11-NA01

片桐孝洋（東京大学情報基盤センター）

高精度行列-行列積アルゴリズムにおける並列化手法の開発

1. 概要

行列-行列積に代表される基本線形計算を集約したライブラリBLAS(Basic Linear Algebra Subprograms)は、多くの線形計算で必須の処理である。

従来の数値計算ライブラリは、演算速度は考慮しているが演算精度の考慮が不十分である。解の精度保証が重要な課題となっている。BLASを用いた汎用的な数値計算ライブラリ（例：LAPACK）において、精度保証をする研究は多くない。

BLASを精度保証する研究が、早稲田大学の大石教授のグループにより進められている。本研究は大石グループで開発された高精度行列-行列演算（尾崎の方法[1]）を基本とし、その並列化を中心する以下の研究開発を行う。

i. 「疎行列-密行列積」、もしくは、「疎行列-疎行列」の実装方式の開発
ii. スレッド並列化手法の開発
iii. 分散メモリ型計算機による並列化手法の開発

2. 共同研究体制

- 片桐孝洋（東大・情報基盤センター）
- 尾崎克久（芝浦工業大・システム理工学部）
- 荻田武史（東京女子大・現代教養学部）
- 大石進一（早大・理工学術院・応用数理学科）

3. 尾崎の高精度行列-行列積アルゴリズム[1]

- $AB = \sum_{k=1}^{n_A \cdot n_B} F^{(k)}$ の変形部分

```
Function F = EFT_Mul(A, B)
   [A, n_A] := Split_A
   [B, n_B] := Split_B
   k := 1;
   for i =1: n_A
      for j =1: n_B
         F{ k } := A{ i } * B{ j };   k := k + 1;
 end;  end;  end
```

※Fの和にfaithfulと呼ばれる結果を得るアルゴリズムを用いている。そのため演算結果は、ほぼ最良になることが知られている。

- 行列Aの分解部分

```
Function [D, n_A] = Split_A(A)
   n_A := 0; n := size(A, 2)
   while (norm (A, inf )~=0)
      n_A := n_A + 1;
      μ := max(abs(A),[ ],2);
      τ := 2.^ceil ((log2(u^-1) + log2(n +1))/2);
      t := 2.^ceil((log2( μ)* τ;
      δ_A^(nA) := repmat(t, 1, q);
      A{n_A} := fl((A + δ_A^(nA)) − δ_A^(nA));
      A := fl(A − A{n_A});
end; end
```

- 表記の説明

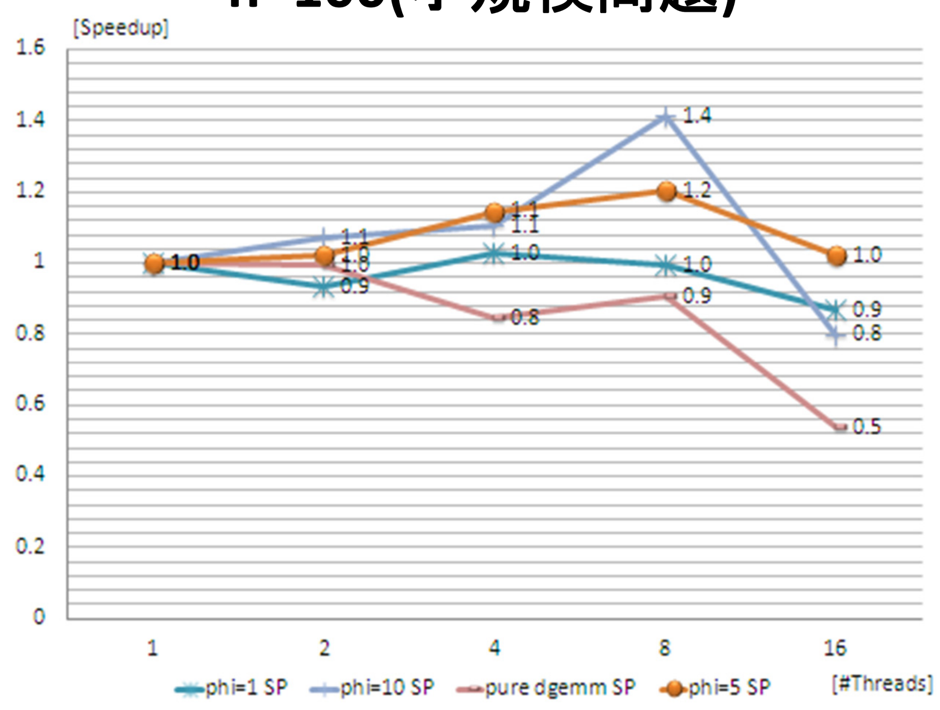
$fl(\cdot)$：　最近点への丸めで計算、　F: 浮動小数点数の集合
A: サイズがm行n列の行列、　B: サイズがn行p列の行列
u: the unit roundoff
　2^{-24} : IEEE 754 binary32 (single precision)
　2^{-53} : IEEE 754 binary64 (double precision)

