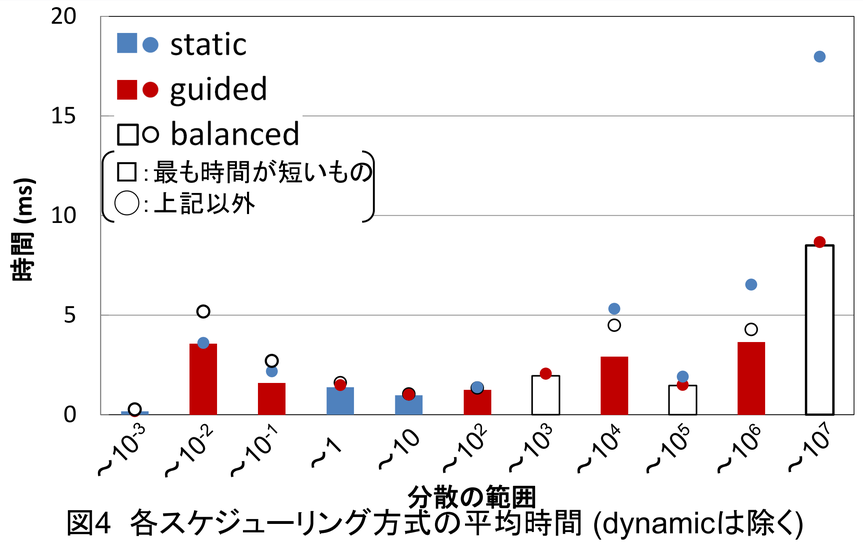

図4　各スケジューリング方式の平均時間 (dynamicは除く)

mix方式

10^7 ≦分散	: balanced
10 ≦分散 <10^7	: guided
otherwise	: static

合計時間 (秒)

static	3.05
dynamic	5.46
guided	2.7*
balanced	2.83
mix	2.59
最適値	2.11

-4%

*最も速いスケジューリング方式

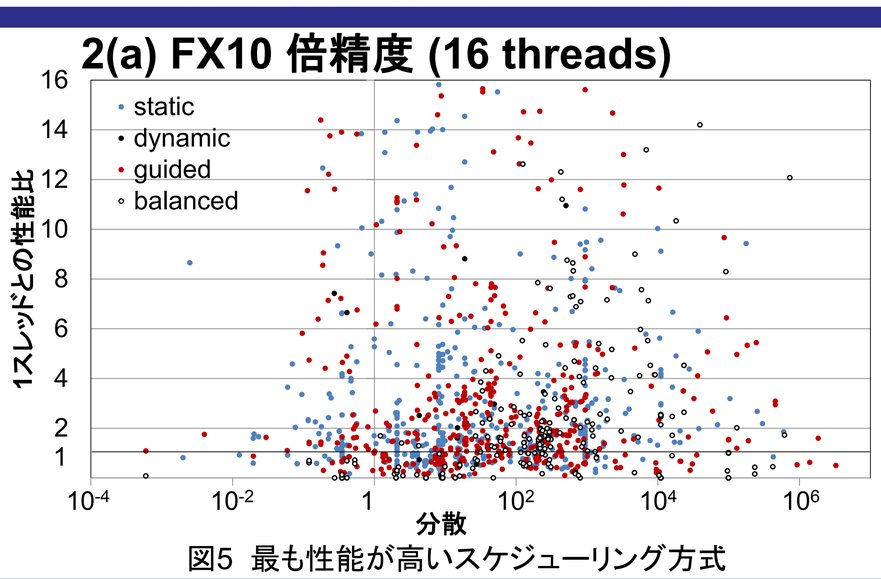

図5　最も性能が高いスケジューリング方式

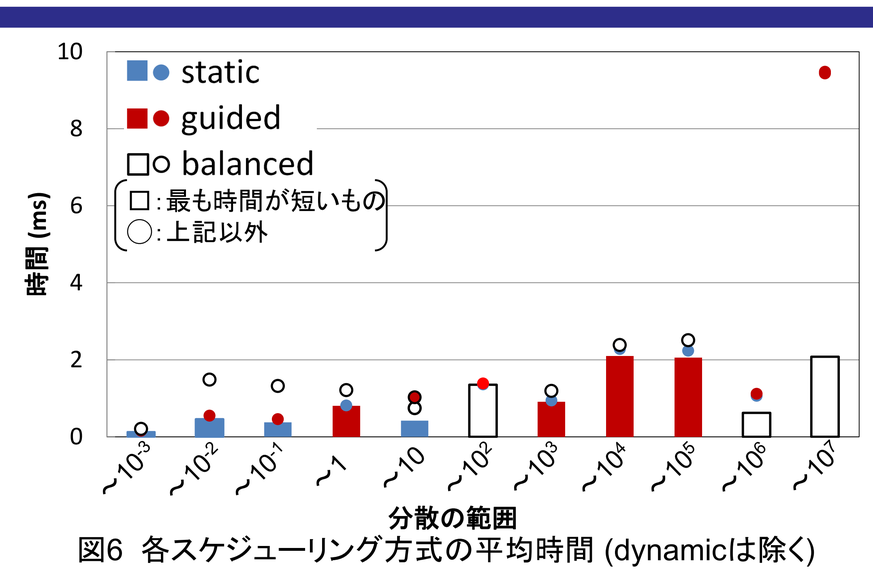

図6　各スケジューリング方式の平均時間 (dynamicは除く)

