ISSN 0286-7419

TOHOKU
UNIVERSITY

東北大学
サイバーサイエンスセンター

大規模科学計算システム広報

SENAC

Vol.48 No.2 2015–4

Cyberscience
Center

Supercomputing System
Tohoku University
www.ss.cc.tohoku.ac.jp

大規模科学計算システム関連案内

＜大規模科学計算システム関連業務は、サイバーサイエンスセンター本館内の情報部情報基盤課が担当しています。＞

http://www.ss.cc.tohoku.ac.jp/

階	係・室名	電話番号(内線)* e-mail	主なサービス内容	サービス時間 平 日
一階	利用相談室	022-795-6153(6153) sodan05@cc.tohoku.ac.jp 相談員不在時 022-795-3406(3406)	計算機利用全般に関する相談 大判プリンタ、利用者端末等の利用	8:30～17:15 8:30～21:00
	利用者談話室	(3444)	各センター広報の閲覧	8:30～21:00
	展 示 室 (分散コンピュータ博物館)		歴代の大型計算機等の展示	9:00～17:00
	可視化機器室	(3428)	三次元可視化システムの利用	9:00～21:00
三階	総務係	022-795-3407(3407) som@cc.tohoku.ac.jp	総務に関すること	8:30～17:15
	会計係	022-795-3405(3405) kaikei@cc.tohoku.ac.jp	会計に関すること、負担金の請求に関すること	8:30～17:15
	共同研究支援係	022-795-6252(6252) rs-sec@cc.tohoku.ac.jp	共同研究、計算機システムに関すること	8:30～17:15
	共同利用支援係 (受 付)	022-795-3406(3406) 022-795-6251(6251) uketuke@cc.tohoku.ac.jp	利用手続き、利用相談、講習会、ライブラリ、アプリケーションに関すること	8:30～17:15
	ネットワーク係	022-795-6253(6253) net-sec@cc.tohoku.ac.jp	ネットワークに関すること	8:30～17:15
四階	研究開発部	022-795-6095(6095)		
五階	端末機室	(3445)	PC 端末機(X 端末)	

＊(　) 内は東北大学内のみの内線電話番号です。青葉山・川内地区以外からは頭に 92 を加えます。

本誌の名前「SENAC」の由来

昭和 33 年に東北地区の最初の電子計算機として、東北大学電気通信研究所において完成されたパラメトロン式計算機の名前で SENAC-1(SENdai Automatic Computer-1)からとって命名された。

［大規模科学計算システム］

スーパーコンピュータ SX-ACE の利用法

情報部情報基盤課　共同利用支援係　共同研究支援係
サイバーサイエンスセンター　スーパーコンピューティング研究部

1章　はじめに

本センターはスーパーコンピュータ SX-ACE の運用を 2015 年 2 月から開始しています。本稿では、SX-ACE システムでのプログラミング利用ガイドとして、プログラムの作成からコンパイル、実行等の使い方と利用負担金についてご紹介します。

■　システムの特徴

■ ハードウェア

本センターの SX-ACE システムを図 1 に示します。SX-ACE（日本電気㈱製）は、前システム SX-9 と同じくベクトル並列型スーパーコンピュータを継承しています。今回導入した SX-ACE のシステムは全 2,560 ノードで構成され、1 ノードあたり理論最大演算性能 276GFLOPS、最大ベクトル性能 256GFLOPS の世界初のマルチコア(4 コア)ベクトルプロセッサを 1 基搭載し、システム全体では約 707TFLOPS となります。主記憶は 1 ノードあたり 64GB を搭載し、256GB/s という高いメモリバ ンド幅でプロセッサと接続されることで、高い演算性能とメモリ性能の最適化を実現しています。

図 1 本センターの SX-ACE システム

マルチコアベクトルプロセッサ 1CPU（4 コア）あたり 276 GFLOPS の理論最大演算性能

1 ノードあたり 64GB の共有メモリ

データ転送性能（メモリバンド幅）1CPU あたり 256 GB/s

■ ベクトル計算機

ベクトル計算機とは、ベクトル演算を行う専用のハードウェアを持つコンピュータです。ベクトル演算は、ループ中で繰り返し処理されるような配列データの演算に対して一括して演算を実行するため、高速に演算することができます。

ベクトル計算機は科学技術用の数値計算に適しており、大量のデータを繰り返し処理するような大規模計算に向いていると言われています。本センターでは、流体解析や気象解析、電磁界解析をはじめとする大規模シミュレーションにご利用いただいています。

■ ソフトウェア

UNIX に準拠した SUPER-UX オペレーティングシステムを継承し、ベクトル処理、並列処理に対応した大規模プログラムの実行が可能です。また、SX-ACE に最適化された科学技術計算ライブラリ ASL も引き続きご利用できます。

プログラミング言語

プログラミング言語は Fortran と C/C++を用意しています。それぞれ自動ベクトル化機能、自動並列化機能を有していますので、既存のプログラムを修正することなくベクトル化、並列化することができます。OpenMP や MPI による並列化プログラムも利用可能です。

並列化プログラミング

並列化の方法として表 1 の 3 種類が利用できます。

自動並列化は、単一コアで動作する逐次処理プログラムを、コンパイラが並列化可能な箇所を自動的に判断して並列化します。プログラムを改めて書き直す必要はなく、お持ちのプログラムをそのまま利用することができます。

OpenMP は、自動並列化と同じく逐次処理プログラムを並列化します。並列化の判断は自動ではなくユーザが行います。ソース中の並列化したい箇所に並列化指示行を追加します。

MPI は、メッセージ交換用ライブラリによりプロセッサ間通信を行います。データの分割、処理方法等の並列処理手順を明示的に記述した MPI プログラムを作成する必要があります。

ノード内では自動並列化あるいは OpenMP による並列処理を、複数のノードを利用する場合は MPI による並列処理を用います。したがって、自動並列化や OpenMP は 4 並列までのノード内並列処理ができます。それ以上の並列数で実行する場合は、複数のノードを必要とするので MPI による並列処理を行います。また、大規模なメモリを必要とするプログラムも、MPI を利用して複数のノードで実行する必要があります。

本センターでは最大 1,024 ノード[1]を利用した 4,096 並列での実行が可能です。

自動並列化と OpenMP によるノード内並列処理手順は 3 章で、MPI による並列処理手順は 4 章で、プログラムの実行方法は 5 章で紹介します。

表 1 並列化の種類、特徴、並列数、メモリ量

並列化の種類	逐次処理プログラムの並列化	並列化の指示	最大並列数	最大メモリ量
自動並列化	そのままで可能	自動 (コンパイラ)	4 並列 ノード内	60GB
OpenMP	指示行を追記することで可能	手動 (ユーザ)	4 並列 ノード内	60GB
MPI	MPI 用プログラムに変更する	MPI ライブラリを用いて プログラミング (ユーザ)	4,096 並列 1,024 ノード	60GB ×ノード数

[1] 513 ノード以上の利用は、2015 年後半を予定しています。

2 章　システムの構成

本センターでは、大型計算機として SX-ACE システム 2,560 ノードと並列コンピュータシステム（LX 406Re-2）68 ノードの 2 つのシステムをサービスしています（図 2）。

並列コンピュータはスーパーコンピュータのフロントエンドサーバの役割も担っています。

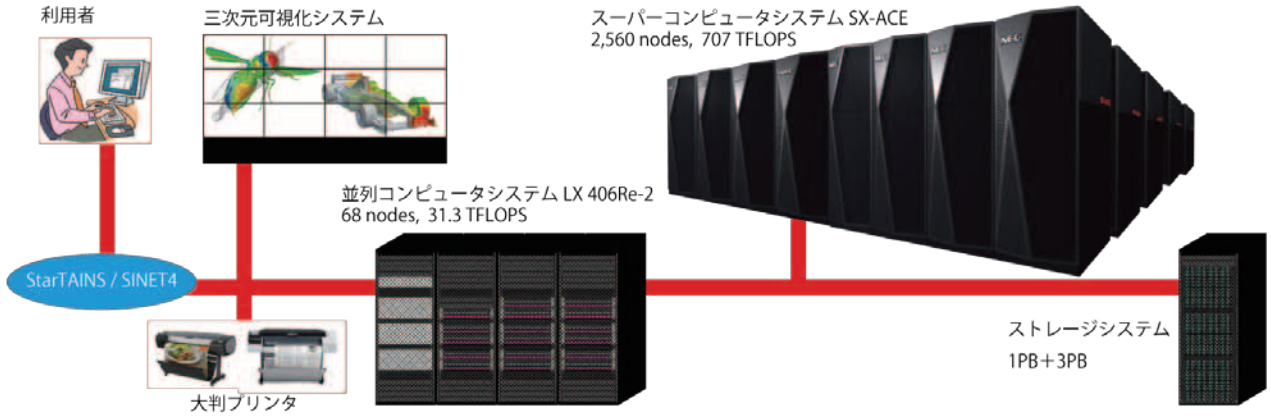


図 2 システム構成図

3 章　プログラミング　～逐次処理、共有メモリ並列処理 ～

本章では、単一のコアで実行する逐次処理と、自動並列化および OpenMP による共有メモリ並列処理について利用手順を紹介します。

MPI による並列化プログラミング手順については、4 章で紹介しますが、コンパイルコマンドに違いがある以外は、同じ手順ですのでこの章と合わせてご覧ください。

■　フロントエンドサーバ（並列コンピュータ）へのログイン

SX-ACE 用のプログラミングは、並列コンピュータ上で全ての作業を行えるようにしています。ソースファイル作成、コンパイル、実行など一連の作業が可能です。SX-ACE システムのパフォーマンスを演算に専念させるために、フロントエンドサーバとの 2 段構成としています（図 3）。

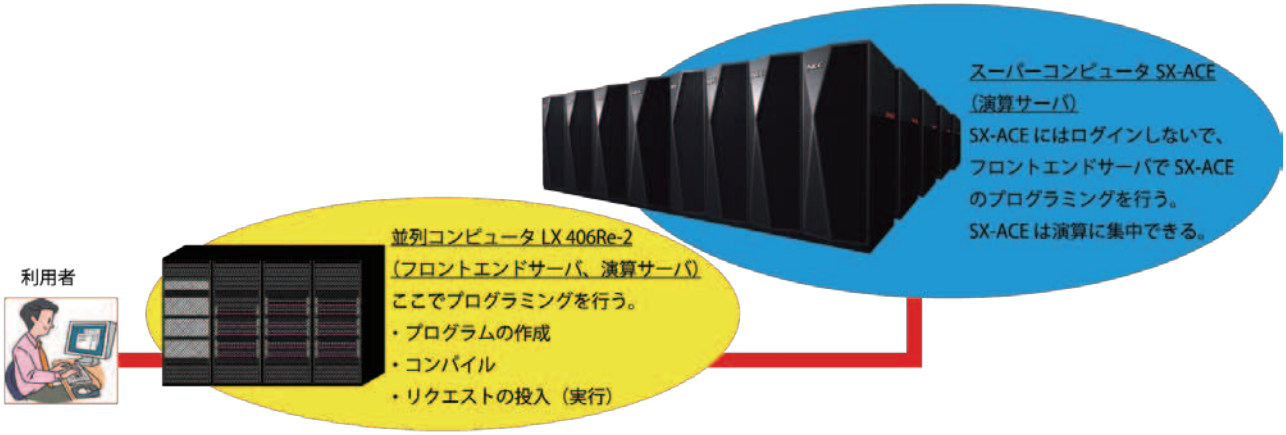


図 3 フロントエンドサーバと演算サーバ

■ ログイン

作業を行うため並列コンピュータにログインします。リモート接続は、ssh コマンドまたは SSH 対応リモート接続ソフト[2]をご利用ください。

並列コンピュータの OS は Linux です。公開鍵暗号方式による認証のみ利用できます[3]。アカウント希望の場合は、共同利用支援係に利用申請し利用者番号と初期パスワードを発行してもらいます。

並列コンピュータへの初回ログイン時には公開鍵と秘密鍵のペアを作成する必要があります。鍵ペアの作成方法については本誌 105 ページの「SSH アクセス認証鍵生成サーバの利用方法」をご参照ください。

なお、他人名義の利用者番号でのシステム利用は禁止します。パスワード、秘密鍵、パスフレーズの使い回しは、不正アクセスのリスク（不正ログイン、クライアントのなりすまし、暗号化された通信の暴露、他サーバへの攻撃等）が非常に高く、大変危険です。利用者登録を行うことによる年間維持費等は発生しませんので、利用される方はそれぞれで利用申請をお願いいたします。

並列コンピュータホスト名

```
front.cc.tohoku.ac.jp
```

リスト 1 ssh コマンドによる接続例

```
localhost$ ssh -i ~/.ssh/id_rsa 利用者番号@front.cc.tohoku.ac.jp
Enter passphrase for key '/home/localname/.ssh/id_rsa':パスフレーズを入力
（初回接続時のメッセージ）： yes を入力
front1$（コマンド待ち状態）
```

暗号鍵は複数の端末や他人と共有してはいけません。また、メールに暗号鍵を添付したり USB メモリにコピーしたりすると不正アクセスのリスクとなります。並列コンピュータにログインする端末を追加する場合は、その端末で新規に鍵ペアを作成し、authorized_keys に追加登録してください。

ログイン端末の追加方法

ログインする端末を追加する場合は、追加する端末で鍵ペアの作成を行います。必ずパスフレーズの設定を行って鍵を作成して下さい。作成した公開鍵を既に並列コンピュータにログイン出来る端末に転送します。公開鍵ですので転送方法はメール本文に記載しても構いません。

作成した公開鍵の内容を利用者のホームディレクトリのファイル（~/.ssh/authorized_keys）に追記します。接続元が Linux、OS X の場合の鍵の作成方法をリスト 2 に示します。

リスト 2 公開鍵と秘密鍵の作成方法

```
【利用する端末で鍵ペアを作成】
localhost $ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/localname/.ssh/id_rsa):（ファイル名を指定）

Enter passphrase (empty for no passphrase):（必ずパスフレーズを設定）
Enter same passphrase again:（同じパスフレーズを入力）

指定した場所(/home/localname/.ssh)に鍵ペア(暗号鍵:id_rsa 公開鍵:id_rsa.pub)が生成される

【作成した公開鍵を既に並列コンピュータにログイン出来る端末に転送】

【作成した公開鍵を並列コンピュータに転送】
localhost $ scp /home/localname/.ssh/id_rsa.pub 利用者番号@front.cc.tohoku.ac.jp:~
```

[2] Windows であれば、TeraTerm 等のフリーソフトが利用できます。

[3] パスワード認証方式は 2015 年 4 月 13 日で廃止しました。

```
（パスフレーズを入力）

【公開鍵を追記登録】
front1 $ cat ~/id_rsa.pub >> ~/.ssh/authorized_keys
front1 $ exit
```

~/.ssh/authorized_keys を削除すると、全ての暗号鍵からのログインが出来なくなります。センターではセキュリティインシデントに対する緊急対応として、全ユーザの~/.ssh/authorized_keys を削除する場合があります。

■ パスワードの変更

パスワードの変更は passwd コマンドで行います。パスワードの変更方法をリスト 3 に示します。
コマンドを実行すると、並列コンピュータ、可視化サーバ、プリンタサーバ、およびファイル転送サーバのログインパスワードが変更されます。入力したパスワードは表示されません。

リスト 3 パスワードの変更方法

```
front1 $ passwd
ユーザー 利用者番号 のパスワードを変更。
Enter login(LDAP) password: （現在のパスワードを入力）
新しいパスワード: （新しいパスワードを入力）
新しいパスワードを再入力してください: （新しいパスワードを入力）
LDAP password information changed for 利用者番号
passwd: 全ての認証トークンが正しく更新できました。
```

■ ログインシェルの確認と変更

ログインシェルの確認と変更は fchsh コマンドで行います。ログインシェルの確認方法と変更方法をリスト 4 に示します。
ログインシェルの変更が SX-ACE と LX 406Re-2 に反映されるまで 15 分程度かかります。また、SX-ACE で利用可能なシェル以外をログインシェルに設定した場合、リクエストが正常に実行されなくなります。

リスト 4 ログインシェルの確認方法と変更方法

```
front1 $ fchsh（ログインシェルの確認）
Enter Password:（パスワードを入力）
loginShell: /bin/tcsh（現在のログインシェルが表示される）

front1 $ fchsh /bin/bash（ログインシェルを/bin/bash に変更）
Enter Password:（パスワードを入力）
Changed loginShell to /bin/bash（ログインシェルが変更された）
```

■ ホームディレクトリ

ホームディレクトリは、プログラムファイル等を置く自分専用のディスク領域です。ディレクトリ名は、/uhome/利用者番号です。利用者番号作成時の容量制限は 1TB です。ファイル容量の追加申請によりディスク領域を増やすことも可能です。ホームディレクトリはスーパーコンピュータシステムと並列コンピュータシステムで共有しています。

ホームディレクトリ

```
/uhome/利用者番号
```

■　プログラムの作成とコンパイル

■ ソースファイルの作成

並列コンピュータ上でソースファイルを作成するためのエディタは、emacs または vi が一般的です。

研究室等のサーバ、パソコンから転送する場合は、SSH 対応のファイル転送ソフト[4]で並列コンピュータのホームディレクトリに転送してください。その際、ソース(テキスト)ファイルは ASCII モードで転送します。

Windows 環境で作成したソースコードの改行コードを Linux 用にするためには dos2unix コマンドを利用します。また、並列コンピュータの文字コードは UTF-8 に設定されていますので、ソースコード内で日本語を利用する際は文字コードを合わせてください。並列コンピュータでは文字コードの変換に nkf コマンドが利用できます(リスト 5)。

リスト 5 dos2unix コマンドと nkf コマンドの利用例

```
front1 $ dos2unix source.f90                  (改行コードを Linux に合わせる)
front1 $ nkf -w --overwrite source.f90        (文字コードを UTF-8 に変換)
```