4. 予備評価（T2Kオープンスパコン1ノード（16コア））

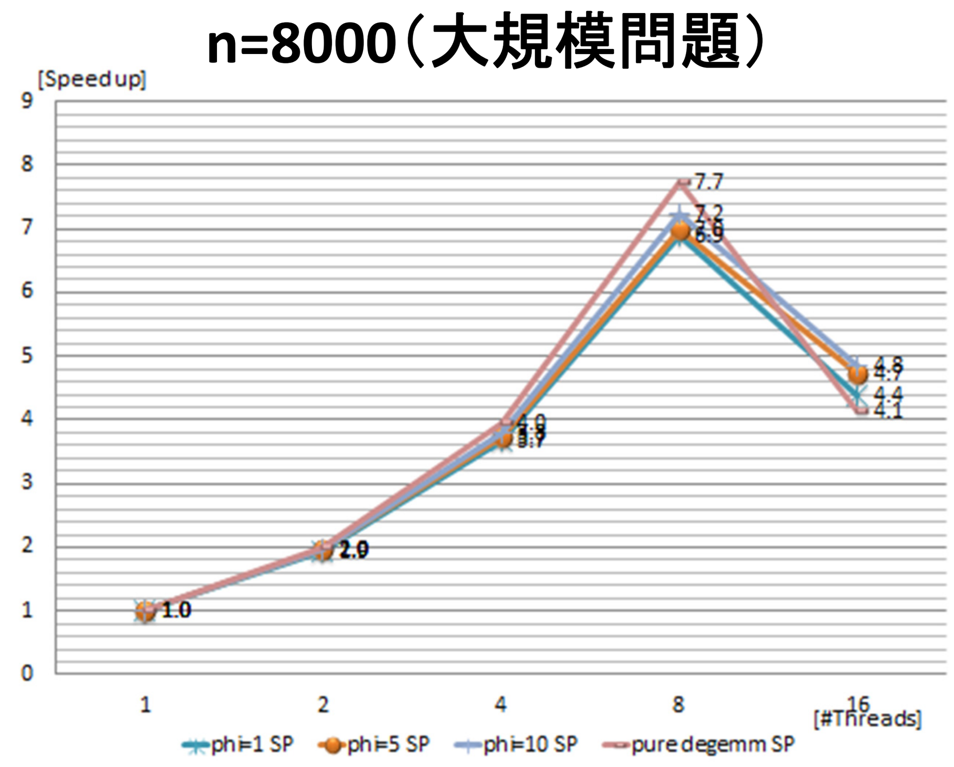
- 日立Cコンパイラ、OpenMP並列化、最適化オプション：-Os -omp
- 後藤 BLAS ver.1.26（スレッド並列版、および逐次版）
- dgemmスレッド並列化　対　逐次dgemmで問題レベル並列化
- 試験行列: A, Bの要素を pow(10,rand()%Φ)　で生成

- dgemm スレッド並列化（台数効果）

n=100(小規模問題)

