行列計算ライブラリインタフェース MATLAB-SILC の実装

Implementation of the MATLAB-SILC Matrix Computation Library Interface

[1,*)] 長谷川 秀彦, [2)] 梶山 民人, [3)] 額田 彰, [4)] 須田 礼仁, [5)] 西田 晃
1) 筑波大学, 2) Universidade Nova de Lisboa, 3) 東京工業大学, 4) 東京大学, 5) 九州大学
[1)]Hidehiko Hasegawa, [2)]Tamito Kajiyama, [3)]Akira Nukada, [4)]Reiji Suda, and [5)]Akira Nishida
1) University of Tsukuba, 2) Universidade Nova de Lisboa, 3) Tokyo Institute of Technology,
4) The University of Tokyo, 5) Kyushu University
* Email: hasegawa@slis.tsukuba.ac.jp

キーワード: MATLAB, 行列計算ライブラリ, アプリケーションプログラムインタフェース, ミドルウェア
Keywords: MATLAB, Matrix Computation Libraries, Application Program Interface, Middleware

1 はじめに

SILC（Simple Interface for Library Collections）[1] は種々の行列計算ライブラリを計算環境やプログラミング言語によらない方法で利用するためのフレームワークであり，図 1 に示すようにクライアント・サーバ方式で実現されている．クライアントであるユーザプログラムは，(1) 入力データの預け入れ，(2) 数式による計算の指示，(3) 計算結果の受け取りの 3 つのステップを通じて SILC サーバの管理する行列計算ライブラリを呼び出す．SILC サーバは与えられた計算指示を適切なライブラリ関数の呼び出しに翻訳して，ユーザプログラムとは独立に計算を実行する．

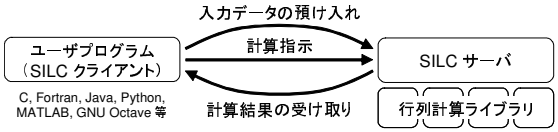

図 1: SILC のシステム構成

対話型数値解析システム MATLAB [3] から SILC の管理する多様な行列計算ライブラリを利用するには，SILC サーバに接続してデータ授受と計算指示を行なうための MATLAB 用 SILC クライアント API（Application Program Interface）を MATLAB 上に設ける必要がある．この API を MATLAB-SILC と呼ぶ．

本報告では，MATLAB-SILC の実現方法，機能，利点，性能，MATLAB との違いについて述べる．

2 MATLAB-SILC の実現方法，機能

MATLAB-SILC は次の 5 つの API 関数から成る．これらの関数は MATLAB の機能拡張ファイルである MEX ファイルとして実現されている．

- `SILC_INIT()`
  SILC サーバへの接続を確立する．
- `SILC_PUT(`*name*`,` *data*`)`
  第 1 引数で与えられた名前を付与して第 2 引数のデータを SILC サーバに預け入れる．
- `SILC_EXEC(`*expression*`)`
  SILC サーバに計算を指示する．引数には計算指示を表す文字列（数式）を与える．
- `SILC_GET(`*name*`)`
  引数で与えられた名前のデータを SILC サーバから受け取って返す．
- `SILC_FINALIZE()`
  SILC サーバとの接続を切る．

例として，2 次元のラプラス方程式を有限差分法で離散化したときに現れる疎な対称正定値行列を係数とする線形方程式 $A\boldsymbol{x} = \boldsymbol{b}$ を解く MATLAB-SILC のユーザプログラムを図 2 に示す．最初に MATLAB の機能を用いて行列 A とベクトル $\boldsymbol{b}$ を作成する（4〜8 行目）．次に MATLAB-SILC の API 関数群を用いて SILC サーバに接続し，線形方程式を解かせる（11〜17 行目）．まず `SILC_PUT` で A と $\boldsymbol{b}$ を SILC サーバに預け入れる．次に `SILC_EXEC` を用いて線形方程式の求解を数式で指示する．数式中の \ は線形方程式の求解を表す SILC の演算子である．次に `SILC_GET` で計算結果を変数 $\boldsymbol{x}$ に受け取る．最後に，データの授受と線形方程式の求解（12〜17 行目）にかかった時間を表示して SILC サーバとの接続を切る（18〜19 行目）．

3 MATLAB-SILC の利点

MATLAB-SILC を用いる利点は，種々の行列計算ライブラリを実行環境によらずに利用できることである．すなわち，SILC サーバとライブラリの実行環境をユーザプログラム（MATLAB）の実行環境から独立して選択でき，SILC サーバの実行環境に関わらず完全に同一のやり方でライブラリを呼び出せる．例えば，図 2 のプログラムはサーバ環境を変えてもそのまま動作する（プログラムの修正は不要である）．サーバ環境としては逐次環境，共有メモリ並列環境，分散並列環境の 3 種が利用できる．ユーザプログラムと SILC サーバの実行環境は同じでも別でもよい．SILC サーバは与えられた計算指示を適当なライブラリ（逐次環境では逐次ライブラリ，並列環境では並列ライブラリ）の呼び出しに翻訳して実行するので，ユーザはサーバ環境やライブラリに応じてプログラムを書き直す必要がない．MATLAB-SILC を用いることで，Windows 上で逐次動作する対話型の MATLAB アプリケーションからリモートの PC クラスタ上で動作する ScaLAPACK を利用したり，ユーザプログラムのデバッグにのみローカルの LAPACK を用いるといったことができる．

```
1:  function lap2d(m)
2:
3:  % 行列 A とベクトル b を作成する
4:  I = sparse(1:m, 1:m, 1);
5:  U = sparse(1:m-1, 2:m, 1, m, m);
6:  D = 4 * I - U - U';
7:  A = kron(I,D) - kron(U,I) - kron(U',I);
8:  b = A * ones(m^2, 1);
9:
10: % MATLAB-SILC を用いて Ax = b を解く
11: SILC_INIT();
12: tic; % 計時開始
13: SILC_PUT('A', A);
14: SILC_PUT('b', b);
15: SILC_EXEC('x = A \ b');
16: x = SILC_GET('x');
17: t = toc; % 計時完了
18: disp(sprintf('%e sec.', t));
19: SILC_FINALIZE();
```