■ コンパイラ

SX-ACE 用に最適化されたコンパイラ Fortran および C/C++を用意しています。両言語とも並列コンピュータ上でコンパイルとリンクを行います。

● Fortran コンパイラの仕様

FORTRAN90/SX（日本電気製）
ISO/IEC 1539-1:1997 準拠
自動ベクトル化、自動並列化、OpenMP 対応

● C/C++コンパイラの仕様

C++/SX（日本電気製）
ISO/IEC 9899:1999 C 準拠
ISO/IEC 14882:2003 C++ 準拠
自動ベクトル化、自動並列化、OpenMP 対応

■ コンパイルを行う

通常のコンパイルと同様に、それぞれの言語用コマンドでコンパイルします。この項目では単一コアで実行する逐次処理と、自動並列化または OpenMP による並列処理のコンパイル手順について解説します。

Fortran プログラムのコンパイル

sxf90 コマンドでコンパイルします。利用したい機能があれば適当なオプションと、ソースファイル名を指定します。

ソースファイルの拡張子は、自由形式(フリーフォーマット)なら.f90 か.F90、固定形式(7カラム目から記述)なら.f か.F を付けます(表 3)。

並列化コンパイルは、ここで並列数を意識する必要はありません。実行する時点で希望する並列数を環境変数で指定します。ノード内並列数は自動並列の場合は環境変数 F_RSVTASK で指定し、OpenMP 並列の場合は環境変数 OMP_NUM_THREADS で指定します。詳細は 5 章で説明します。

[4] Windows であれば、WinSCP 等のフリーソフトがあります。

● **コンパイル【逐次処理】**

```
front1 $ sxf90 オプション ソースファイル名
```

● **コンパイル【自動並列化】**

```
front1 $ sxf90 -Pauto オプション ソースファイル名
```

・OpenMP プログラムなら、-Pauto の箇所を-Popenmp にします。

表 2 は、主なオプションです。詳細は、sxman sxf90 コマンドでご覧ください。

表 2 主なオプション

オプション	機能
-Pauto	自動並列化機能を利用する。
-Popenmp	OpenMP 対応機能を利用する。
-R5	ベクトル化／並列化状況を表示した編集リストを出力する。
-ftrace	簡易性能解析機能を利用する。
-eC	配列要素参照時、添字の値が範囲内であるか検査する（バウンズチェック）。

表 3 拡張子とフォーマット

拡張子	フォーマット
.f90 .F90	自由形式
.f .F	固定形式

C/C++プログラムのコンパイル

sxcc コマンドで C プログラムを、sxc++コマンドで C++プログラムをコンパイルします。利用したい機能があれば適当なオプションと、ソースファイル名を指定します。

並列化コンパイルは、ここで並列数を意識する必要はありません。実行する時点で希望する並列数を環境変数で指定します。ノード内並列数は自動並列の場合は環境変数 F_RSVTASK で指定し、OpenMP 並列の場合は環境変数 OMP_NUM_THREADS で指定します。詳細は 5 章で説明します。

● **コンパイル【逐次処理】**

```
front1 $ sxcc  オプション ソースファイル名
front1 $ sxc++ オプション ソースファイル名
```

● **コンパイル【自動並列化】**

```
front1 $ sxcc -Pauto  オプション ソースファイル名
front1 $ sxc++ -Pauto オプション ソースファイル名
```

・OpenMP プログラムなら、-Pauto の箇所を-Popenmp にします。

表 4 は、主なオプションです。詳細は、sxman sxcc コマンド、または sxman sxc++コマンドでご確認ください。

表 4 主なオプション

オプション	機能
-Pauto	自動並列化機能を利用する。
-Popenmp	OpenMP 対応機能を利用する。
-size_t64	4G バイト以上の配列、構造体を利用する。
-Rfmtlist	ベクトル化／並列化状況を表示した編集リストを出力する。
-ftrace	簡易性能解析機能を利用する。

表 5 拡張子とフォーマット

拡張子	言語
.c	C
.cpp	C++

逐次処理、自動並列化または OpenMP により並列化されたプログラムは、1 ノードのみ利用できます。複数ノードを利用する場合は MPI による並列化が必要です。MPI については次章をご参照ください。

4 章 プログラミング ～ MPI 並列処理 ～

本章では、MPI による並列化プログラミング手順について紹介します。基本的な手順は前章と同じですので、コンパイルコマンドの異なる点について紹介します。

■ コンパイルを行う

MPI プログラムは、MPI 用コマンド sxmpif90、sxmpic++、sxmpicc コマンドでコンパイルします。

MPI 並列 Fortran プログラムのコンパイル

MPI 並列の Fortran コードは sxmpif90 コマンドでコンパイルします。利用したい機能があればオプションと、ソースファイル名を指定します。

```
front1 $ sxmpif90 オプション ソースファイル名
```

・オプションは、sxf90 コマンドと共通です。sxman sxf90 コマンドでご覧ください。

MPI 並列 C/C++プログラムのコンパイル

MPI 並列の C コードは sxmpicc コマンドで、MPI 並列の C++コードは sxmpic++コマンドでコンパイルします。利用したい機能があれば適当なオプションと、ソースファイル名を指定します。

```
front1 $ sxmpicc  オプション ソースファイル名
front1 $ sxmpic++ オプション ソースファイル名
```

・オプションは、sxcc コマンド、sxc++コマンドと共通です。詳細は sxman sxcc コマンド、sxman sxc++コマンドでご確認ください。

■ ハイブリッド並列

MPI と自動並列、または MPI と OpenMP を組み合わせたハイブリッド並列と呼ばれる並列化手法があります(図 4)。ノード間は MPI 並列化し、ノード内は自動並列化や OpenMP による並列化する形をとります。

例えば、512 ノードを使ったハイブリッド並列を考えた場合、512 プロセスで実行する MPI プログラムを作成します。さらに、自動並列化オプション-Pauto をつけコンパイルすると、MPI と自動並列のハイブリッド並列用オブジェクトが作成されます。ノード間は MPI の 512 並列、ノード内は自動並列化の 4 並列の、計 2,048 並列で実行できます。実行方法についての詳細は次章で説明します。

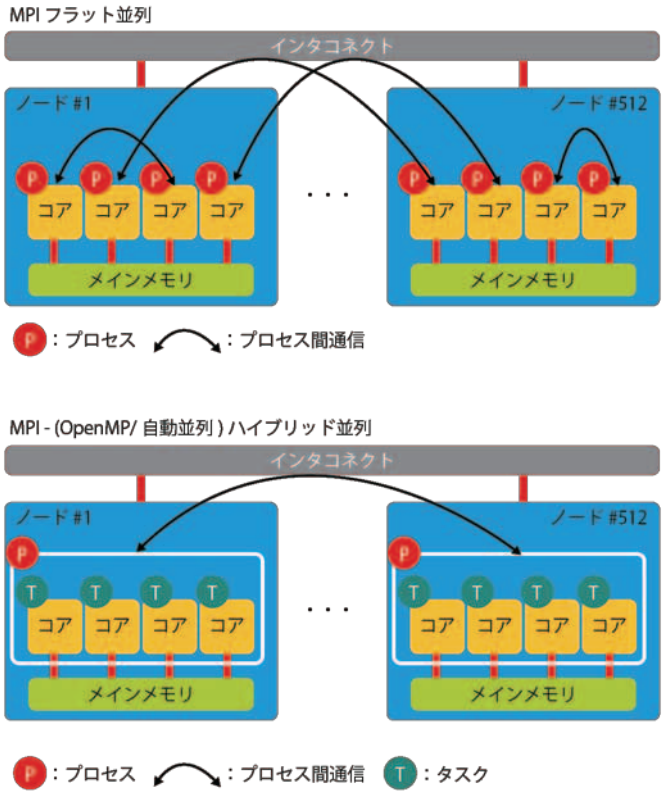


図 4 MPI フラット並列とハイブリッド並列

```
front1 $ sxmpif90 -Pauto オプション ソースファイル名
```

• ノード内が OpenMP 並列なら、-Popenmp オプションを使います。

5 章　バッチリクエスト

■　バッチリクエストの概要

フロントエンドサーバでコンパイル作業を行って作成したプログラムの実行は、バッチリクエストと呼ばれる方法で計算機に実行を依頼します。本センターではバッチリクエストの処理に NQS Ⅱを採用しています。以下では NQS Ⅱによるバッチリクエストについて説明します。

始めに NQS Ⅱにプログラムの実行を依頼するため、実行の手続きを書いたバッチリクエストファイルを作成します。このファイルを NQS Ⅱに投入することで、プログラムの実行が可能になります（図 5）。利用者は実行する計算機とノード数を指定して投入します。

順番が回ってくると NQS Ⅱは自動的にリクエストを実行します。リクエストはバックフィルスケジュール機能で実行順序が制御されています。利用者が実行時間を実行時間制限の規定値以下に指定すると、利用ノード数が少ない場合ノードの空き状況によっては実行開始時間が早くなることがあります。

リクエスト投入後は、リクエストの状態の確認、また投入済みのリクエストをキャンセルすることも可能です。プログラムの実行が終了するとリクエストは NQS Ⅱの情報から消え、標準出力ファイルと標準エラー出力ファイルが出力されます。

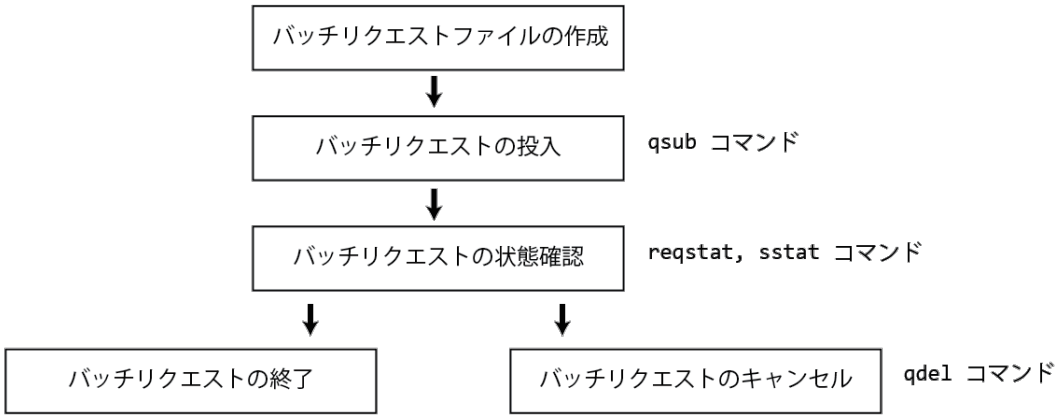


図 5 バッチリクエストの流れ

■　バッチリクエストファイルの作成

プログラムの実行手続きを、通常のシェルスクリプトと同じ形式で記述します。csh スクリプトと sh スクリプト、どちらでも記述できます(以降、解説は csh スクリプトで記述します)。適当なファイル名を付け作成します。以降ではバッチリクエストファイル名を run.csh とします。

基本的に必要となるのは、実行するマシンとノード数の指定、ホームディレクトリから作業ディレクトリへ移動、プログラムの実行、です。他に環境変数の指定、ファイルの操作コマンド等があれば適切な箇所に手続きを記述します。

SX-ACE で実行する場合の利用形態と必須オプションを表 6 に示します。通常の利用の場合、-q の後に sx を指定し、-b の後に利用ノード数を指定します。

実行時間の設定は、-l elapstim_req=hh:mm:ss で設定します。通常利用の場合、実行時間制限の規定値でリクエストは自動的に終了します。実行時間が規定値を超えるリクエストは、必ず実行時間を指定してください。実行時間は最大値まで設定が可能です。その他のオプションは表 7 をご参照ください。

表 6 SX-ACE の利用形態と-q および-b オプション

利用形態	利用ノード数[5]	実行時間制限 (経過時間)	メモリサイズ制限	-q オプション	-b オプション
通常	1～256	規定値： 2 週間 最大値： 1 ヶ月	60GB×ノード数	sx	利用ノード数
	257～1,024	規定値： 1 ヶ月 最大値： 1 ヶ月			
無料	1	1 時間	60GB	sx	f
デバッグ用	1～16	2 時間	60GB×ノード数	debug	利用ノード数
	17～32	24 時間			

表 7 qsub コマンドの主なオプション

オプション	機能
-q(必須)	計算機名 sx または debug を指定します。
-b(必須)	利用ノード数または f を指定します。

[5] 513 ノード以上の利用は、2015 年後半を予定しています。

-A	課金先のプロジェクトコードを指定します。指定が無ければデフォルトのプロジェクトコードに課金されます。
-N	リクエスト名を指定します。指定がなければ、リクエストファイル名がリクエスト名になります。
-o	標準出力のファイル名を指定します。指定がなければ、リクエスト投入時のディレクトリに「リクエスト名.o リクエスト ID の番号」のファイル名で出力されます。
-e	標準エラー出力のファイル名を指定します。指定がなければ、リクエスト投入時のディレクトリに「リクエスト名.e リクエスト ID の番号」のファイル名で出力されます。
-jo	標準エラー出力を標準出力と同じファイルへ出力します。
-l elapstim_req=hh:mm:ss	最大経過時間を指定します。設定時間は、時:分:秒を hh:mm:ss の形式で指定します。
-m b	リクエストの処理が開始したときにメールが送られます。
-m e	リクエストの処理が終了したときにメールが送られます。
-M メールアドレス	メールの送信先を指定します。指定がなければ、「利用者番号@front.cc.tohoku.ac.jp」宛に送られます。

● 逐次プログラム、自動並列/OpenMP 並列の場合

リスト 6 は逐次プログラムを実行する場合のバッチリクエストファイルの一例です。ホームディレクトリ直下 work ディレクトリの a.out を実行する手続きを記述しています。

リスト 6 バッチリクエストファイル例

```
# test job.1              コメント行
#PBS -q sx -b 1          #実行マシンとノード数を指定
cd work                  #実行ディレクトリへ移動
./a.out                  #実行形式ファイルを指定
```

- 1行目:# 以降はコメントです。動作には影響しません。
- 2行目:SX-ACE の 1 ノードを使う指定です。
- 3行目:cd work で作業ディレクトリ(実行形式ファイルのあるディレクトリ)へ移動します。省略するとホームディレクトリを指定したことになります。
- 4行目:a.out はコンパイルして作成した実行形式ファイル名です。あらかじめコンパイルし作成しておきます。自動並列や OpenMP による並列処理も、同じ形式で指定します。

参考 1 : 作業ディレクトリの指定

NQSⅡ用の環境変数のひとつに PBS_O_WORKDIR 変数があります。この変数には、qsub コマンドを実行した時点のカレントディレクトリが設定されます。

つまり、work ディレクトリでこのリクエストを投入する(qsub を実行する)と、$PBS_O_WORKDIR にはカレントディレクトリの work が設定され cd work と同じことになります。PBS_O_WORKDIR 変数を設定することで、ディレクトリの具体名を記述する必要がなくなります。

リスト 7 環境変数 PBS_O_WORKDIR

```
# test job.2              コメント行
#PBS -q sx -b 1          #実行マシンとノード数を指定
cd $PBS_O_WORKDIR        #実行ディレクトリを環境変数で指定
./a.out                  #実行形式ファイルを指定
```

参考 2：qsub コマンドオプションの埋め込み

表 7 に qsub コマンドの主なオプションを示します。-q と-b オプションは必須指定です。

qsub コマンドのオプションはバッチリクエストファイルに埋め込みます（リスト 8）。設定方法は、最初のコマンドより前の行に、#PBS という文字列を先頭に指定します。#PBS の後に空白を一文字以上入れ、指定したいオプションを続けます。一行に複数のオプション指定も可能です。

リスト 8 の例は、2 行目で実行計算機に SX-ACE を指定(-q sx)と、利用ノード数 1 を指定（-b 1）し、3 行目で標準エラー出力を標準出力ファイルにひとまとめにし(-jo)、4 行目でリクエスト名を test04 に指定(-N test04)し、5 行目で最大経過時間を 1 時間（-l elapstim_req=01:00:00）に、それぞれ指定しています。

埋め込みオプションとコマンド列に同じオプションを指定した場合は、コマンド列の方が優先されます。

リスト 8 オプションの埋め込み

```
# test job.4
#PBS -q sx -b 1
#PBS -jo                          #標準エラー出力を標準出力と同じファイルへ出力
#PBS -N test04                    #リクエスト名を test04 にする
#PBS -l elapstim_req=01:00:00     #最大経過時間を 1 時間に設定
cd $PBS_O_WORKDIR
./a.out
```

● MPI 並列の場合

MPI プログラムの場合は mpirun コマンドにオプションと実行形式ファイル名を指定します。mpirun コマンドのオプションに -np と -nn の指定が必要です。-np オプションには総 MPI プロセス数、-nn には利用するノード数を指定します。-b で指定する値と、-nn で指定する値は必ず同数に指定します。SX-ACE は 1 ノードに 4 コアを搭載していますので、-np には -b の 4 倍の数まで指定できます。

なお、バッチリクエストファイル中で指定した環境変数は、MPIEXPORT のあとにダブルクオーテーション中に記述し、全てのノードに設定を反映させる必要があります。

MPI 並列プログラムを 32 ノードを利用して実行する例をリスト 9 に示します。

```
mpirun オプション ./実行形式ファイル名
```

表 8 mpirun コマンドのオプション（必須）

オプション	引数
-np	総 MPI プロセス数
-nn	使用ノード数（-b と同数を指定）

リスト 9 MPI 並列実行（32 ノード利用）

```
# test job.3
#PBS -q sx -b 32                  #SX-ACE の 32 ノードを利用する
cd $PBS_O_WORKDIR
mpirun -np 128 -nn 32 ./a.out     #128MPI プロセスで実行する
```

● ハイブリッド並列の場合

MPI と自動並列/OpenMP 並列されたハイブリッド並列のプログラムを実行する場合、ノード内の並列数を環境変数で指定する必要があります。SX-ACE では自動並列数と OpenMP 並列数を指定する環境変数が異なりますのでご注意ください（表 9）。

