多倍長環境下における高精度並列線形計算ライブラリの開発

情報科学科　坂口一希　　指導教員：代田 健二

1　はじめに

偏微分方程式の順問題に対する数値解法には様々な方法が提案されているが，それらは最終的に連立一次方程式の求解へと帰着することが多い．すなわち，偏微分方程式の順問題を高精度に解くためには，近似手法とともに連立一次方程式を高精度に解く必要がある．高精度近似解を得るためには，大次元連立一次方程式を解くことも必要になり, 多倍長環境下で解くことを想定すると，大容量メモリとともに並列化が必須となる．多倍長環境下における線形計算ライブラリ [1]，テンプレートによるライブラリ [2] は既に開発されているものの，それらは MPI などの並列化に対応していない．本研究では, 多倍長ライブラリに対応したテンプレート関数並列線形計算ライブラリの作成を目的とする．対象とする線形計算法は，BiCGSTAB 法 [3] である．

2　BiCGSTAB 法

BiCGSTAB 法は BCG 法から派生した反復解法である．$A \in \boldsymbol{R}^{n\times n}$，$\boldsymbol{b} \in \boldsymbol{R}^n$ とする．対象とする連立一次方程式

$$A\boldsymbol{x} = \boldsymbol{b}$$

に対する BiCGSTAB 法のアルゴリズムは次の通りである．

Algorithm 1 BiCGSTAB 法

1: 初期ベクトル $\boldsymbol{x}_0$をとる; $\boldsymbol{r}_0 := \boldsymbol{b} - A\boldsymbol{x}_0$; $\boldsymbol{p}_0 := \boldsymbol{r}_0$;
2: $\boldsymbol{r}_0^\bullet$を適当にとる;
3: $for\ k := 0, 1, 2, \cdots until\ ||\boldsymbol{r}_k|| \le \varepsilon||\boldsymbol{b}||\ do$
4: $begin$
5: $\alpha_k := \frac{\langle \boldsymbol{r}_0^\bullet, \boldsymbol{r}_k\rangle}{\langle \boldsymbol{r}_0^\bullet, A\boldsymbol{p}_k\rangle}$;
6: $\boldsymbol{t}_{k+1} := \boldsymbol{r}_k - \alpha_k A\boldsymbol{p}_k$;　$\omega_k := \frac{(\boldsymbol{t}_{k+1}, A\boldsymbol{t}_{k+1})}{(A\boldsymbol{t}_{k+1}, A\boldsymbol{t}_{k+1})}$;
7: $\boldsymbol{x}_{k+1} := \boldsymbol{x}_k + \alpha_k\boldsymbol{p}_k + \omega_k\boldsymbol{t}_{k+1}$;　$\boldsymbol{r}_{k+1} := \boldsymbol{t}_{k+1} - \omega_k A\boldsymbol{t}_{k+1}$;
8: $\beta_k := \frac{\alpha_k}{\omega_k}\frac{\langle \boldsymbol{r}_0^\bullet, \boldsymbol{r}_{k+1}\rangle}{\langle \boldsymbol{r}_0^\bullet, \boldsymbol{r}_k\rangle}$;　$\boldsymbol{p}_{k+1} := \boldsymbol{r}_{k+1} + \beta_k(\boldsymbol{p}_k - \omega_k A\boldsymbol{p}_k)$
9: end

並列ライブラリとしては MPI を使用することにし，ローカルノード内でのみ計算可能な部分，送受信が必要な部分を精査・検討し，並列化を実施した．

3　数値実験と考察

多倍長計算ライブラリ Exflib[4] を使用し，使用計算機に愛知県立大学情報科学部・並列計算機 09 を用いて数値実験を実施した．扱った問題は，係数行列として次元数 6000，$\gamma = 0.5$ の Toeplitz 行列

$$A = \begin{bmatrix} 2 & 1 & & & & \\ 0 & 2 & 1 & & & \\ \gamma & 0 & 2 & 1 & & \\ & \gamma & 0 & 2 & \ddots & \\ & & \ddots & \ddots & \ddots & 1 \\ & & & \gamma & 0 & 2 \end{bmatrix}$$

を用い, 真の解ベクトルとして, 検証毎にメルセンヌツイスタによる乱数を要素にもつベクトルを生成した．多倍長計算環境としては，検証回数は 30 回とした．10 進 100 桁における計算結果は，表 1 の通りである．また，並列計算の効率を評価する指標 [5]

$$\text{スピードアップ} = \frac{1\ \text{ノード使用時の実行時間}}{N\ \text{ノード使用時の実行時間}},$$

$$\text{並列化効率} = \frac{\text{スピードアップ}}{N}$$

の結果は，図 1，2 の通りである．実験結果より, 並列ノード数が増加するごとに計算時間は短縮され，解に含まれる絶対誤差も許容の範囲内であるので，プログラムは正しく動作していることが示された．一方，スピードアップおよび並列化効率はノード数が増加するごとに悪化する結果となった．並列化効率は，$N = 12$ 以降は 0.5 を下回っており，並列化は十分ではなかった．この原因は通信回数の多さであり，今後は，アルゴリズムの改良も含めた通信回数減少の検討が課題である．

並列ノード数	計算時間 [s]	絶対誤差	反復回数
$N = 1$	754.8133667	1.36×10^{-112}	246
$N = 4$	202.6606667	1.42×10^{-112}	248
$N = 12$	134.5231	1.45×10^{-112}	245
$N = 20$	123.5357333	1.21×10^{-112}	245
$N = 28$	119.3748667	1.05×10^{-112}	247
$N = 36$	117.3807333	1.22×10^{-112}	246

表 1　計算結果 (10 進 100 桁, 平均値)

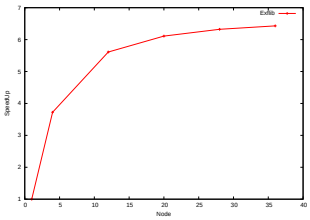

図 1　スピードアップ

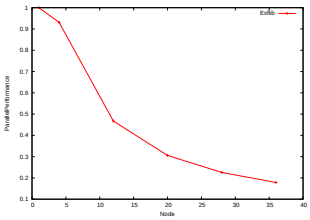

図 2　並列化効率

参考文献

[1] The MPACK; Multiple precision arithmetic BLAS (MBLAS) and LAPACK (MLAPACK)：`http://mplapack.sourceforge.net/`

[2] Template Numerical Toolkit：`http://math.nist.gov/tnt/`

[3] 杉原正顯，室田一雄，線形計算の数理，岩波数学叢書，2009.

[4] Exflib - extend precision floating-point arithmetic library：`http://www-an.acs.i.kyoto-u.ac.jp/~fujiwara/exflib/`

[5] 樫山和夫，西村直志，牛島省，並列計算法入門，日本計算工学会，2003.