図 2: MATLAB-SILC を用いたユーザプログラムの例

4 数値実験

図 2 のユーザプログラムをノート PC とクラスタ上の SILC サーバで実行した．ユーザプログラムの実行環境は ThinkPad T61（Intel Core2 Duo 1.8 GHz, Windows Vista Business（64 ビット），MATLAB R2007b），SILC サーバの実行環境は 8 コアの PC クラスタ（4-way AMD Opteron 2.2 GHz × 2 ノード, SuSE Linux 9.3（x86_64），GCC 3.3.5，LAM/MPI 7.1.1），接続は Gigabit Ethernet とした．線形方程式の求解には反復解法ライブラリ Lis [2] の共役勾配法（CG 法）を 8 プロセス，倍精度，前処理なしで用いた．

行列 A の次数が $N = 10^6$ のとき，ユーザプログラムの求解部分（12〜17 行目）の実行時間は 25.1 秒，そのうち CG 法の計算時間は 23.1 秒（92.2%）であった（反復回数 2,180）．残りの実行時間はデータ授受にかかる通信コストであり，内訳はクライアント・サーバ間通信が 1.35 秒（データ量 76.2 MB，スループット 451.6 Mbps），サーバプロセス間通信が 0.62 秒であった．

通信時間は授受するデータ量に比例するのに対して反復解法の計算時間は概ね反復回数 K と行列の非零要素数 Z に比例する．データ量は主として Z に依存して決まるので，通信時間と計算時間の比は K に比例する．したがって反復回数の多い（解きにくい）問題ほど実行時間に占める通信時間の割合は小さくなる．

また，サーバ側で Lis の CG 法を 1 プロセスで実行した場合，ユーザプログラムの求解部分の実行時間は 169.6 秒であった．一方，MATLAB の組み込み関数 `pcg`（前処理なし CG 法）を 1 スレッドで用いると 812.2 秒かかる．このように同じ解法であっても MATLAB-SILC を通じて高速なライブラリを利用する方が通信コストを加味しても実行時間の面で有利なことがある．

5 MATLAB における並列化との違い

MATLAB は（a）逐次環境および共有メモリ並列環境，（b）分散並列環境をサポートしている．前者のサポートは MATLAB 本体に組み込まれており，例えば LU 分解を行なう MATLAB の組み込み関数 `lu` は逐次または共有メモリ並列（マルチスレッド）で動作する．（b）のサポートは Parallel Computing Toolbox と MATLAB Distributed Computing Server を通じて提供されている [4]．ユーザプログラムの書き方は（a）の場合と（b）の場合とで異なる．（a）の環境向けに書いたプログラムを（b）の環境で並列に動作させるには，並列ループや分散配列（distributed array）を用いてユーザ自身がプログラムを並列化しなければならない．

一方，MATLAB-SILC では SILC サーバの実行環境に関わらず同じ数式で計算を指示する．ユーザはサーバ環境を逐次環境から並列環境に変更するだけで自動的に並列計算の恩恵を享受できる．

MATLAB ではデータ並列とタスク並列の両方の並列化手法をサポートしている．ただし，これらの並列化手法を用いてプログラムを並列化するのはユーザの責任である．MATLAB-SILC は今のところデータ並列のみをサポートしており，すべての行列計算がサーバ側で自動的にデータ並列により並列化される．また，`x1 = A \ b1; x2 = A \ b2` のようなデータ依存性のない 2 つの数式をタスク並列性に基づいて自動的に並列化することも今後の課題として検討している．

6 おわりに

SILC は通信コストと引き換えに多様な行列計算ライブラリや計算環境を柔軟に組み合わせられる便利さを提供するツールである．本報告では MATLAB 用の SILC クライアント API である MATLAB-SILC の概要について述べた．また，通信コストを考慮しても，より高速なライブラリを利用することによって実行時間を短縮できるメリットの方が大きいことを数値実験で確かめた．より現実味のある規模の大きな問題に MATLAB-SILC を適用して道具としての実用性を確かめることは今後の課題の 1 つである．

謝辞 本研究のきっかけと計算機資源の利用機会を与えてくださった早稲田大学理工学部の大石進一教授，東京女子大学文理学部の荻田武史講師に感謝いたします．本研究の一部は，科学技術振興機構（JST）戦略的創造研究推進事業（CREST）「大規模シミュレーション向け基盤ソフトウェアの開発」プロジェクトにおいて実施した．

参考文献

[1] T. Kajiyama *et al.* SILC: Flexible and environment independent interface for matrix computation libraries. In *PPAM 2005, LNCS 3911*, pp. 928–935, 2006. http://www.ssisc.org/silc/.

[2] 小武守, 他. 反復法ライブラリ向け 4 倍精度演算の実装と SSE2 を用いた高速化．情報処理学会論文誌：コンピューティングシステム, 1(1):73–84, 2008. http://www.ssisc.org/lis/.

[3] MATLAB. http://www.mathworks.com/.

[4] MATLAB Parallel Computing Toolbox 3.3. http://www.mathworks.com/products/parallel-computing/.