MPI と自動並列のプログラムをハイブリッド並列で実行する例を、リスト 10 に示します。

表 9 ノード内並列数を指定する環境変数

並列方法		環境変数
自動並列(-Pauto)	Fortran プログラムの場合	F_RSVTASK
	C/C++プログラムの場合	C_RSVTASK
OpenMP 並列(-Popenmp)		OMP_NUM_THREADS

リスト 10 ハイブリッド並列実行(32 ノード利用、自動並列)

```
# test job.4
#PBS –q sx -b 32                #SX-ACE の 32 ノードを利用する
cd $PBS_O_WORKDIR
setenv F_RSVTASK 4              #ノード内は自動並列で 4 並列実行
setenv MPIEXPORT “F_RSVTASK”    #全てのノードに環境変数を設定
mpirun –np 32 –nn 32 ./a.out    #32MPI プロセス x4 並列＝128 並列で Fortran プログラムを実行する
```

● 標準入出力ファイルの指定

1 つのノードでの実行では、標準入出力ファイルをリダイレクション記号(<, >)で指定することができますが、MPI プログラムの実行ではこれらのリダイレクション記号は利用できません。

この場合、標準入力ファイルの指定は環境変数 F_FF05 で行います(リスト 11)。標準出力および標準エラー出力は、MPI 並列数分複数出力されます。このとき mpisep.sh を利用すると、同一ファイルにプロセス毎の出力が入り混じって出力されないように、MPI プロセスごとにファイルを分けて出力することができます(リスト 12)。シェルスクリプト /usr/lib/mpi/mpisep.sh を実行形式ファイル a.out の前に記述し、環境変数 MPISEPSELECT に 1～4 の値を指定することで、表 10 の動作を選択します。

リスト 11 標準入出力ファイルの指定例

```
# test job.5
#PBS –q sx -b 32
setenv F_FF05 infile            #標準入力は装置番号 05
setenv MPIEXPORT “F_FF05”       #全てのノードに環境変数を設定
cd $PBS_O_WORKDIR
mpirun –np 32 –nn 32 ./a.out
```

リスト 12 標準出力、標準エラーファイルをプロセス毎に分割

```
# test job.6
#PBS -q sx -b 32
setenv MPISEPSELECT 4           #出力形式を 4 に設定
setenv MPIEXPORT “MPISEPSELECT” #全てのノードに環境変数を設定
cd $PBS_O_WORKDIR
mpirun -np 32 -nn 32 /usr/lib/mpi/mpisep.sh ./a.out
```

表 10 MPISEPSELECT の引数と動作

MPISEPSELECT の引数	動作
1	各 MPI プロセスの標準出力のみを stdout.uuu:rrr へ格納
2	各 MPI プロセスの標準エラーのみを stderr.uuu:rrr へ格納(規定値)
3	各 MPI プロセスの標準出力と標準エラーを各々stdout.uuu:rrr および stderr.uuu:rrr へ格納
4	各 MPI プロセスの標準出力と標準エラーを同一ファイル stdout.uuu:rrr へ格納

・実行時、stdout.*や stderr.*の同名ファイルが存在する場合は、上書きでなく追加出力します。

● MPI 用実行性能情報

MPIPROGINF 環境変数の設定により、MPI プロセス毎のプログラム性能情報(PROGINFO)の表示形式を変更することができます。DETAIL は Min/Max/Ave にまとめたもの、ALL_DETAIL は全てのプロセスについて、それぞれ性能情報を表示します。性能情報はプログラム実行終了後、標準エラー出力ファイルに出力されます。

表 11 MPIPROGINF の引数と動作

MPIPROGINF の引数	動作
DETAIL	性能情報を集約形式で出力します。
ALL_DETAIL	性能情報を拡張形式で出力します。 全ランクの情報を出力します。

■　バッチリクエストの投入

プログラムの実行は、作成したバッチリクエストファイルを NQSⅡに投入することで行います。

```
front1 $ qsub オプション　バッチリクエストファイル名
```

リクエストが正常に投入されると、システムからのメッセージが返ります(リスト 13)。1234.job1 がリクエスト ID で、リクエストの状況確認やキャンセル等、リクエストの操作の際に指定が必要になります。

リスト 13 qsub コマンド実行時のメッセージ例

```
front1$ qsub run.csh
Request 1234.job1 submitted to queue : sx32.
```

■ バッチリクエストの状態確認

● reqstat コマンド

reqstat コマンドは、投入されたリクエストの状態を表示します(
リスト 14)。状態は STATE 項目に表示されます(表 13)。リクエストはバッチサーバ(job1 または job2)毎に出力されます。

リソースに空きがなければ実行待ち状態になり、順番が回ってくると自動的に実行状態に入ります。システム内に自分のリクエストが存在しない場合は、「No request.」と表示されます(リスト 15)。

リスト 14 reqstat コマンド例

```
front1$ reqstat
statistics sampled at 2015/02/20 19:18:01 in job1.
REQUEST ID      USER     GROUP    QUEUE    NODE ELAPS     STATE       TIMES             REQUEST NAME
--------------- -------- -------- -------- ---- --------- ----------- ----------------- ---------------
1733.job1       利用者番号 users    sx64      64   1:00:00  running     15/02/20 12:00:00 test01
1735.job1       利用者番号 users    sx64      64   1:00:00  queued      15/02/20 18:55:00 test03

statistics sampled at 2015/02/20 19:18:01 in job2.
REQUEST ID      USER     GROUP    QUEUE    NODE ELAPS     STATE       TIMES             REQUEST NAME
--------------- -------- -------- -------- ---- --------- ----------- ----------------- ---------------
1734.job2       利用者番号 users    sx64      64   0:00:00  wait        15/02/20 13:55:00 test02
```

表 12 reqstat コマンドの主な表示項目

項目名	内容
RequestID	リクエスト ID
USER	利用者番号
GROUP	利用者の所属グループ
QUEUE	キュー名
NODE	利用ノード数
ELAPS	経過時間制限
STATE	リクエストのステータス
TIMES	ステータスが変化した日時
REQUEST NAME	-N で指定したリクエスト名もしくはバッチリクエストファイル名

表 13 STATE 項目の主な表示とリクエスト状態

表示	リクエスト状態
wait	実行ノードの決定待ち
queued	実行ノードが決まり、実行順待ち
running	実行中

リスト 15 投入したリクエストがない場合

```
front1 $ reqstat
No request.
```

● **sstat コマンド**

sstat コマンドは投入されたリクエストの実行開始予定時刻を表示します(リスト 16)。ただし、255 ノード以下の利用の場合のみ表示されます。利用ノード数が少ない場合バックフィルスケジュール機能により、予定実行開始時刻が早まることがあります。-l elapstim_req オプションで指定した時間が短いほど、待ちリクエストの隙間にリクエストの実行が割り当てられる可能性が大きくなりますので、必要十分な実行時間を指定することをお勧めいたします。

リスト 16 sstat コマンド例

```
front1 $ sstat

RequestID       ReqName  UserName Queue                  Pri STT PlannedStartTime
--------------- -------- -------- -------- ----------------- --- -------------------
2512.job1       test03   利用者番号 sx32       0.5002/  0.5002 ASG 2015-02-20 08:28:20
2513.job1       test04   利用者番号 sx32       0.5002/  0.5002 ASG 2015-02-20 09:28:20
```

表 14 sstat コマンドの表示項目

項目名	内容
RequestID	リクエスト ID
ReqName	-N で指定したリクエスト名 指定がなければ、リクエストファイル名がリクエスト名になります。
UserName	利用者番号
QUEUE	キュー名
Pri	優先度
STT	リクエストのステータス
PlannedStartTime	予定実行開始時刻

■ バッチリクエストのキャンセル

投入したリクエストの削除、または実行中のリクエストを停止する場合は、qdel コマンドにリクエスト ID を指定します(リスト 17)。リクエスト投入時、または reqstat コマンドで表示されるリクエスト ID をバッチサーバ名まで指定してください。

リスト 17 qdel コマンド例

```
front1 $ qdel 1234.job1
Request 1234.job1 was deleted.
```

6 章　スーパーコンピュータ用ライブラリ

■　科学技術計算ライブラリ ASL

ASL(Advanced Scientific Library)は、科学技術計算の広範な分野の数値シミュレーションプログラムの作成を強力に支援する FORTRAN 版数学ライブラリです。ASL を用いることによって、難解な数値計算アルゴリズムの詳細に煩わされること無く高度な科学技術計算プログラムを作成することができます。また、ASL C 言語インタフェース(Advanced Scientific Library C INTerface)は、C/C++プログラムから FORTRAN 版 ASL を容易に利用できるように作成されたインタフェースライブラリです。

ASL は次のような数値計算分野に対応しています。

＜基本機能＞
　格納モードの変換、基本行列演算、最小二乗法、固有値・固有ベクトル、連立 1 次方程式(直接法)、連立 1 次方程式(反復法)、対称連立一次方程式(反復法)、非対称連立一次方程式(反復法)、フーリエ変換とその応用／時系列分析、微分方程式とその応用、数値微分、数値積分、3 次元境界要素法用の数値積分法、近似・補間、スプライン関数、特殊関数、乱数、ソート・順位付け、方程式の根、極値問題・最適化
＜統計機能＞
　確率分布、基礎統計量、推定と検定、分散分析・実験計画、ノンパラメトリック検定、多変量解析、フーリエ変換とその応用／時系列分析、近似・回帰分析、乱数、ソート・順位付け
＜共有メモリ並列処理機能＞
　基本行列演算、連立 1 次方程式(直接法)、固有値・固有ベクトル、フーリエ変換とその応用／時系列分析、乱数、ソート・順位付け
＜分散メモリ並列処理機能＞
　二次元複素フーリエ変換、三次元複素フーリエ変換

■ Fortran プログラムから利用する場合

Fortran プログラムから ASL ライブラリを利用する場合、リスト 18 のようにコンパイルとリンクを行います。コンパイルオプションとリンクオプションの指定方法については
表 15 をご参照ください。

リスト 18 ASL のリンク方法(Fortran)

```
シングル版・共有メモリ並列版
front1 $ sxf90 〈コンパイルオプション〉 source.f90 〈リンクオプション〉

分散メモリ並列版
front1 $ sxmpif90 〈コンパイルオプション〉 source.f90 〈リンクオプション〉
```

表 15 ASL とコンパイルオプション及びリンクオプションの指定方法（Fortran）

機能			コンパイル オプション	リンク オプション
基本機能	32 ビット整数型	シングル版		-lasl
		共有メモリ並列版	-Pmulti	-lasl64
	64 ビット整数型	シングル版		-lasl
		共有メモリ並列版	-Pmulti	-lasl64
統計機能	32 ビット整数型			-laslstat
	64 ビット整数型			-laslstat64
分散メモリ並列処理機能	dw 版		-Pmulti	-laslmpi -lasl
	ew 版		-Pmulti -ew	-laslmpiew -laslew
Fortran90 インタフェース				-laslf90 -lasl

・自動並列化を利用する場合は-Pauto、OpenMP を利用する場合は-Popenmp をコンパイルオプションにします。

リスト 19 自動並列の Fortran プログラムで 64 ビット整数型共有メモリ並列版 ASL をリンクする例

```
front1 $ sxf90 -Pauto source.f90 -lasl64
```

■ C/C++プログラムから利用する場合

C/C++プログラムから ASL ライブラリを利用する場合、以下のようにコンパイルとリンクを行います。
表 15 のオプションに加え、コンパイルオプションに「-f90lib」および「-size_t32 または-size_t64」を追加し、リンクオプションに 32 ビット整数型または dw 版を使用する場合は 「-laslcint」を、64 ビット整数型または ew 版を使用する場合は 「-laslcint64」を追加して指定してください。

リスト 20 ASL のリンク方法（C/C++）

```
シングル版・共有メモリ並列版
front1 $ sxcc 〈コンパイルオプション〉 source.c 〈リンクオプション〉
front1 $ sxc++ 〈コンパイルオプション〉 source.cpp 〈リンクオプション〉

分散メモリ並列版
front1 $ sxmpic 〈コンパイルオプション〉 source.c 〈リンクオプション〉
front1 $ sxmpic++ 〈コンパイルオプション〉 source.cpp 〈リンクオプション〉
```

リスト 21 C プログラムで 32 ビット整数型共有メモリ並列版 ASL をリンクする例

```
front1 $ sxcc -f90lib -size_t32 -Pmulti source.c -laslcint -lasl
```

■ 数学ライブラリ集 MathKeisan

MathKeisan for SX は NEC のハイパフォーマンス・コンピュータ用に高度に最適化された数学ライブラリ集です。MathKeisan に含まれるライブラリは以下のとおりです。MathKeisan のいくつかのサブルーチンについては、同機能のものが ASL にも含まれています。同機能であれば SX-ACE 用に最適化された ASL の利用をお薦めします。

■ Fortran プログラムから利用する場合

Fortran プログラムから MathKeisan ライブラリを利用する場合、リスト 22 のようにコンパイルとリンクを行います。-dw オプション(既定値)を使用してリンクする場合には、表 16 のように指定します。-ew オプションを使用してリンクする場合には、表 17 のように指定します。表に示された順番にリンクするか、またはリンク時に-hlib_cyclic オプションを指定する必要があります。

リスト 22 MathKeisan ライブラリのリンク方法（Fortran）

```
シングル版・共有メモリ並列版
front1 $ sxf90 <コンパイルオプション> source.f90 <リンクオプション>      (dw 版)
front1 $ sxf90 -ew <コンパイルオプション> source.f90 <リンクオプション> (ew 版)

分散メモリ並列版
front1 $ sxmpif90 <コンパイルオプション> source.f90 <リンクオプション>      (dw 版)
front1 $ sxmpif90 -ew <コンパイルオプション> source.f90 <リンクオプション> (ew 版)
```

表 16 MathKeisan ライブラリとリンクオプションの指定方法（-dw（規定値）を使用する場合）

MathKeisan ライブラリ	リンクオプション
BLAS	-lblas
LAPACK	-llapack -lblas
ScaLAPACK	-lscalapack -lblacsF90init -lblacs -lblacsF90init -llapack -lblas -lmpi
BLACS	-lblacsF90init -lblacs -lblacsF90init -lmpi
PARBLAS	-lparblas -Popenmp
CBLAS	-lcblas -lblas
SBLAS	-lsblas
FFT	-lfft
PARFFT	-lparfft -Popenmp
ARPACK	-larpack -llapack -lblas
PARPACK	-lparpack -larpack -llapack -lblas -lmpi

表 17 MathKeisan ライブラリとリンクオプションの指定方法（-ew を使用する場合）

MathKeisan ライブラリ	リンクオプション
BLAS	-lblas_64
LAPACK	-llapack_64 -lblas_64
ScaLAPACK	-lscalapack_64 -lblacsF90init_64 -lblacs_64 -lblacsF90init_64 -llapack_64 -lblas_64 -lmpiw
BLACS	-lblacsF90init_64 -lblacs_64 -lblacsF90init_64 -lmpiw
PARBLAS	-lparblas_64 -Popenmp
CBLAS	サポートなし
SBLAS	-lsblas_64
FFT	-lfft_64
PARFFT	-lparfft_64 -Popenmp
ARPACK	-larpack_64 -llapack_64 -lblas_64
PARPACK	-lparpack_64 -larpack_64 -llapack_64 -lblas_64 -lmpiw

■ C/C++プログラムから利用する場合

C/C++コンパイラを使って CBLAS、 BLACS 以外のライブラリをリンクする場合は、コマンドラインで MathKeisan ライブラリを指定したの後に、-f90lib オプションを付けてください。この時、-dw の MathKeisan ライブラリをリンクする場合は-f90lib dw オプションをつけ、-ew の MathKeisan ライブラリをリンクする場合は-f90lib ew オプションを付けてください。

7章 プログラムについての補足

■ 使用メモリサイズ

実行に必要なメモリサイズを事前に確認することができます。ただし allocate 等で動的に確保するメモリサイズは含まれません。

逐次プログラムの場合

リスト 23 メモリサイズの確認（逐次プログラム）

```
front1 $ sxsize 実行形式ファイル名
```

並列化プログラムの場合（自動並列または OpenMP による並列化）

リスト 24 メモリサイズの確認（並列プログラム）

```
front1 $ sxsize -fl 並列数 実行形式ファイル名
```

リスト 25 sxsize コマンド表示例

```
front1$ sxsize a.out.s （逐次プログラム）
956688(.txt) + 1274432(.data) + 542080(.bss) + 84434(.comment) + 4856(.ELNI) +
4908(.whoami) = 2867398
（逐次プログラム a.ous.s を実行すると、2,867,398 byte = 約 2.8MB 使用します）

front1$ sxsize -fl 4 a.out.p （並列プログラム）
1267584(.text) + 1287920(.data) + 457568(.bss) + 111678(.comment) + 5088(.whoami) +
1080528(logical task region) * 4 = 7451950
（並列化プログラム a.out.p を 4 並列実行すると、7,451,950 byte = 約 7.2MB 使用します）
```

■ スタックサイズの指定

並列化オプション（-Pauto/-Popenmp）オプションを付けてコンパイルしたプログラムを実行した際に、Segmentation fault で実行がエラー終了することがあります。コンパイラが推定するスタックサイズの不足が原因となることがありますので、リンク時にオプションで適切なスタック領域を確保してください。リスト 26 の例では 2GB のスタック領域を確保します。

リスト 26 スタックサイズを 2GB に指定する例

```
front1 $ sxf90 -Pauto source.f90 -Wl,-Z 2G
```

コンパイラはリンク時に以下のようにスタックサイズを見積もります。

- **並列処理用ロードモジュールの場合（-Pauto/-Popenmp）**

  スタックサイズの合計値に 35KB を加えた値

- **並列処理用でないロードモジュールの場合**

  約 96MB

■ バイナリファイルの扱い [Fortran]