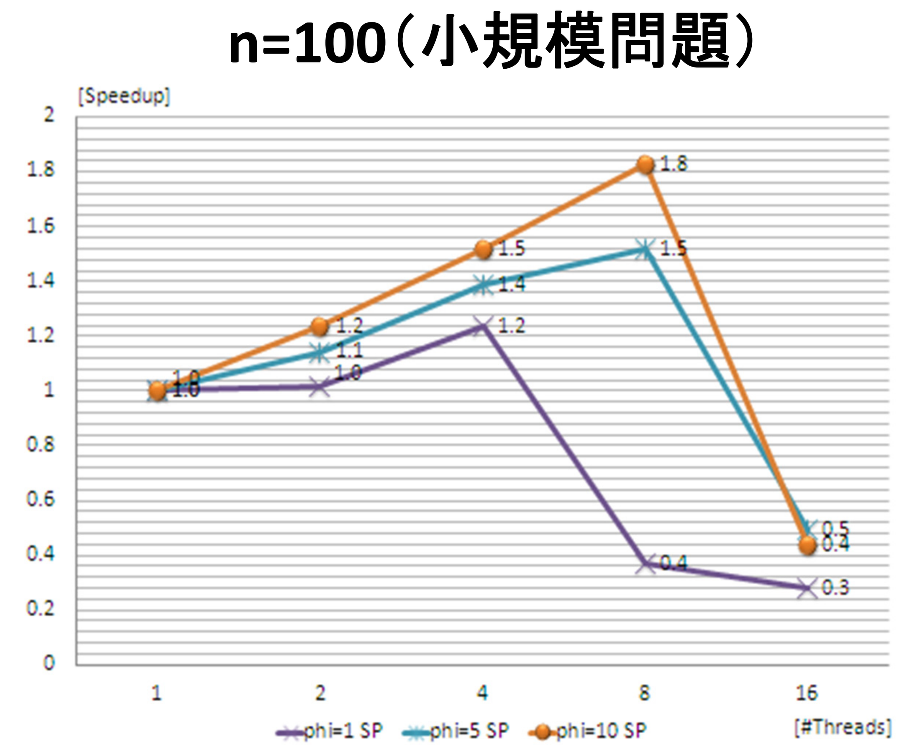


n=8000（大規模問題）

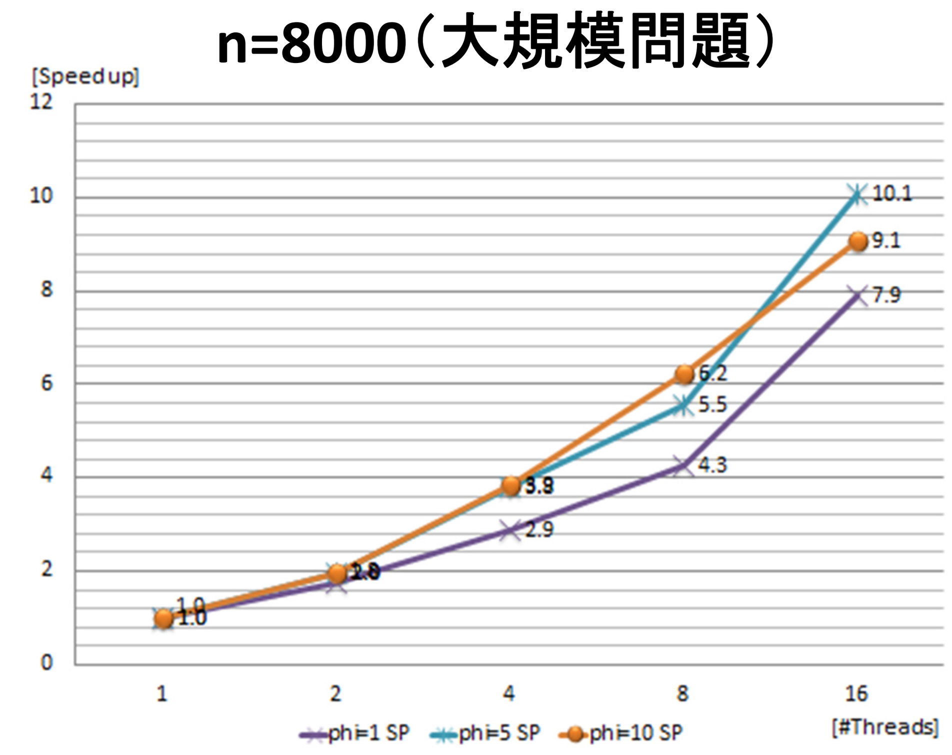


- 逐次dgemm で問題レベル並列化（台数効果）

n=100（小規模問題）



n=8000（大規模問題）



考察

- 小規模問題では＜逐次dgemmで問題レベル並列化＞が高速。
- 大規模問題では、8スレッドまでは＜dgemmスレッド並列化＞が高速。16スレッドでは＜逐次dgemmで問題レベル並列化＞が高速。
- ＜dgemmスレッド並列化＞では、行列の分割数（φの値）は並列性能にあまり影響しない。一方、＜逐次dgemmで問題レベル並列化＞は、行列の分割数が多い（φの値が大きい）時、並列性能がよい（小規模問題）。

5. 今後の予定

- 16スレッド並列dgemmの性能改良。ccNUMA最適化（ファーストタッチ等）の実施
- 問題サイズ、スレッド数に応じた自動チューニングの導入
- 疎行列-疎行列積用の専用ルーチンの開発と導入
- MPIによる分散メモリ並列化方式の開発

参考文献

[1] K. Ozaki, T. Ogita, S. Oishi, S. M. Rump: Error-Free Transformation of Matrix Multiplication by Using Fast Routines of Matrix Multiplication and its Applications, accepted for Numerical Algorithms, Springer.