mix方式

10^6 ≦分散	: balanced
10^2 ≦分散 <10^6	: guided
otherwise	: static

合計時間 (秒)

static	1.79*
dynamic	74.4
guided	2.04
balanced	2.15
mix	1.68
最適値	0.51

-6%

*最も速いスケジューリング方式

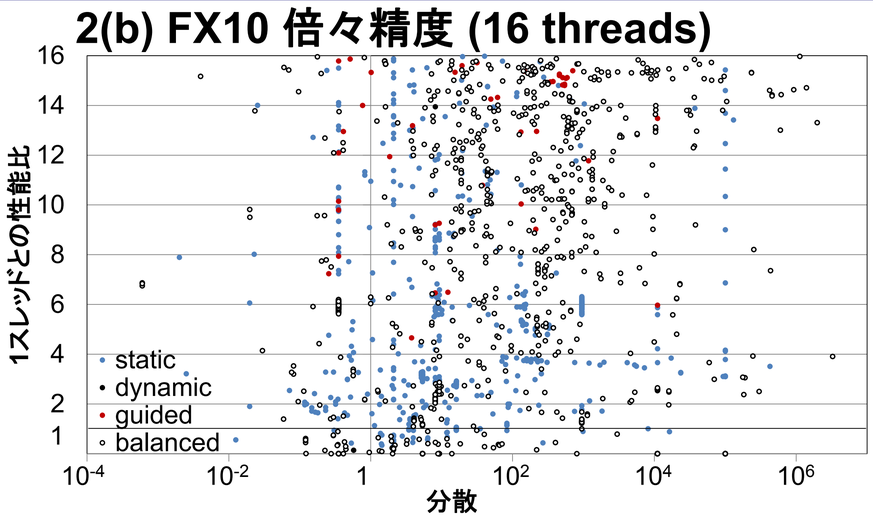

図7　最も性能が高いスケジューリング方式

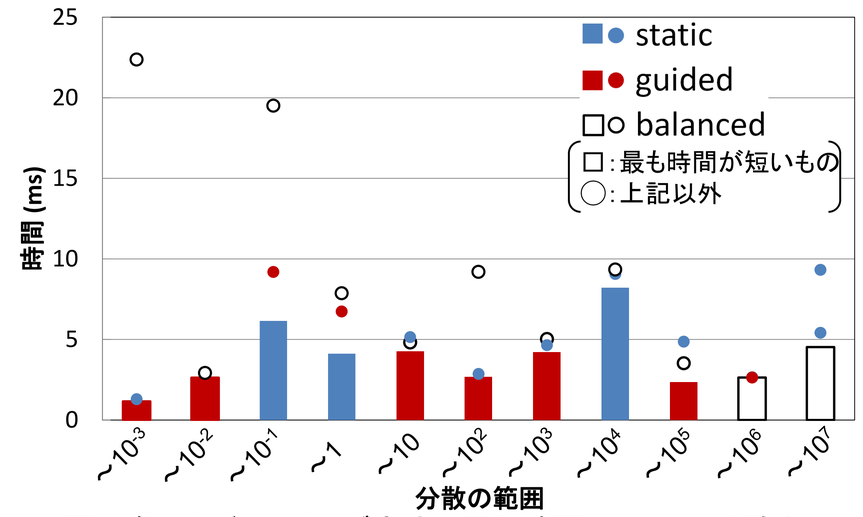

図8　各スケジューリング方式の平均時間 (dynamicは除く)

mix方式

10^6 ≦分散	: balanced
10^2 ≦分散 <10^6	: guided
otherwise	: static

合計時間 (秒)

static	7.88
dynamic	42.9
guided	7.62*
balanced	11.0
mix	6.87
最適値	1.44

-10%

*最も速いスケジューリング方式

Conclusion

- 分散が小さい問題はstaticが有効．分散が大きい問題はguided, balancedが有効．dynamicは有効でない
- 分散からスケジューリング方式を使い分けることで合計時間を他のスケジューリング方式と比べ最大10%削減した
- スレッド数が多いとき(4→16　threads)や，byte/flopの小さい演算(倍→倍々精度)ほど，mix方式の効果は大きい