データファイルを入力するなど他のサーバで作成したバイナリファイルを扱う場合、注意が必要です。SX-ACE システムは Big-Endian 仕様ですので、Little-Endian 仕様[6]のバイナリファイルを扱うには、Endian を合わせる必要があります。変換は環境変数 F_UFMTENDIAN を用い、Big-Endian に変換したい Little-Endian ファイルの装置番号を指定します。

リスト 27 環境変数 F_UFMTENDIAN

```
setenv F_UFMTENDIAN u[,u]…
```

・u は Little-Endian ファイルの装置番号です。複数ある場合は、”,”(カンマ)で区切って羅列します。

リスト 28 バッチリクエストファイル例

```
# test job.7
#PBS -q sx -b 32
setenv F_UFMTENDIAN 30,40       #装置番号 30,40 を Big-Endian に変換する
setenv MPIEXPORT “F_UFMTENDIAN”
cd $PBS_O_WORKDIR
./a.out
mpirun –np 32 –nn 32 ./a.out
```

■ バウンズチェック [Fortran]

配列の添字について、宣言している範囲外を参照していないか検査します。

プログラムの実行中に、適切な値を処理しているはずなのにエラー番号 250(オーバーフロー)や 252(ゼロ割り)、253(演算例外)等が発生する、または実行エラーは発生しないが実行する毎に動作や演算結果が異なるという場合、配列の添字検査をしてみてください。配列で参照されている添字が許される範囲にあるかどうかを調べ、範囲外であれば配列の大きさ、添字の値を表示します。

チェック方法は、-eC オプションを付けコンパイル(
リスト 29)してから実行します。範囲外添字が見つかると、標準エラー出力ファイルにエラーメッセージを出力します(リスト 30)。

リスト 29 バウンズチェックのオプション

```
front1 $ sxf90 –eC ソースファイル名
```

・-eC オプションを指定すると、最適化やベクトル化および並列化はされませんので、通常の実行に比べ実行時間が非常に長くなります。添字検査を行う時のみ、このオプションを用いてください。

リスト 30 エラーメッセージ例

```
*240 Subscript error array=b size=10 subscript=11 eln=756 PROG=main ELN=756(400021981)

配列 b はサイズ 10 の宣言であるが、添字 11 を使っているためエラーが発生している。
エラー発生箇所は、main ルーチン 756 行目。
```

[6] 本センターの並列コンピュータでは、Fortran プログラムでは環境変数で Big-Endian に設定されています。

8 章　マニュアル

■　コンパイラマニュアル

● テキスト版マニュアル

テキスト版のマニュアルはのリスト 31 のコマンドで参照できます。

リスト 31 コンパイラマニュアルの参照方法（テキスト版）

```
front1 $ sxman sxf90     (Fortran コンパイラ)
front1 $ sxman sxcc      (C コンパイラ)
front1 $ sxman sxc++     (C++コンパイラ)
```

● PDF 版マニュアル

PDF 版のマニュアルを閲覧するには並列コンピュータの SSH 接続時に X forwarding を行う必要があります。リスト 32 のコマンドを実行すると Firefox ブラウザが起動し、PDF マニュアルが閲覧できます。

リスト 32 コンパイラマニュアルの参照方法（PDF 版）

```
front1 $ mansxlangj      (日本語版)
front1 $ mansxlange      (英語版)
```

■　ライブラリマニュアル

■ ASL

リスト 33 のディレクトリに PDF 版マニュアルがあります。

リスト 33 ASL の PDF 版マニュアル

```
/usr/ap/ASL-man-super/PDF/ASL          (ASL マニュアル日本語版)
/usr/ap/ASL-man-super/PDF/ASL_e        (ASL マニュアル英語版)
/usr/ap/ASL-man-super/PDF/CINT         (ASL C 言語インタフェースマニュアル日本語版)
/usr/ap/ASL-man-super/PDF/CINT_e       (ASL C 言語インタフェースマニュアル英語版)
```

日本語マニュアルには表 18 に示したものがあります。

表 18 ASL マニュアル（日本語版）

ファイル名	タイトル	内容
1main.pdf	基本機能編 第 1 分冊	格納モードの変換,基本行列演算,最小二乗法,固有値・固有ベクトル
2main.pdf	基本機能編 第 2 分冊	連立1次方程式(直接法)
3main.pdf	基本機能編 第 3 分冊	連立1次方程式(反復法),対称連立一次方程式(反復法),非対称連立一次方程式(反復法)
4main.pdf	基本機能編 第 4 分冊	フーリエ変換とその応用／時系列分析

5main.pdf	基本機能編 第 5 分冊	微分方程式とその応用,数値微分,数値積分,3次元境界要素法用の数値積分法近似・補間,スプライン関数
6main.pdf	基本機能編 第 6 分冊	特殊関数,乱数,ソート・順位付け,方程式の根,極値問題・最適化
7main.pdf	統計機能 ASLSTAT 利用の手引	確率分布,基礎統計量,推定と検定,分散分析・実験計画,ノンパラメトリック検定,多変量解析,近似・回帰分析
8main.pdf	共有メモリ並列処理機能編	基本行列演算,連立一次方程式(直接法),固有値・固有ベクトル,連立一次方程式(反復法)フーリエ変換とその応用／時系列分析,乱数,ソート・順位付
9main.pdf	分散メモリ並列処理機能編	複素フーリエ変換

■ MathKeisan ライブラリ

並列コンピュータに SSH X Forwarding 接続し、 Firefox ブラウザを起動してリスト 34 のファイルをご参照ください。

リスト 34 MathKeisan ライブラリのマニュアル

```
/usr/ap/Mathkeisan-man/SX-ACE/J/index.html   （日本語版）
/usr/ap/Mathkeisan-man/SX-ACE/E/index.html   （英語版）
```

9 章　利用負担金

■　利用負担金について

利用負担金は、演算負担経費、ファイル負担経費、出力負担経費、可視化負担経費の 4 つがあります(表 19、表 20)。スーパーコンピュータと並列コンピュータを利用すると、演算負担経費が発生します。

共有利用は利用するノード数と経過時間によって負担額が決定し、計算資源を利用者間で共有利用する利用形態です。

また、占有利用は計算資源を待ち時間なく占有して利用することができ、申請する利用期間によって負担額が決定する利用形態です。

請求書は四半期(3 ヶ月)ごとに、利用者を取りまとめている支払責任者に発行します。最新の情報は以下の Web サイトをご覧ください。

```
http://www.ss.cc.tohoku.ac.jp/utilize/academic.html（学術利用）
http://www.ss.cc.tohoku.ac.jp/utilize/business.html（民間機関利用）
```

表 19 基本利用負担金(大学・学術利用)

区 分	項 目	利用形態	負 担 額		
演 算 負担経費	スーパー コンピュータ	共有	利用ノード数 1	(実行数、実行時間の制限有)	無料(備考2)
			利用ノード数 1～32 まで	経過時間 1 秒につき	0.06 円
			利用ノード数 33～256 まで	経過時間 1 秒につき	(利用ノード数-32)×0.002 円+0.06 円
			利用ノード数 257 以上	経過時間 1 秒につき	(利用ノード数-256)×0.0016 円+0.508 円
		占有	利用ノード数 32	利用期間 3 ヶ月につき 利用期間 6 ヶ月につき	400,000 円 720,000 円
			利用ノード数 64	利用期間 3 ヶ月につき 利用期間 6 ヶ月につき	720,000 円 1,300,000 円
			利用ノード数 128	利用期間 3 ヶ月につき 利用期間 6 ヶ月につき	1,300,000 円 2,340,000 円
	並列 コンピュータ	共有	利用ノード数 1～6 まで	経過時間 1 秒につき	0.04 円
			利用ノード数 7～12 まで	経過時間 1 秒につき	0.07 円
			利用ノード数 13～18 まで	経過時間 1 秒につき	0.1 円
			利用ノード数 19～24 まで	経過時間 1 秒につき	0.13 円
		占有	利用ノード数 1	利用期間 3 ヶ月につき (可視化システムの 20 時間無料利用を含む) 利用期間 6 ヶ月につき (可視化システムの 40 時間無料利用を含む)	160,000 円 320,000 円
ファイル 負担経費	1TB まで無料、追加容量 1TB につき年額				3,000 円
出 力 負担経費	大判プリンタによるカラープリント			フォト光沢用紙 1 枚につき クロス 1 枚につき	600 円 1,200 円
可視化 機器室利用 負担経費	1 時間の利用につき				2,500 円

表 20 基本利用負担金(民間機関利用)

区 分	項 目	利用形態	負 担 額		
演 算 負担経費	スーパー コンピュータ	共有	利用ノード数 1	(実行数、実行時間の制限有)	無料(備考2)
			利用ノード数 1～32 まで	経過時間 1 秒につき	0.18 円
			利用ノード数 33～256 まで	経過時間 1 秒につき	(利用ノード数-32)×0.006 円+0.18 円
			利用ノード数 257 以上	経過時間 1 秒につき	(利用ノード数-256)×0.0048 円+1.524 円
		占有	利用ノード数 32	利用期間 3 ヶ月につき 利用期間 6 ヶ月につき	1,200,000 円 2,160,000 円
			利用ノード数 64	利用期間 3 ヶ月につき 利用期間 6 ヶ月につき	2,160,000 円 3,900,000 円
			利用ノード数 128	利用期間 3 ヶ月につき 利用期間 6 ヶ月につき	3,900,000 円 7,020,000 円
	並列 コンピュータ	共有	利用ノード数 1～6 まで	経過時間 1 秒につき	0.12 円
			利用ノード数 7～12 まで	経過時間 1 秒につき	0.21 円
			利用ノード数 13～18 まで	経過時間 1 秒につき	0.3 円
			利用ノード数 19～24 まで	経過時間 1 秒につき	0.39 円

	並列 コンピュータ	占有	利用ノード数 1	利用期間 3 ヶ月につき （可視化システムの 20 時間無料利用を含む） 利用期間 6 ヶ月につき （可視化システムの 40 時間無料利用を含む）	480,000 円 960,000 円
ファイル 負担経費	1TB まで無料、追加容量 1TB につき年額				9,000 円
出　力 負担経費	大判プリンタによるカラープリント			フォト光沢用紙 1 枚につき クロス 1 枚につき	1,800 円 3,600 円
可視化 機器室利用 負担経費	1 時間の利用につき				7,500 円

備考

1　負担額算定の基礎となる測定数量に端数が出た場合は、切り上げる。
2　負担額が無料となるのは専用のジョブクラスで実行されたものとし、制限時間を超えた場合には強制終了する。
3　占有利用期間は年度を超えないものとし、期間中に障害、メンテナンス作業が発生した場合においても、原則利用期間の延長はしない。また、占有利用期間中のファイル負担経費は 10TB まで無料とする。
4　ファイル負担経費については申請日から当該年度末までの料金とする。

■　利用負担金の確認方法

● プロジェクトコードの確認（project コマンド）

利用負担金はプロジェクトコード毎に合算されます。利用可能なプロジェクトコードの確認、デフォルトのプロジェクトコードの設定は project コマンドをご利用ください。プロジェクトコードの名称変更、追加などについては共同利用支援係までお問い合わせください。

リスト 35 プロジェクトコードの確認

```
front1 $ project
使用可能なプロジェクトおよびキュー名は次のとおりです

利用者番号:(利用者番号)
-----------------------  使用可能なプロジェクト一覧  -----------------------

  プロジェクト名称     :  運営費交付金
  プロジェクトコード   :  un0000
  使用可能なキュー名   :  sx32 sx64 ･･･

デフォルトのジョブ実行プロジェクトは un0000 です

    1.デフォルトプロジェクト変更     9.終了

何番の処理を選びますか ？
```

● プロジェクト課金情報表示（pkakin コマンド、ukakin コマンド）

コマンドを実行した利用者が利用可能なプロジェクトについて、負担額や請求情報を表示します。負担額は前日の午前 9 時までに終了したリクエストの利用額までが反映されています。

リスト 36 プロジェクトごとの負担額、請求情報の表示

```
front1 $ pkakin
   2 月 20 日  現在の利用負担金は次のとおりです

プロジェクトコード      累計負担額      調整額累計      固定費合計      請求済額   今期請求予定額(うち請求持越額)
un0000                    15,404               0               0             0       15,404               0

支払責任者と経理担当者の所属(学校、学部)が異なる場合はセンターに連絡してください
```

```
支払費目の指定は各部局の経理担当者(学内の場合)、もしくはセンター会計係(学外の場合)へお伝えください

    1.負担金の明細表示  9.終了

何番の処理を選びますか ? 1
出力する年度を入力して下さい ( 1.今年度  2.前年度 ) : 1
開始月を入力してください : 2
終了月を入力してください : 2
出力先を選択してください  ( 1.画面  2.ファイル ) : 1

================================================================================================================
                                    プロジェクト負担金明細情報

    2 月 20 日  現在の利用額および負担額は次のとおりです

  支払責任者          :  東北 太郎
  支払責任者番号      :  aaaaaa
  プロジェクト名称    :  運営費交付金
  プロジェクトコード  :  un0000

                        合計          演算 SX          演算 LX        ファイル            出力          可視化
  利用額              15,143          14,635              508             0               0               0
  負担額              15,143          14,635              508             0               0               0
----------------------------------------------------------------------------------------------------------------
月別利用額
         2 月             15,143
----------------------------------------------------------------------------------------------------------------
利用者別利用額
        abc000               21
        abc001
           :                  6
        abc020               27
        abc021            4,895
```

リスト 37 利用者ごとの利用額情報の表示

```
front1 $ ukakin

        利用者番号=(利用者番号)

 ----------------------  利 用 額  ----------------------
  支払責任者          :  東北 太郎
  支払責任者番号      :  aaaaaa
  プロジェクト名称    :  運営費交付金
  プロジェクトコード  :  un0000

    月       演算 SX      演算 LX      ファイル         出力         可視化         合計
    4              0            0             0            0            0            0
    5              0            0             0            0            0            0
    6              0            0             0            0            0            0
    7              0            0             0            0            0            0
    8              0            0             0            0            0            0
    9              0            0             0            0            0            0
   10              0            0             0            0            0            0
   11              0            0             0            0            0            0
   12            170           10             0            0            0          180
    1            546          162             0            0            0          708
    2              0            0             0            0            0            0
    3              0            0             0            0            0            0
 合計            716          172             0            0            0          888
```

● ジャーナル(利用明細)の抜粋(ulist コマンド、plist コマンド)

コマンドを実行した利用者のジャーナル情報、プロジェクトごとのジャーナル情報を抜粋して CSV 形式(カンマ区切り)のファイルに出力します。表 21 の項目が出力されます。

リスト 38 利用者のジャーナル情報を抜粋

```
front1 $ ulist

出力する年度を入力して下さい ( 1.今年度  2.前年度 ) : 1
開始月を入力してください : 2
終了月を入力してください : 2
出力するファイル名を入力してください ( 省略時:ulist.csv ) :

ホスト ID は次のとおりです
  20:SX  04:LX  02:front  05:ストレージ  06:プリンタ  08:可視化
```

リスト 39 プロジェクトのジャーナル情報を抜粋

```
front1 $ plist

出力する年度を入力して下さい ( 1.今年度  2.前年度 ) : 1
開始月を入力してください : 2
終了月を入力してください : 2
出力するファイル名を入力してください ( 省略時:plist.csv ) :
プロジェクトを選択してください

    1.(un0000)     運営費交付金

何番のプロジェクトを選びますか ? 1

ホスト ID は次のとおりです
  20:SX  04:LX  02:front  05:ストレージ  06:プリンタ  08:可視化
```

表 21 ulist コマンドの出力項目

課金計上年月,ジャーナル連番,利用者番号,支払責任者番号,プロジェクトコード,ホスト ID,クラス ID,キュー名,投入日時,開始日時,終了日時,経過時間,使用ノード数,ユーザ CPU 時間,最大メモリサイズ,ベクトル時間,ベクトル演算率,ノード時間(使用量),利用額

● ジャーナルの集計(usum コマンド、psum コマンド)

コマンドを実行した利用者のジャーナル情報、プロジェクトごとのジャーナル情報を集計して表示します。

リスト 40 利用者のジャーナル情報を集計

```
front1 $ usum

出力する年度を入力して下さい ( 1.今年度  2.前年度 ) : 1
開始月を入力してください : 2
終了月を入力してください : 2

----------------------  利 用 情 報  ----------------------

利用者番号  プロジェクトコード  ホスト ID       経過時間合計  ユーザ CPU 時間合計    最大メモリサイズ  ノード時間(使用量)合計
abc001      un0000            20        15:10:00          45:55:58         61,394           18:42:08
abc001      un0000            02      1867:16:29            :  : 0          5,596             :43:03

ホスト ID は次のとおりです
  20:SX  04:LX  02:front  05:ストレージ  06:プリンタ  08:可視化
```

リスト 41 プロジェクトのジャーナル情報を集計

```
front1 $ psum

出力する年度を入力して下さい ( 1.今年度  2.前年度 ) : 1
開始月を入力してください : 2
```

```
終了月を入力してください : 2
プロジェクトを選択してください

    1.(un0000)     運営費交付金

何番のプロジェクトを選びますか ? 1

-----------------------  利 用 情 報  -----------------------

プロジェクトコード   ホスト ID        経過時間合計   ユーザ CPU 時間合計      最大メモリサイズ   ノード時間(使用量)合計
un0000              0         7545:43:59          :  : 0            34,816              1:25:04
un0000              20          13:13:14     2642:34:25             61,436          2171:14:38

ホスト ID は次のとおりです
  20:SX  04:LX  02:front  05:ストレージ  06:プリンタ  08:可視化
```

■ 利用見込み額の設定

プロジェクトコードの請求先が学外の場合は、請求処理の都合上 2 月中旬以降、3 月末までの利用額を見込み金額として設定します。設定した金額は、1 月から 2 月中旬までの利用負担金に合算し、2 月末に請求いたします。見込み金額の設定は mikomi コマンドで行います。

リスト 42 mikomi コマンドの例

```
front1 $ mikomi

プロジェクトを選択してください
プロジェクトコード : プロジェクト名称  支払責任者番号 : 支払責任者氏名
un0000        : 運営費交付金        aaaaaa      : 東北 太郎

プロジェクトコードを入力してください un0000

 2 月 1 日 現在の見込み額は次のとおりです

  支払責任者      : 東北 太郎
  支払責任者番号 : aaaaaa
  プロジェクト名称 : 運営費交付金
  プロジェクトコード: un0000
  見込み額指定者 :

  見込み額                :         0 円
  今期請求予定額          :   300,000 円
  合計請求額(見込み額込) :   300,000 円

 1.見込み額の指定 2.削除 9.終了
 何番の処理を選びますか ? 1

  見込み額を入力してください (円) ? 200000
  登録してよろしいですか (yes/no) ? yes

 2 月 1 日 現在の見込み額は次のとおりです

  支払責任者      : 東北 太郎
  支払責任者番号 : aaaaaa
  プロジェクト名称 : 運営費交付金
  プロジェクトコード: un0000
  見込み額指定者 : 東北 太郎 (利用者番号 変更日: 2 月 1 日)

  見込み額                :     200,000 円
  今期請求予定額          :     500,000 円
  合計請求額(見込み額込) :     500,000 円

 見込み額の指定 2.削除 9.終了
 何番の処理を選びますか ?
```

10 章　運用に関する情報、お問い合わせ

■　運用に関するお知らせ

大規模科学計算システムのホームページで利用方法やシステムの運用状況について、最新情報をお知らせします。計画停電による運用の停止スケジュールなどについても、こちらでお知らせしております。

URL http://www.ss.cc.tohoku.ac.jp/

■　利用申請

利用申請に関する詳細は、以下のウェブページをご覧ください。申請書の提出、お問合せは共同利用支援係までお願いいたします。

URL http://www.ss.cc.tohoku.ac.jp/utilize/index.html
メールアドレス　uketuke@cc.tohoku.ac.jp　**電話番号**　022-795-3406

■　利用に関するお問い合わせ

■ システムの利用方法についてのお問合せ

利用相談員が対応いたします。以下のメールアドレスまでお問合せください。

メールアドレス　sodan05@cc.tohoku.ac.jp

■ 利用負担金についてのお問合せ

共同利用支援係までお問い合わせください。

メールアドレス　uketuke@cc.tohoku.ac.jp　**電話番号**　022-795-3406

■ 請求書についてのお問合せ

請求書の発送等については会計係までお問い合わせください。

メールアドレス　kaikei@cc.tohoku.ac.jp　**電話番号**　022-795-3405

■ その他のお問合せ

総務係までお問い合わせください。

メールアドレス　som@cc.tohoku.ac.jp　**電話番号**　022-795-3407

おわりに

本稿では、SX-ACE システムのプログラミング利用ガイドとして基本的な手順を紹介しました。研究室のサーバでは実現できなかったプログラムやアイデアを、ぜひ最新鋭のスーパーコンピュータシステムでお試しください。研究の強力なツールとしてご活用いただければ幸いです。ご不明な点、ご質問等ございましたら、お気軽にセンターまでお問い合わせください。

［大規模科学計算システム］

並列コンピュータ LX 406Re-2 の利用法

情報部情報基盤課　共同利用支援係　共同研究支援係
サイバーサイエンスセンター　スーパーコンピューティング研究部

1 章　はじめに

本センターは並列コンピュータ LX 406Re-2 の運用を 2014 年 4 月から開始しています。本稿では、LX 406Re-2 システムでのプログラミング利用ガイドとして、プログラムの作成からコンパイル、実行等の使い方と利用負担金についてご紹介します。

2 章　システム構成

システム構成は、既設のシステムを含め図1のようになっています。

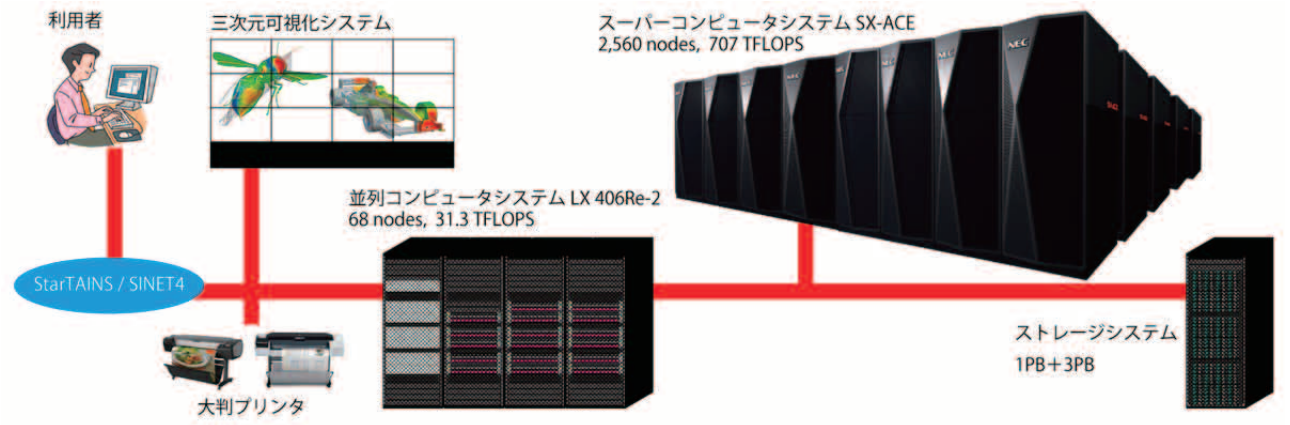


図 1. 大規模科学計算システム構成図

並列コンピュータ LX 406Re-2

並列コンピュータ LX 406Re-2 は 1 ノードに、インテル Xeon プロセッサ E5-2695v2（12 コア）を 2 基と 128GB の主記憶装置を搭載し、合計 68 ノードで構成されます。自動並列化・OpenMP・MPI を利用したノード内の並列処理は 24 並列まで可能で、ノードあたりの最大演算性能は 460.8GFLOPS（倍精度）となります。複数のノードを使用した並列処理は、MPI の利用により最大 576 並列まで実行可能です。ベクトル演算に不向きなプログラムの高速な実行が可能です。また、スーパーコンピュータ SX-ACE のフロントエンドサーバとしての役割も担っています。

3 章　プログラミング　～逐次処理、共有メモリ並列処理 ～

本章では、単一のコアで実行する逐次処理と、自動並列化および OpenMP による共有メモリ並列処理について利用手順を紹介します。

MPI による並列化プログラミング手順については、4 章で紹介しますが、コンパイルコマンドに違いがある以外は、同じ手順ですのでこの章と合わせてご覧ください。

■ ログイン

作業を行うため並列コンピュータにログインします。リモート接続は、ssh コマンドまたは SSH 対応リモート接続ソフト[1]をご利用ください。

並列コンピュータの OS は Linux です。公開鍵暗号方式による認証のみ利用できます[2]。アカウント希望の場合は、共同利用支援係に利用申請し利用者番号と初期パスワードを発行してもらいます。

並列コンピュータへの初回ログイン時には公開鍵と秘密鍵のペアを作成する必要があります。鍵ペアの作成方法については本誌 105 ページの「SSH アクセス認証鍵生成サーバの利用方法」をご参照ください。

なお、他人名義の利用者番号でのシステム利用は禁止します。パスワード、秘密鍵、パスフレーズの使い回しは、不正アクセスのリスク（不正ログイン、クライアントのなりすまし、暗号化された通信の暴露、他サーバへの攻撃等）が非常に高く、大変危険です。利用者登録を行うことによる年間維持費等は発生しませんので、利用される方はそれぞれで利用申請をお願いいたします。

並列コンピュータホスト名

```
front.cc.tohoku.ac.jp
```

リスト 1. ssh コマンドによる接続例

```
localhost$ ssh -i ~/.ssh/id_rsa 利用者番号@front.cc.tohoku.ac.jp
Enter passphrase for key '/home/localname/.ssh/id_rsa':パスフレーズを入力
（初回接続時のメッセージ） : yes を入力
Front1$ （コマンド待ち状態）
```

暗号鍵は複数の端末や他人と共有してはいけません。また、メールに暗号鍵を添付したり USB メモリにコピーしたりすると不正アクセスのリスクとなります。並列コンピュータにログインする端末を追加する場合は、その端末で新規に鍵ペアを作成し、authorized_keys に追加登録してください。

■ ログイン端末の追加方法

ログインする端末を追加する場合は、追加する端末で鍵ペアの作成を行います。必ずパスフレーズの設定を行って鍵を作成して下さい。作成した公開鍵を並列コンピュータにログイン出来る端末に転送します。公開鍵ですので転送方法はメール本文に記載しても構いません。

作成した公開鍵の内容を利用者のホームディレクトリのファイル（~/.ssh/authorized_keys）に追記します。接続元が Linux、OS X の場合の鍵の作成方法をリスト 2 に示します。

[1] Windows であれば、TeraTerm 等のフリーソフトが利用できます。
[2] パスワード認証方式は 2015 年 4 月 13 日で廃止しました。

リスト 2. 公開鍵と秘密鍵の作成方法

```
【利用する端末で鍵ペアを作成】
localhost $ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/localname/.ssh/id_rsa):（ファイル名を指定）

Enter passphrase (empty for no passphrase):（必ずパスフレーズを設定）
Enter same passphrase again:（同じパスフレーズを入力）

指定した場所（/home/localname/.ssh）に鍵ペア（暗号鍵：id_rsa　公開鍵：id_rsa.pub）が生成される

【作成した公開鍵を既に並列コンピュータにログイン出来る端末に転送】

【作成した公開鍵を並列コンピュータに転送】
localhost $ scp /home/localname/.ssh/id_rsa.pub 利用者番号@front.cc.tohoku.ac.jp:~
（パスフレーズを入力）

【公開鍵を追記登録】
front1 $ cat ~/id_rsa.pub >> ~/.ssh/authorized_keys
front1 $ exit
```

~/.ssh/authorized_keys を削除すると、全ての暗号鍵からのログインが出来なくなります。センターではセキュリティインシデントに対する緊急対応として、全ユーザの~/.ssh/authorized_keys を削除する場合があります。

■ パスワードの変更

パスワードの変更は passwd コマンドで行います。パスワードの変更方法をリスト 3 に示します。

コマンドを実行すると、フロントエンドサーバ（並列コンピュータ）、可視化サーバ、プリンタサーバ、およびファイル転送サーバのログインパスワードが変更されます。入力したパスワードは表示されません。

リスト 3. パスワードの変更方法

```
front1 $ passwd
ユーザー 利用者番号 のパスワードを変更。
Enter login(LDAP) password:  （現在のパスワードを入力）
新しいパスワード:  （新しいパスワードを入力）
新しいパスワードを再入力してください:（新しいパスワードを入力）
LDAP password information changed for 利用者番号
passwd: 全ての認証トークンが正しく更新できました。
```

■ ログインシェルの確認と変更

ログインシェルの確認と変更は fchsh コマンドで行います。ログインシェルの確認方法と変更方法をリスト 4 に示します。

ログインシェルの変更が SX-ACE と LX 406Re-2 に反映されるまで 15 分程度かかります。

リスト 4. ログインシェルの確認方法と変更方法

```
front1 $ fchsh （ログインシェルの確認）
Enter Password: （パスワードを入力）
loginShell: /bin/tcsh （現在のログインシェルが表示される）

front1 $ fchsh /bin/bash （ログインシェルを/bin/bash に変更）
Enter Password: （パスワードを入力）
Changed loginShell to /bin/bash （ログインシェルが変更された）
```

■ ホームディレクトリ

ホームディレクトリは、プログラムファイル等を置く自分専用のディスク領域です。ディレクトリ名は、/uhome/利用者番号です。利用者番号作成時の容量制限は 1TB です。ファイル容量の追加申請によりディスク領域を増やすことも可能です。ホームディレクトリはスーパーコンピュータシステムと並列コンピュータシステムで共有しています。

```
/uhome/利用者番号
```

■ プログラミング言語、ライブラリ

プログラミング言語および科学技術計算用ライブラリとして表 1 に示すものが利用できます。

表1．プログラミング言語およびライブラリ

Fortran	Intel Fortran Composer XE
C/C++	Intel C++ Composer XE
MPI	Intel MPI ライブラリ
数値演算ライブラリ	NEC NumericFactory, Intel MKL 他

■ ファイルエディット

ソースファイルは、並列コンピュータにログインし、emacs エディタまたは vi エディタで作成します。研究室等のパソコンにあるソースファイルを利用するには、front.cc.tohoku.ac.jp の利用者ディレクトリにファイル転送してください。送り元のホストが Windows の場合、転送モードの設定を”ASCII”にすることで適切な改行コードで転送できます。転送手順につきましては、以下の Web ページをご参照ください。

http://www.ss.cc.tohoku.ac.jp/application/setting.html

■ コンパイル

Fortran および C/C++コンパイラの基本的な使用方法です。詳しいオプション等については man コマンド、および**マニュアル**をご覧ください。

Fortran プログラムのコンパイル

ifort コマンドでコンパイルします。利用したい機能があれば適当なオプションと、ソースファイル名を指定します。

ソースファイルの拡張子は、自由形式（フリーフォーマット）なら.f90 か.F90、固定形式（7カラム目から記述）なら.f か.F を付けます。

● コンパイル【逐次処理】

```
front1 $ ifort  オプション   ソースファイル名
```

● コンパイル【自動並列化】

```
front1 $ ifort  -Pauto オプション  ソースファイル名
```

・OpenMP プログラムなら、-Pauto の箇所を-Popenmp にします。

主なオプション

-parallel -par-report	自動並列化機能を利用する。 自動並列化されたループの行番号を表示する。
-openmp -openmp-report	OpenMP を利用する。 OpenMP 指示行により並列化されたループ、領域、セクションの行番号を表示する。
-O0	最適化を無効にする。
-O1	最適化を行うが、コードサイズが増える最適化は行わない。
-O2 または -O	一般的な最適化を行う。（規定値）
-O3	高度の最適化（プリフェッチ、スカラリプレスメント、ループ変換等）を行う。
-ip	インライン展開を行う。
-c	コンパイルのみ行う。（リンクはしない）
-o	実行可能形式のオブジェクトファイルの名前を指定する。省略時は a.out になる。
-w90	非標準 Fortran 機能に関する警告メッセージを抑止する。
-r8	精度の自動拡張を行う。（倍精度化）
-help	オプションの種類と説明を表示する。

ソースファイル名

Fortran のソースプログラムファイル名を指定します。複数のファイルを指定するときは、空白で区切ります。
ソースファイル名には、サフィックス.f90 か.F90（自由形式）、または.f か.F（固定形式）が必要です。

C/C++プログラムのコンパイル

C プログラムを **icc** コマンドで C++プログラムを、**icpc** コマンドでコンパイルします。利用したい機能があれば適当なオプションと、ソースファイル名を指定します。

並列化コンパイルは、ここで並列数を意識する必要はありません。実行する時点で希望する並列数を環境変数で指定します。ノード内並列数は自動並列の場合は環境変数 F_RSVTASK で指定し、OpenMP 並列の場合は環境変数 OMP_NUM_THREADS で指定します。詳細は 5 章で説明します。

● コンパイル【逐次処理】

```
front1 $ icc  オプション  ソースファイル名
front1 $ icpc  オプション  ソースファイル名
```

● コンパイル【自動並列化】

```
front1 $ icc -Pauto  オプション  ソースファイル名
front1 $ icpc -Pauto オプション  ソースファイル名
```

・OpenMP プログラムなら、-Pauto の箇所を-Popenmp にします。

主なオプション

-parallel -par-report	自動並列化機能を利用する。 自動並列化されたループの行番号を表示する。
-openmp -openmp-report	OpenMP を利用する。 OpenMP 指示行により並列化されたループ、領域、セクションの行番号を表示する。
-O0	最適化を無効にする。
-O1	最適化を行うが、コードサイズが増える最適化は行わない。
-O2 または -O	一般的な最適化を行う。（規定値）
-O3	高度の最適化（プリフェッチ、スカラリプレスメント、ループ変換等）を行う。
-ip	インライン展開を行う。
-c	コンパイルのみ行う。（リンクはしない）
-o	実行可能形式のオブジェクトファイルの名前を指定する。省略時は a.out になる。
-help	オプションの種類と説明を表示する。

ソースファイル名

C/C++のソースプログラムファイル名を指定します。複数のファイルを指定するときは、空白で区切ります。
ソースファイル名にはサフィックス .c 、C++のソースファイル名にはサフィックス .cc または .C が必要です。

4 章 プログラミング ～ MPI 並列処理 ～

本章では、MPI による並列化プログラミング手順について紹介します。基本的な手順は前章と同じですので、コンパイルコマンドの異なる点について紹介します。

■ コンパイルを行う

MPI プログラムは、MPI 用コマンド mpiifort、mpiicc、mpiicpc コマンドでコンパイルします。

MPI 並列 Fortran プログラムのコンパイル

MPI 並列の Fortran プログラムは mpiifort コマンドでコンパイルします。利用したい機能があればオプションと、ソースファイル名を指定します。

```
front1 $ mpiifort  オプション  ソースファイル名
```

・オプションは、ifort コマンドと共通です。man ifort コマンドでご覧ください。

MPI 並列 C/C++プログラムのコンパイル

MPI 並列の C プログラムは **mpiicc** コマンドで、MPI 並列の C++プログラムは **mpiicpc** コマンドでコンパイルします。利用したい機能があれば適当なオプションと、ソースファイル名を指定します。

```
front1 $ mpiicc  オプション  ソースファイル名
front1 $ mpiicpc オプション  ソースファイル名
```

・オプションは、**icc** コマンド、**icpc** コマンドと共通です。詳細は man icc コマンド、man icpc コマンドでご確認ください。

5 章 バッチリクエスト

■ プログラムの実行

コンパイルして作成された実行形式ファイルを実行するには、以下の 2 つの処理方法があります。通常はバッチ処理を利用します。

【バッチ処理】

バッチ処理は、実行の手続きをジョブという単位でジョブ管理システムに登録し、一括に処理します。ジョブ管理システムは NQSⅡ(Network Queuing SystemⅡ)を用意しており、ジョブの操作は NQSⅡのコマンドで行います。通常のプログラム(長時間実行するプログラム、並列実行するプログラム等)はバッチ処理で実行します。

【会話型処理】

会話型処理は、コマンドラインでプログラムを実行する形式です。CPU 時間や使用できるメモリサイズに制限がありますので、短時間の演算やデバッグ作業にお使いください。

■ バッチ処理

プログラムの実行は、NQSⅡのコマンドを用いて操作します。図2は作業の流れを示しています。まず NQSⅡにプログラムの実行を依頼するため、実行の手続きを書いたバッチリクエストを作成します。このバッチリクエストを NQSⅡに投入することで、プログラムの実行が可能になります。

バッチリクエストの投入後は、バッチリクエストの状態や、混み具合の確認、また投入済みのバッチリクエストをキャンセルすることも可能です。プログラムの実行が終了するとバッチリクエストは NQSⅡの情報から消え、標準出力ファイルと標準エラー出力ファイルが出力されます。

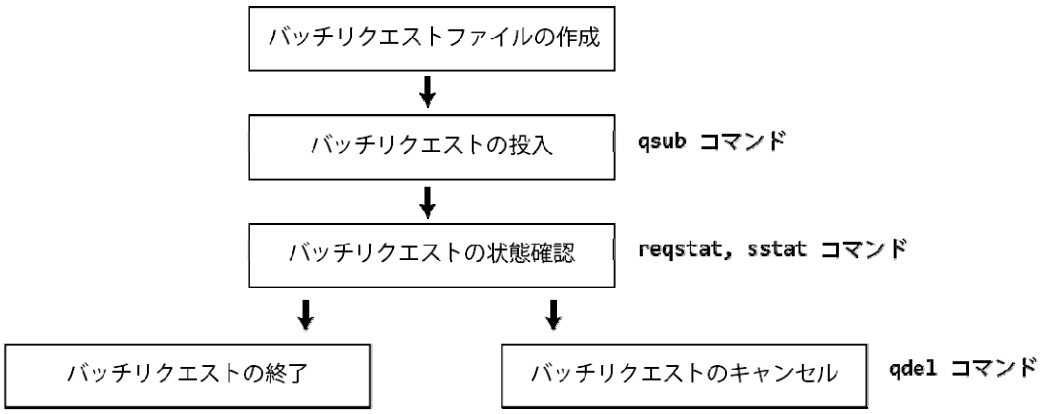


図 2. NQSII によるバッチリクエストの流れ

【バッチリクエストファイルの作成】

プログラムの実行手続きを、通常のシェルスクリプトと同じ形式で記述します。csh スクリプトと sh スクリプト、どちらでも記述できます(以降、解説は csh スクリプトで記述します)。適当なファイル名を付け作成します。以降ではバッチリクエストファイル名を run.csh とします。

基本的に必要となるのは、実行するマシンとノード数の指定、ホームディレクトリから作業ディレクトリへ移動、プログラムの実行、です。他に環境変数の指定、ファイルの操作コマンド等があれば適切な箇所に手続きを記述します。

並列コンピュータ LX 406Re-2 で実行する場合の利用形態と必須オプションを表 2 に示します。通常の利用の場合、-q の後に lx を指定し、-b の後に利用ノード数または a を指定します。

実行時間の設定は、-l elapstim_req=hh:mm:ss で設定します。通常利用で 1～24 ノードを利用する場合、実行時間制限の規定値でリクエストは自動的に終了します。実行時間が規定値を超えるリクエストは、必ず実行時間を指定してください。実行時間は最大値まで設定が可能です。その他のオプションは表 3 をご参照ください。

表 2. 並列コンピュータ LX 406Re-2 の利用形態と-q および-b オプション

利用形態	利用ノード数	実行時間制限 （経過時間）	メモリサイズ制限	-q オプション	-b オプション
通常	1～24	規定値：1 カ月 最大値：1 カ月	128GB×ノード数	lx	利用ノード数
ｱﾌﾟﾘｹｰｼｮﾝ	1	なし	128GB	lx	a

表 3. qsub コマンドの主なオプション

-q（必須）	計算機名 lx を指定します。
-b（必須）	リクエストを実行するノード数、または a を指定します。
-A	課金先のプロジェクトコードを指定します。指定が無ければデフォルトのプロジェクトコードに課金されます。
-N	リクエスト名を指定します。指定がなければ、リクエストファイル名がリクエスト名になります。
-o	標準出力のファイル名を指定します。指定がなければ、リクエスト投入時のディレクトリに「リクエスト名.o リクエスト ID」のファイル名で出力されます。
-e	標準エラー出力のファイル名を指定します。指定がなければ、リクエスト投入時のディレクトリに「リクエスト名.e リクエスト ID」のファイル名で出力されます。
-jo	標準エラー出力を標準出力と同じファイルへ出力します。
-l elaptim_req=hh:mm:ss	最大経過時間を指定します。設定時間は、時:分:秒を hh:mm:ss の形式で指定します。
-m b	リクエストの処理が開始したときにメールが送られます。
-m c	リクエストの処理が終了したときにメールが送られます。
-M メールアドレス	メールの送信先を指定します。指定がなければ、 「利用者番号@front.cc.tohoku.ac.jp」宛に送られます。

・その他オプションの詳細は、man qsub コマンドでご覧ください。

● 逐次プログラム、自動並列/OpenMP 並列の場合

リスト 5 は逐次プログラムを実行する場合のバッチリクエストファイルの一例です。ホームディレクトリ直下 work ディレクトリの a.out を実行する手続きを記述しています。

リスト 5. バッチリクエストファイル例

```
# test job-a                    コメント行
cd work                         #作業ディレクトリへ移動
./a.out                         #実行形式ファイルを指定
```

・1 行目 :#以降はコメントです。動作には影響しません。
・2 行目 :cd work で作業ディレクトリ（実行形式ファイルのあるディレクトリ）へ移動します。省略するとホームディレクトリを指定したことになります。
・3 行目 : a.out はコンパイルして作成した実行形式ファイル名です。あらかじめコンパイルし作成しておきます。
自動並列や OpenMP による並列処理も同じ形式で指定します。

● 作業ディレクトリの指定

NQSⅡ用の環境変数のひとつに PBS_O_WORKDIR 変数があります。この変数には、qsub コマンドを実行した時点のカレントディレクトリが設定されます（リスト 6）。つまり、work ディレクトリでこのバッチリクエストを投入する(qsub を実行する)と、$PBS_O_WORKDIR にはカレントディレクトリの work が設定され cd work と同じことになります。PBS_O_WORKDIR 変数を設定することで、ディレクトリの具体名を記述する必要がなくなります。

リスト 6. バッチリクエストファイル（環境変数 PBS_O_WORKDIR の指定）

```
# test job-a1                  コメント行
cd $PBS_O_WORKDIR              #作業ディレクトリを環境変数で指定
./a.out                        #実行形式ファイルを指定
```

● 実行時のデータファイル指定

Fortran プログラムで入出力ファイルを割り当てる環境変数 FORT*n* です。n が 1～9 の場合には 0 をつけず 1 桁で指定します（リスト 7）。
正しい指定方法:　　setenv FORT2 datafile

リスト 7. バッチリクエストファイル（入出力ファイルの指定例）

```
# test job-b                   コメント行
setenv FORT1 datafile          #装置番号 1 に、ファイル datafile を割り当てる
cd $PBS_O_WORKDIR              #作業ディレクトリを環境変数で指定
./a.out < infile > outfile     #標準入出力ファイルはリダイレクションでも可能
```

● qsub コマンドオプションの埋め込み

qsub コマンドに毎回オプションを入力することもできますが、手間を省くためバッチリクエストファイルに指定しておくこともできます。

指定方法は、最初のコマンドより前の行に、#PBS という文字列を先頭に指定します。#PBS の後に空白を一文字以上入れ、指定したいオプションを続けます。一行に複数のオプション指定も可能です。

リスト 8 の例は、2 行目で実行計算機に lx を指定(-q lx)、利用ノード数 6 を指定(-b 6)、3 行目で標準エラー出力を標準出力ファイルにひとまとめにし(-jo)、4 行目でリクエスト名を test04 とする(-N test04)を、それぞれ指定しています。

埋め込みオプションとコマンド列に同じオプションを指定した場合は、コマンド列の方が優先されます。

リスト 8. run.csh バッチリクエストファイル（オプションの埋め込み）

```
# test job-a2
#PBS -q lx -b 6                #実行マシンとノード数を指定
#PBS -jo                       #標準エラー出力を標準出力と同じファイルへ出力
#PBS -jo -N test04             #リクエスト名を test04 にする
cd $PBS_O_WORKDIR              #作業ディレクトリを環境変数で指定
./a.out
```

・【バッチリクエストの投入】

プログラムの実行は、作成したバッチリクエストファイルを NQSⅡに投入することで行います。

```
front1 $ qsub オプション　バッチリクエストファイル名
```

リクエストが正常に投入されると、システムからのメッセージが返ります（リスト 9）。1234.job1 がリクエスト ID で、リクエストの状況確認やキャンセル等、リクエストの操作の際に指定が必要になります。

リスト 9. qsub コマンドの実行例

```
front1$ qsub run.csh
Request 1234.job1 submitted to queue: lx6.
```

フラット MPI プログラムの実行（MPI のみの並列）

MPI プログラムは、mpirun コマンドを使用して実行します。バッチリクエストファイルに mpirun コマンドとオプションを記述します（リスト 10）。

-ppn オプションに **1 ノードあたりのプロセス数**を指定します。-np オプションに**合計プロセス数**を指定します。

-ppn オプションは-np オプションよりも前で指定する必要があります。

表 4. mpirun コマンドのオプション（必須）

オプション	引数
-ppn	1 ノードあたりのプロセス数
-np	合計プロセス数を指定

リスト 10. フラット MPI プログラム用バッチリクエストファイル例

（MPI 並列数 144 で実行する場合）

```
# test job-a
#PBS -q lx -b 6                    #実行マシンとノード数を指定
cd $PBS_O_WORKDIR                  #作業ディレクトリを環境変数で指定
mpirun -ppn 24 -np 144 ./a.out     #MPI プログラムの実行
```

ハイブリッド並列プログラムの実行（MPI と自動並列/OpenMP を組み合わせた並列）

ハイブリッド並列プログラム実行時の並列数は「プログラム並列数＝MPI 並列数×SMP 並列（自動並列/OpenMP）」になります。MPI プログラムの並列数は mpirun コマンドの -ppn オプションと -np オプションで制御し、自動並列/OpenMP 並列数は OMP_NUM_THREADS 環境変数で制御します（リスト 11）。

表 5. ノード内並列数を指定する環境変数

並列方法		環境変数
自動並列（-Pauto）	Fortran プログラムの場合	F_RSVTASK
	C/C++プログラムの場合	C_RSVTASK
OpenMP 並列（-Popenmp）		OMP_NUM_THREADS

リスト 11. ハイブリッド並列プログラム用バッチリクエストファイル例

（MPI 並列数 6、自動並列数/OpenMP 並列数 24 の 144 並列で実行する場合）

```
# test job-a
#PBS -q lx -b 6                    #埋め込みオプション
setenv OMP_NUM_THREADS 24          #自動並列/Open MP での並列数
cd $PBS_O_WORKDIR                  #作業ディレクトリを環境変数で指定
mpirun -ppn 1 -np 6 ./a.out        #MPI プログラムの実行
```

■ バッチリクエストの状態確認

● reqstat コマンド

reqstat コマンドは、投入されたリクエストの状態を表示します（リスト 12）。状態は STATE 項目に表示されます（表 7）。リクエストはジョブサーバ（job1 または job2）毎に出力されます。

リソースに空きがなければ実行待ち状態になり、順番が回ってくると自動的に実行状態に入ります。システム内に自分のリクエストが存在しない場合は、「No request.」と表示されます（リスト 13）。

リスト 12. reqstat コマンド例

```
front1$ reqstat
statistics sampled at 2015/02/20 19:18:01 in job1.
REQUEST ID      USER     GROUP    QUEUE    T NODE ELAPS     STATE       TIMES              REQUEST NAME
--------------- -------- -------- -------- - ---- --------- ----------- ------------------ ---------------
2512.job1       利用者番号 users    lx6      S   6    1:00:00 running     15/02/20 12:00:00 test02
2513.job1       利用者番号 users    lx6      S   1    1:00:00 queued      15/02/20 18:55:00 test04
```

表 6. reqstat コマンドの主な表示項目

項目名	内容
RequestID	リクエスト ID
USER	利用者番号
GROUP	利用者の所属グループ
QUEUE	キュー名
T	リクエストタイプ（通常は S）
NODE	利用ノード数
ELAPS	経過時間制限
STATE	リクエストのステータス
TIMES	ステータスが変化した日時
REQUEST NAME	-N で指定したリクエスト名もしくはバッチリクエストファイル名

表 7. STATE 項目の主な表示とリクエスト状態

表示	リクエスト状態
wait	実行ノードの決定待ち
queued	実行ノードが決まり、実行順待ち
running	実行中

リスト 13. 投入したリクエストがない場合

```
front1 $ reqstat
No request.
```

● sstat コマンド

sstat コマンドは投入されたリクエストの実行開始予定時刻を表示します（リスト 14）。ただし、マップスケジュール機能が有効な場合のみです。エスカレーション機能により、予定実行開始時刻が他の待ちリクエストより早まることがあります。-l elapstim_req オプションで指定した時間が短いほど、待ちリクエストの隙間にリクエストの実行が割り当てられる可能性が大きくなりますので、必要十分な実行時間を指定することをお勧めいたします。

リスト 14. sstat コマンド例

```
front1 $ sstat

RequestID       ReqName  UserName Queue                  Pri STT PlannedStartTime
--------------- -------- -------- -------- ----------------- --- -------------------
2512.job1       test03   利用者番号 lx6       0.5002/  0.5002 RUN Already Running...
2513.job1       test04   利用者番号 lx6       0.5002/  0.5002 ASG 2015-02-20 19:22:20
```

表 8. sstat コマンドの表示項目

項目名	内容
RequestID	リクエスト ID
ReqName	-N で指定したリクエスト名
UserName	利用者番号
QUEUE	キュー名
Pri	優先度
STT	リクエストのステータス
PlannedStartTime	予定実行開始時刻

■ バッチリクエストのキャンセル

投入したリクエストの削除、または実行中のリクエストを停止する場合は、qdel コマンドにリクエスト ID を指定します(リスト 15)。リクエスト投入時、または reqstat コマンドで表示されるリクエスト ID をジョブサーバ名まで指定してください。

リスト 15. qdel コマンド例

```
front1 $ qdel 1234.job1
Request 1234.job1 was deleted.
```

■ 会話型処理

会話型処理は、短時間の演算やデバッグ作業に使用します。一般的な UNIX を利用する手順と同様で、コマンドラインから実行形式ファイル名を入力し実行する形式です(リスト 16)。表 9 は会話型処理の制限値です。時間制限は CPU 時間の合計ですので、並列実行した場合はそれぞれの CPU 時間の合計値となり、1 時間経過する前にジョブが終了します。

リスト 16. 会話型処理の例(a.out を実行する)

```
yourhost$ ssh front.cc.tohoku.ac.jp -l 利用者番号      front にログインする
    :
front1$ a.out
（プログラム実行中）
front1$
（実行終了）
```

表 9. 会話型処理の制限値

利用ノード数（最大並列数）	時間制限 [時間]	最大メモリ [GB]
1(6)	1 時間（CPU 時間合計）	8

6 章　ライブラリ

以下のライブラリを使用することができます。

Fortran,C/C++ 用

数値計算ライブラリ集	NEC NumericFactory
数値演算ライブラリ	Intel MKL
画像処理ライブラリ	Intel IPP
マルチスレッドライブラリ	Intel TBB

■ 数値計算ライブラリ集　NEC NumericFactory

【機能概要】

NumericFactory は、NEC が独自に開発している数値計算ライブラリと、数値シミュレーションプログラムで頻繁に利用される OSS（Open Source Software）により、多彩な数値計算アルゴリズムを提供します。NumericFactory の使用により、プログラム開発の時間を短縮でき、高品質なプログラムを開発することが出来ます（表 10）。

NumericFactory は、全 12 種類のライブラリで構成されています。ライブラリにより、使用できる言語が異なります（表 11）。また、並列化された機能を含むものと含まないものがあります。OpenMP 並列の機能がないライブラリでも、下位で使用する Intel MKL が並列化されている場合、マルチスレッドで動作することがあります。OpenMP 並列、または、MPI 並列機能がないライブラリでも、OpenMP/MPI プログラムから利用することは可能です。

表 10. NumericFactory の機能概要

ライブラリ名	機能概要
ASL	行列積、疎行列用連立 1 次方程式（直接法／反復法）、固有値方程式、FFT、乱数、特殊関数、近似・補間、スプライン、微分方程式、数値微積分、方程式の根、数理計画法、ソート・順位付け
ASLSTAT	乱数、基礎統計量、推定・検定、分散分析・実験計画、多変量解析、フーリエ解析、回帰分析
ASLQUAD	四倍精度演算機能（基本演算、連立 1 次方程式、固有値方程式、特殊関数）
SFMT	メルセンヌツイスター擬似乱数生成（整数）
dSFMT	メルセンヌツイスター擬似乱数生成（実数）
SuperLU	疎行列用連立 1 次方程式（直接法）
MUMPS	疎行列用連立 1 次方程式（直接法）
Lis	疎行列用連立 1 次方程式、疎行列用固有値方程式（反復法）
ARPACK	大規模固有値問題
PARPACK	大規模固有値問題（MPI 版）
XBLAS	精度拡張／精度混合行列積
METIS	行列、グラフ並べ替え、グラフ分割

表 11. NumericFactory のライブラリと利用可能言語

ライブラリ名	Fortran から利用	C から利用	OpenMP 並列機能	MPI 並列機能
ASL	○	○	○	×
ASLSTAT	○	○	×	×
ASLQUAD	○	×（C++可）	○	×
SFMT	×	○	×	×
dSFMT	×	○	×	×
SuperLU	×	○	×	×
MUMPS	○	○	×	○
Lis	○	○	○	○
ARPACK	○	×	×	×
PARPACK	○	×	×	○
XBLAS	○	○	×	×
METIS	○	○	×	×

【ライブラリのリンク方法】

● 逐次版/OpenMP 版プログラムの場合

各ライブラリのリンクには、ifort、icc コマンドを使用します。利用するプログラム言語に応じて、リスト 17、18 のようにリンクしてください。リンクオプションは使用するライブラリと言語に応じて指定します（表 12、表 13）。

さらに今回から、MKL のリンクオプションが，以下のように簡潔に指定することもできるようになりました。

-lmkl_intel_ilp64 -lmkl_sequential -lmkl_core -pthread

↓

-mkl=sequential

詳細は、man コマンドでご覧ください。

NumericFactory でサポートしているライブラリを使用する場合、ライブラリによってはユーザプログラム側でモジュールファイルやヘッダファイルをインクルードする必要があります（表 14）。

Fortran から ASL または ASLSTAT の 64 ビット整数に対応したライブラリを利用する場合、コンパイル時に必ずオプション"-i8"を付けてコンパイルしてください。このオプションは、integer 型を 64 ビット整数と翻訳する Intel コンパイラのオプションです。

リスト 17. Fortran の場合（逐次版/OpenMP 版）

```
[front1 ~]$ ifort source.f90 <リンクオプション>
```

リスト 18. C の場合（逐次版/OpenMP 版）

```
[front1 ~]$ icc source.c <リンクオプション>
```

表 12. NumericFactory のリンクオプション（逐次版/OpenMP 版 Fortran プログラムから利用する場合）

ライブラリ名	リンクオプション	
ASL	32bit整数/逐次版	-lasl -mkl=sequential
	64bit整数/逐次版	-lasl64 -mkl=sequential
	32bit整数/OpenMP版	-lasl -mkl=parallel
	64bit整数/OpenMP版	-lasl64 -mkl=parallel
ASLSTAT	32bit整数版	-laslstat -mkl=sequential
	64bit整数版	-laslstat64 -mkl=sequential
ASLQUAD	32bit整数/逐次版	-laslquad
	32bit整数/OpenMP版	-laslquad
Lis	逐次版	-llis_seq
	OpenMP版	-llis_omp
ARPACK		-larpack -mkl=sequential
XBLAS		-lxblas
METIS		-lmetis

表 13. NumericFactory のリンクオプション（逐次版/OpenMP 版 C プログラムから利用する場合）

ライブラリ名	リンクオプション	
ASL	32bit整数/逐次版	-laslcint -lasl -mkl=sequential -lifcore -limf
	64bit整数/逐次版	-laslcint64 -lasl64 -mkl=sequential -lifcore -limf lilp64
	32bit整数/OpenMP版	-laslcint -lasl -mkl=parallel
	64bit整数/OpenMP版	-laslcint64 -lasl64 -mkl=parallel
ASLSTAT	32bit整数版	-laslstatc -laslstat -mkl=sequential -lifcore -limf
	64bit整数版	-laslstatc64 -laslstat64 -mkl=sequential -lifcore -limf -lilp64
ASLQUAD	32bit整数/逐次版	-laslquadc++ -laslquad -lifcore -limf
	32bit整数/OpenMP版	-laslquadc++ -laslquad -lifcore -limf
dSFMT		-ldsfmt
SFMT		-lsfmt
SuperLU		-lsuperlu -mkl=sequential
Lis	逐次版	-llis_seq
	OpenMP版	-llis_omp
XBLAS		-lxblas
METIS		-lmetis

表 14. モジュール/ヘッダファイル（逐次版/OpenMP 版）

ライブラリ名	使用する言語	モジュール/ヘッダファイル
ASL	Fortran	不要
	C	asl.h
ASLSTAT	Fortran	不要
	C	aslstat.h
ASLQUAD	Fortran	aslquad.mod
	C++	aslquad.h
dSFMT	C	dSFMT.h
SFMT	C	SFMT.h
SuperLU	C	slu_sdefs.h (単精度実数版) slu_ddefs.h (倍精度実数版) slu_cdefs.h (単精度複素数版) slu_zdefs.h (倍精度複素数版)
Lis	Fortran	lisf.h
	C	lis.h
ARPACK	Fortran	不要
XBLAS	Fortran	不要
	C	blas_extended.h
METIS	C	metis.h

● MPI 版プログラムの場合

各ライブラリのリンクには、mpiifort、mpiicc コマンドを使用します。利用するプログラム言語に応じて、リスト 19、20 のようにリンクしてください。リンクオプションは使用するライブラリと言語に応じて指定します（表 15、表 16）。

NumericFactory でサポートしているライブラリを使用する場合、ライブラリによってはユーザプログラム側でモジュールファイルやヘッダファイルをインクルードする必要があります（表 17）。

Fortran から ASL または ASLSTAT の 64 ビット整数に対応したライブラリ利用する場合、コンパイル時に必ずオプション"-i8"を付けてコンパイルしてください。このオプションは、integer 型を 64 ビット整数と翻訳する Intel コンパイラのオプションです。

リスト 19. Fortran の場合（MPI 版）

```
[front1 ~]$ mpiifort source.f90  <リンクオプション>
```

リスト 20. C の場合（MPI 版）

```
[front1 ~]$ mpiicc source.c  <リンクオプション>
```

表 15. NumericFactory のリンクオプション（MPI 版 Fortran プログラムから利用する場合）

ライブラリ名		リンクオプション
ASL	32bit整数/逐次版	-lasl -mkl=sequential
	64bit整数/逐次版	-lasl64 -mkl=sequential
	32bit整数/OpenMP版	-lasl -mkl=parallel
	64bit整数/OpenMP版	-lasl64 -mkl=parallel
ASLSTAT	32bit整数版	-laslstat -mkl=sequential
	64bit整数版	-laslstat64 -mkl=sequential
ASLQUAD	32bit整数/逐次版	-laslquad
	32bit整数/OpenMP版	-laslquad
MUMPS	単精度実数版	-lsmumps -lmumps_common -lpord -lmetis -lmkl_scalapack_lp64 -lmkl_blacs_intelmpi_lp64 -mkl=sequential
	倍精度実数版	-ldmumps -lmumps_common -lpord -lmetis -lmkl_scalapack_lp64 -lmkl_blacs_intelmpi_lp64 -mkl=sequential
	単精度複素数版	-lcmumps -lmumps_common -lpord -lmetis -lmkl_scalapack_lp64 -lmkl_blacs_intelmpi_lp64 -mkl=sequential
	倍精度複素数版	-lzmumps -lmumps_common -lpord -lmetis -lmkl_scalapack_lp64 -lmkl_blacs_intelmpi_lp64 -mkl=sequential
Lis	逐次版	-llis_seq
	OpenMP版	-llis_omp
	MPI版	-llis_mpi
	MPI版+OpenMP版	-llis_omp_mpi
ARPACK		-larpack -mkl=sequential
PARPACK	MPI版	-lparpack -larpack -mkl=sequential
	BLACS版	-lparpack-blacs -larpack -lmkl_blacs_intelmpi_lp64 -mkl=sequential
XBLAS		-lxblas
METIS		-lmetis

表 16. NumericFactory のリンクオプション（MPI 版 C プログラムから利用する場合）

ライブラリ名		リンクオプション
ASL	32bit整数/逐次版	-laslcint -lasl -mkl=sequential -lifcore -limf
	64bit整数/逐次版	-laslcint64 -lasl64 -mkl=sequential -lifcore -limf -lilp64
	32bit整数/OpenMP版	-laslcint -lasl -mkl=parallel
	64bit整数/OpenMP版	-laslcint64 -lasl64 -mkl=parallel
ASLSTAT	32bit整数版	-laslstatc -laslstat -mkl=sequential -lifcore -limf
	64bit整数版	-laslstatc64 -laslstat64 -mkl=sequential -lifcore -limf -lilp64
ASLQUAD	32bit整数/逐次版	-laslquadc++ -laslquad -lifcore -limf
	32bit整数/OpenMP版	-laslquadc++ -laslquad -lifcore -limf
dSFMT		-ldsfmt
SFMT		-lsfmt
SuperLU		-lsuperlu -mkl=sequential
MUMPS	単精度実数版	-lsmumps -lmumps_common -lpord -lmetis -lmkl_scalapack_lp64 -lmkl_blacs_intelmpi_lp64 -mkl=sequential -lifcore -limf
	倍精度実数版	-ldmumps -lmumps_common -lpord -lmetis -lmkl_scalapack_lp64 -lmkl_blacs_intelmpi_lp64 -mkl=sequential -lifcore -limf
	単精度複素数版	-lcmumps -lmumps_common -lpord -lmetis -lmkl_scalapack_lp64 -lmkl_blacs_intelmpi_lp64 -mkl=sequential -lifcore -limf
	倍精度複素数版	-lzmumps -lmumps_common -lpord -lmetis -lmkl_scalapack_lp64 -lmkl_blacs_intelmpi_lp64 -mkl=sequential -lifcore -limf
Lis	逐次版	-llis_seq
	OpenMP版	-llis_omp
	MPI版	-llis_mpi
	MPI版+OpenMP版	-llis_omp_mpi
XBLAS		-lxblas
METIS		-lmetis

表 17. モジュール/ヘッダファイル（MPI 版）

ライブラリ名	使用する言語	モジュール/ヘッダファイル
ASL	Fortran	不要
	C	asl.h
ASLSTAT	Fortran	不要
	C	aslstat.h
ASLQUAD	Fortran	aslquad.mod
	C++	aslquad.h
dSFMT	C	dSFMT.h
SFMT	C	SFMT.h
SuperLU	C	slu_sdefs.h (単精度実数版) slu_ddefs.h (倍精度実数版) slu_cdefs.h (単精度複素数版) slu_zdefs.h (倍精度複素数版)
MUMPS	Fortran	smumps_struc.h (単精度実数版) dmumps_struc.h (倍精度実数版) cmumps_struc.h (単精度複素数版) zmumps_struc.h (倍精度複素数版)
	C	smumps_c.h (単精度実数版) dmumps_c.h (倍精度実数版) cmumps_c.h (単精度複素数版) zmumps_c.h (倍精度複素数版)
Lis	Fortran	lisf.h
	C	lis.h
ARPACK PARPACK	Fortran	不要
	C	不要
XBLAS	Fortran	不要
	C	blas_extended.h
METIS	C	metis.h

■ Intel 製ライブラリ

【機能概要】

表 18 で示したライブラリが利用可能です。

表 18. Intel 製ライブラリの機能概要

ライブラリ名	機能概要
数値演算ライブラリ MKL (Math Kernel Library)	工学、科学、金融向けの数値演算関数を提供する。最適化とマルチスレッド化されたライブラリです。
画像処理ライブラリ IPP (Integrated Performance Primitives)	マルチメディア、データ処理、通信／信号処理などのアプリケーションを作成するための、最適化された基関数から構成されるライブラリです。
マルチスレッドライブラリ TBB (Threading Building Blocks)	アプリケーションをマルチスレッド化する場合に最適な C++テンプレート・ライブラリです。

【ライブラリのリンク方法】

- MKL

以下の Intel Math Kernel Libraly リンクアドバイザーをご利用ください。Select Intel Product では Intel MKL 11.1 を選択してください。

http://software.intel.com/en-us/articles/intel-mkl-link-line-advisor

- IPP, TBB

各ライブラリのマニュアルをご覧ください。

■　マニュアル

■ NEC NumericFactory

ライブラリのマニュアルを並列コンピュータ上で提供しています。front.cc.tohoku.ac.jp にログインし、以下のディレクトリから閲覧してください。

```
/usr/ap/NFMAN200
```

■ Intel コンパイラ、Intel 製ライブラリ

コンパイラと各ライブラリのマニュアルを並列コンピュータ上で提供しています。front.cc.tohoku.ac.jp にログインし、以下のディレクトリから閲覧してください。

```
/opt/intel/composerxe/Documentation
```

7 章 プログラムについての補足

■ プログラムの使用メモリサイズ

プログラムを実行した際、使用するメモリサイズをバイト単位で表示します(リスト 21)。あらかじめ、必要とするメモリサイズが判断できます。なお、allocate 等で動的に確保するメモリサイズは含まれません。

【形式】　　**size**　実行形式ファイル名

リスト 21. 使用メモリサイズの表示

```
front1$ size a.out
1046912 + 140272 + 418928 = 1606112        1,606,112 バイト使用します
```

■ バイナリファイルの扱い(Fortran の場合)

センター以外のマシンで作成したバイナリファイルを扱う場合、注意が必要です。センターでは、並列コンピュータ LX 406Re-2 のエンディアン仕様は Big-Endian に設定しています。Little-Endian のバイナリファイルを扱う場合は、環境変数 F_UFMTENDIAN の設定をクリアします。設定はホームディレクトリの.chsrc やバッチリクエストファイルに記述します。

Little-Endian 仕様のファイルを扱う設定(csh 形式)

【形式】　　unsetenv F_UFMTENDIAN

■ メモリ使用量が 2GB を越える配列を扱う方法

コンパイルオプションに"-mcmodel=medium"または"-mcmodel=large"の指定とともに"-shared-intel"を指定してください。

・-mcmodel=medium　コードは IP 相対アドレス指定、データは絶対アドレス指定でアクセスされます
・mcmodel=large　コードもデータも絶対アドレス指定でアクセスされます

メモリ使用量が 2GB を越える配列を扱う場合のコンパイル方法

【形式】　　ifort -mcmodel=large -shared-intel　オプション ソースファイル名

■ アプリケーションプログラム

表 19 は、センターでサービスを行うアプリケーションプログラム一覧です。それぞれの詳しい利用方法は、以下の Web ページまたは本誌 85 ページの「アプリケーションサービスの紹介」をご参照ください。

http://www.ss.cc.tohoku.ac.jp/application/index.html

表 19. アプリケーションソフトウェアとサービスホスト

アプリケーションソフトウェア		サービスホスト
分子軌道計算ソフトウェア	Gaussian	front.cc.tohoku.ac.jp
反応経路自動探索プログラム	GRRM11	
統合型数値計算ソフトウェア	Mathematica	
汎用構造解析プログラム	Marc/Mentat	
対話型解析ソフトウェア	MATLAB	

8 章　利用負担金

■　利用負担金について

利用負担金は、演算負担経費、ファイル負担経費、出力負担経費、可視化負担経費の 4 つがあります（表 20、表 21）。スーパーコンピュータと並列コンピュータを利用すると、演算負担経費が発生します。

共有利用は利用するノード数と経過時間によって負担額が決定し、計算資源を利用者間で共有利用する利用形態です。

また、占有利用は計算資源を待ち時間なく占有して利用することができ、申請する利用期間によって負担額が決定する利用形態です。

請求書は四半期（3 ヶ月）ごとに、利用者を取りまとめている支払い責任者に発行します。最新の情報は以下の Web サイトをご覧ください。

```
http://www.ss.cc.tohoku.ac.jp/utilize/academic.html   （学術利用）
http://www.ss.cc.tohoku.ac.jp/utilize/business.html   （民間期間利用）
```

表 20.　基本利用負担金（大学・学術利用）

区　分	項　目	利用形態	負　担　額
演　算 負担経費	スーパー コンピュータ	共有	利用ノード数 1　(実行数、実行時間の制限有)　無料(備考2)
			利用ノード数 1～32 まで　経過時間 1 秒につき　0.06 円
			利用ノード数 33～256 まで　経過時間 1 秒につき　(利用ノード数-32)×0.002 円+0.06 円
			利用ノード数 257 以上　経過時間 1 秒につき　(利用ノード数-256)×0.0016 円+0.508 円
		占有	利用ノード数 32　利用期間 3 ヶ月につき　400,000 円
			利用期間 6 ヶ月につき　720,000 円
			利用ノード数 64　利用期間 3 ヶ月につき　720,000 円
			利用期間 6 ヶ月につき　1,300,000 円
			利用ノード数 128　利用期間 3 ヶ月につき　1,300,000 円
			利用期間 6 ヶ月につき　2,340,000 円
	並列 コンピュータ	共有	利用ノード数 1～6 まで　経過時間 1 秒につき　0.04 円
			利用ノード数 7～12 まで　経過時間 1 秒につき　0.07 円
			利用ノード数 13～18 まで　経過時間 1 秒につき　0.1 円
			利用ノード数 19～24 まで　経過時間 1 秒につき　0.13 円
		占有	利用ノード数 1　利用期間 3 ヶ月につき　160,000 円 (可視化システムの 20 時間無料利用を含む) 利用期間 6 ヶ月につき　320,000 円 (可視化システムの 40 時間無料利用を含む)
ファイル 負担経費	1TB まで無料、追加容量 1TB につき年額		3,000 円
出力 負担経費	大判プリンタによるカラープリント		フォト光沢用紙 1 枚につき　600 円 クロス 1 枚につき　1,200 円
可視化 機器室利用 負担経費	1 時間の利用につき		2,500 円

21.　基本利用負担金（民間機関利用）

区　分	項　目	利用形態	負　担　額
演　算 負担経費	スーパー コンピュータ	共有	利用ノード数 1　(実行数、実行時間の制限有)　無料(備考2)
			利用ノード数 1～32 まで　経過時間 1 秒につき　0.18 円
			利用ノード数 33～256 まで　経過時間 1 秒につき　(利用ノード数-32)×0.006 円+0.18 円
			利用ノード数 257 以上　経過時間 1 秒につき　(利用ノード数-256)×0.0048 円+1.524 円

		占有	利用ノード数 32	利用期間 3 ヶ月につき	1,200,000 円
				利用期間 6 ヶ月につき	2,160,000 円
			利用ノード数 64	利用期間 3 ヶ月につき	2,160,000 円
				利用期間 6 ヶ月につき	3,900,000 円
			利用ノード数 128	利用期間 3 ヶ月につき	3,900,000 円
				利用期間 6 ヶ月につき	7,020,000 円
	並列コンピュータ	共有	利用ノード数 1〜6 まで	経過時間 1 秒につき	0.12 円
			利用ノード数 7〜12 まで	経過時間 1 秒につき	0.21 円
			利用ノード数 13〜18 まで	経過時間 1 秒につき	0.3 円
			利用ノード数 19〜24 まで	経過時間 1 秒につき	0.39 円
		占有	利用ノード数 1	利用期間 3 ヶ月につき (可視化システムの 20 時間無料利用を含む)	480,000 円
				利用期間 6 ヶ月につき (可視化システムの 40 時間無料利用を含む)	960,000 円
ファイル負担経費	1TB まで無料、追加容量 1TB につき年額				9,000 円
出力負担経費	大判プリンタによるカラープリント			フォト光沢用紙 1 枚につき	1,800 円
				クロス 1 枚につき	3,600 円
可視化機器室利用負担経費	1 時間の利用につき				7,500 円

備考

1　負担額算定の基礎となる測定数量に端数が出た場合は、切り上げる。
2　負担額が無料となるのは専用のジョブクラスで実行されたものとし、制限時間を超えた場合には強制終了する。
3　占有利用期間は年度を超えないものとし、期間中に障害、メンテナンス作業が発生した場合においても、原則利用期間の延長はしない。また、占有利用期間中のファイル負担経費は 10TB まで無料とする。
4　ファイル負担経費については申請日から当該年度末までの料金とする。

■　利用負担金の確認方法

● プロジェクトコードの確認（project コマンド）

利用負担金はプロジェクトコード毎に合算されます。利用可能なプロジェクトコードの確認、デフォルトのプロジェクトコードの設定は project コマンドをご利用ください。プロジェクトコードの名称変更、追加などについては共同利用支援係までお問い合わせください。

リスト 22.　プロジェクトコードの確認

```
front1 $ project
使用可能なプロジェクトおよびキュー名は次のとおりです

利用者番号：（利用者番号）
```

```
----------------------  使用可能なプロジェクト一覧  ----------------------

  プロジェクト名称      :  運営費交付金
  プロジェクトコード  :  un0000
  使用可能なキュー名  :  sx32 sx64 ・・・

デフォルトのジョブ実行プロジェクトは un0000 です

    1.デフォルトプロジェクト変更      9.終了

何番の処理を選びますか ？
```

● プロジェクト課金情報表示（pkakin コマンド、ukakin コマンド）

コマンドを実行した利用者が利用可能なプロジェクトについて、負担額や請求情報を表示します。負担額は前日の午前 9 時までに終了したリクエストの利用額までが反映されています。

リスト 23. プロジェクトごとの負担額、請求情報の表示

```
front1 $ pkakin
   2 月 20 日  現在の利用負担金は次のとおりです

プロジェクトコード       累計負担額       調整額累計       固定費合計          請求済額   今期請求予定額（うち請求持越額）
un0000               15,404             0             0             0       15,404               0

支払責任者と経理担当者の所属（学校、学部）が異なる場合はセンターに連絡してください
支払費目の指定は各部局の経理担当者（学内の場合）、もしくはセンター会計係（学外の場合）へお伝えください

    1.負担金の明細表示   9.終了

何番の処理を選びますか ? 1
出力する年度を入力して下さい （ 1.今年度   2.前年度 ） : 1
開始月を入力してください : 2
終了月を入力してください : 2
出力先を選択してください  （ 1.画面   2.ファイル ） : 1

================================================================================
                                プロジェクト負担金明細情報

    2 月 20 日  現在の利用額および負担額は次のとおりです

  支払責任者          :  東北　太郎
  支払責任者番号      :  aaaaaa
```

```
  プロジェクト名称      :  運営費交付金
  プロジェクトコード    :  un0000

                        合計          演算 SX          演算 LX         ファイル           出力          可視化
  利用額              15,143          14,635              508              0              0              0
  負担額              15,143          14,635              508              0              0              0
-------------------------------------------------------------------------------------------------------------
月別利用額
         2 月             15,143
-------------------------------------------------------------------------------------------------------------
利用者別利用額
        abc000               21
        abc001                6
           :                  :
        abc020               27
        abc021            4,895
```

リスト 24. プロジェクトごとの利用額情報の表示

```
front1 $ ukakin

        利用者番号=（利用者番号）

 ----------------------  利 用 額  ----------------------
  支払責任者          :  東北　太郎
  支払責任者番号      :  aaaaaa
  プロジェクト名称    :  運営費交付金
  プロジェクトコード  :  un0000

   月       演算 SX       演算 LX       ファイル          出力          可視化          合計
   4              0             0              0             0              0             0
   5              0             0              0             0              0             0
   6              0             0              0             0              0             0
   7              0             0              0             0              0             0
   8              0             0              0             0              0             0
   9              0             0              0             0              0             0
  10              0             0              0             0              0             0
  11              0             0              0             0              0             0
  12            170            10              0             0              0           180
   1            546           162              0             0              0           708
   2              0             0              0             0              0             0
   3              0             0              0             0              0             0
 合計           716           172              0             0              0           888
```

● ジャーナル（利用明細）の抜粋（ulist コマンド、plist コマンド）

コマンドを実行した利用者のジャーナル情報、プロジェクトごとのジャーナル情報を抜粋して CSV 形式（カンマ区切り）のファイルに出力します。表 22 の項目が出力されます。

リスト 25. 利用者のジャーナル情報を抜粋

```
front1 $ ulist

出力する年度を入力して下さい ( 1.今年度  2.前年度 ) : 1
開始月を入力してください : 2
終了月を入力してください : 2
出力するファイル名を入力してください ( 省略時 : ulist.csv ) :

ホスト ID は次のとおりです
  20:SX  04:LX  02:front  05:ストレージ  06:プリンタ  08:可視化
```

リスト 26. プロジェクトのジャーナル情報を抜粋

```
front1 $ plist

出力する年度を入力して下さい ( 1.今年度  2.前年度 ) : 1
開始月を入力してください : 2
終了月を入力してください : 2
出力するファイル名を入力してください ( 省略時 : plist.csv ) :
プロジェクトを選択してください

    1.(un0000)      運営費交付金

何番のプロジェクトを選びますか ? 1

ホスト ID は次のとおりです
  20:SX  04:LX  02:front  05:ストレージ  06:プリンタ  08:可視化
```

表 22. ulist コマンドの出力項目

課金計上年月,ジャーナル連番,利用者番号,支払責任者番号,プロジェクトコード,ホスト ID,クラス ID,キュー名,投入日時,開始日時,終了日時,経過時間,使用ノード数,ユーザ CPU 時間,最大メモリサイズ,ベクトル時間,ベクトル演算率,ノード時間（使用量）,利用額

● ジャーナルの集計（usum コマンド、psum コマンド）

コマンドを実行した利用者のジャーナル情報、プロジェクトごとのジャーナル情報を集計して表示します。

リスト 27. 利用者のジャーナル情報を集計

```
front1 $ usum

出力する年度を入力して下さい ( 1.今年度  2.前年度 ) : 1
開始月を入力してください : 2
終了月を入力してください : 2

----------------------  利 用 情 報  ----------------------

利用者番号  プロジェクトコード  ホスト ID       経過時間合計  ユーザ CPU 時間合計     最大メモリサイズ  ノード時間(使用量)合計
abc001      un0000              02        1850:24:22          :  : 0             997            2:14:39
abc001      un0000              20         150:34:21       4141:51:07          59,649         4090:20:56
ホスト ID は次のとおりです
  20:SX  04:LX  02:front  05:ストレージ  06:プリンタ  08:可視化
```

リスト 28. プロジェクトのジャーナル情報を集計

```
front1 $ psum

出力する年度を入力して下さい ( 1.今年度  2.前年度 ) : 1
開始月を入力してください : 2
終了月を入力してください : 2
プロジェクトを選択してください

    1.(un0000)     運営費交付金

何番のプロジェクトを選びますか ? 1

----------------------  利 用 情 報  ----------------------

プロジェクトコード  ホスト ID       経過時間合計  ユーザ CPU 時間合計     最大メモリサイズ  ノード時間(使用量)合計
un0000              02         14354:22:54          :  : 0          33,856            9:51:13
un0000              04            25:16:44       238:45:30         118,672           47:59:41
un0000              05               :  : 0          :  : 0               0              :  : 1
un0000              06               :  : 0          :  : 0               0              :  :15
un0000              20           247:49:04       5930:43:04          61,414         6711:14:15

ホスト ID は次のとおりです
  20:SX  04:LX  02:front  05:ストレージ  06:プリンタ  08:可視化
```

■ 利用見込み額の設定

プロジェクトコードの請求先が学外の場合は、請求処理の都合上 2 月中旬以降、3 月末までの利用額を見込み金額として設定します。設定した金額は、1 月から 2 月中旬までの利用負担金に合算し、2 月末に請求いたします。見込み金額の設定は mikomi コマンドで行います。

リスト 29.　mikomi コマンドの例

```
front1 $ mikomi

プロジェクトを選択してください
プロジェクトコード : プロジェクト名称　支払責任者番号 : 支払責任者氏名
un0000        : 運営費交付金        aaaaaa      : 東北　太郎

プロジェクトコードを入力してください un0000

 2 月 1 日 現在の見込み額は次のとおりです

  支払責任者      : 東北　太郎
  支払責任者番号 : aaaaaa
  プロジェクト名称 : 運営費交付金
  プロジェクトコード: un0000

  見込み額              :         0 円
  今期請求予定額        :   300,000 円
  合計請求額(見込み額込) :   300,000 円

 1.見込み額の指定 2.削除 9.終了
 何番の処理を選びますか ? 1

  見込み額を入力してください (円) ? 200000

  登録してよろしいですか (yes/no) ? yes

 2 月 1 日 現在の見込み額は次のとおりです

  支払責任者      : 東北　太郎
  支払責任者番号 : aaaaaa
  プロジェクト名称 : 運営費交付金
  プロジェクトコード: un0000
  見込み額指定者 : 東北　太郎 (利用者番号 変更日: 2 月 1 日)

  見込み額              :     200,000 円
  今期請求予定額        :     500,000 円
  合計請求額(見込み額込) :     500,000 円

 見込み額の指定 2.削除 9.終了
 何番の処理を選びますか ?
```

9章　運用に関する情報、お問い合わせ

■　運用に関するお知らせ

大規模科学計算システムのホームページで利用方法やシステムの運用状況について、最新情報をお知らせします。計画停電による運用の停止スケジュールなどについても、こちらでお知らせしております。

URL http://www.ss.cc.tohoku.ac.jp/

■　利用申請

利用申請に関する詳細は、以下の Web ページをご覧ください。申請書の提出、お問合せは共同利用支援係までお願いいたします。

URL http://www.ss.cc.tohoku.ac.jp/utilize/index.html
メールアドレス　uketuke@cc.tohoku.ac.jp
電話番号　022-795-3406

■ ストレージの追加手順、申請

ホームディレクトリの追加容量:1TB につき年額 3,000 円です。追加容量の申請は、以下のページより「ストレージ【追加／削減】申請書」をダウンロードして必要事項を記入の上、郵送またはメールでお送りください。

URL http://www.ss.cc.tohoku.ac.jp/utilize/form.html

送付先(共同利用支援係)

住所　〒980-8578　宮城県仙台市青葉区荒巻字青葉６番３号　東北大学サイバーサイエンスセンターセンター・本館
メールアドレス　uketuke@cc.tohoku.ac.jp

利用に関するお問い合わせ

■ システムの利用方法についてのお問合せ

利用相談員が対応いたします。以下のメールアドレスまでお問合せください。

メールアドレス sodan05@cc.tohoku.ac.jp

■ 利用負担金についてのお問合せ

共同利用支援係までお問い合わせください。

メールアドレス uketuke@cc.tohoku.ac.jp
電話番号 022-795-6251

■ 請求書についてのお問合せ

請求書の発送等については会計係までお問い合わせください。

メールアドレス kaikei@cc.tohoku.ac.jp
電話番号 022-795-3405

10 章　おわりに

ジョブ管理システム NQSⅡを中心に利用方法の解説と、ライブラリの紹介をしました。研究の強力なツールとしてセンターのシステムをご活用いただければ幸いです。ご不明な点、ご質問等ございましたら、お気軽にセンターまでお問い合わせください。

［大規模科学計算システム］

SX-ACE でのプログラミング（並列化編）

— 共有並列化と分散並列化 —

林 康晴
日本電気株式会社

SX-ACE は、4 個の CPU コアを備えたノードを、ノード間スイッチ IXS（Inter-node X-bar Switch）により接続したスーパーコンピュータシステムです。SX-ACE のハードウェア性能を一つのプログラムから十分に引き出すには、プログラムの並列化により、複数の CPU コアを有効に活用する必要があります。SX-ACE は、並列処理環境としてプログラミング言語 Fortran、C、及び C++、並びに 通信ライブラリ MPI を用意しています。本稿は、第 1 章で並列処理の基本事項をご説明した後、第 2 章では SX-ACE の Fortran 95 コンパイラ FORTRAN90/SX（以後、単にコンパイラと記します）による共有並列化、第 3 章では SX-ACE の MPI ライブラリ MPI/SX による分散並列化を中心にご紹介します。

1. 並列処理

並列処理とは、一つの仕事を複数の小さな仕事に分割し、それらの仕事を複数同時に実行することです。例えば、2 重ループを四つの仕事に分割し、4 個の CPU コア上で並列実行すると、図 1.1 のようになります。この場合、外側の do j のループの 100 回の繰返しを 4 等分し、各 CPU コア上で並列に実行しています。

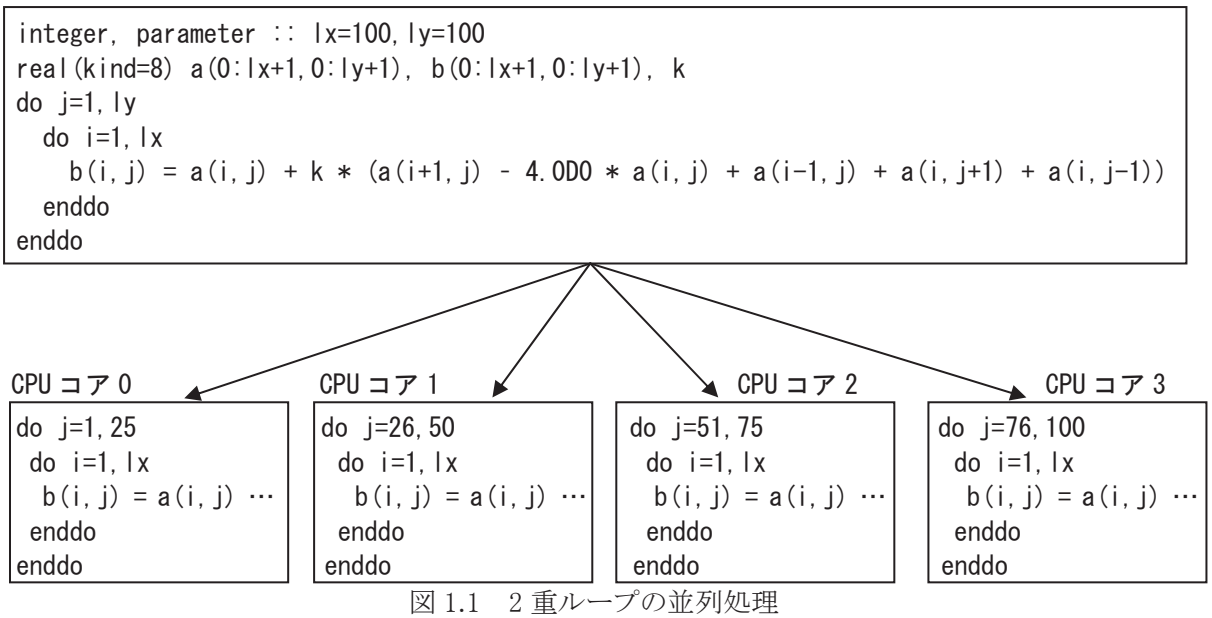


図 1.1　2 重ループの並列処理

1.1　並列化可能な条件

並列実行中、複数の仕事は、一般には実行タイミングの調整（同期）なしで、それぞれ独立に実行されます。そのため、複数の仕事内の各処理の間には、定まった実行順序はなく、一般には実行毎に異なる順序で実行されます。従って、ループが並列実行可能であるためには、ループの繰返しをどのような順序で実行しても実行結果が変わらないことが必要です。例えば、図 1.2 の二つのループ(a)、(b)は、いずれもループ中で確定される各配列要素 a(i) 及び a(i,1), (i=2,3,…,n) が、各ループ(a)、(b)の全繰返し中でそれぞれ 1 度しか出現しません。そのため、ループの繰返しをどのような順序で実行しても結果は同じ

になるので、並列実行可能です。一方、図 1.3 の(a)、(b)の場合、いずれもループの k 回目の繰返しで値が確定される配列要素 a(k)が、(a)の場合 k+1 回目の繰返しで、(b)の場合 k-1 回目の繰返しでそれぞれ引用されます。そのため、ループの繰返しの実行順序に依存して、確定前の値を引用するか、確定後の値を引用するかが変わってしまい、並列化することはできません。図 1.3 の(c)の場合も、ループの実行終了時に変数 ifound に残る値が、ループの繰返しの実行順序に依存して変わってしまうので、並列化することはできません。このように、ループを並列化するためには、一つの繰返しで確定したデータ要素は、他の繰返しでは確定も引用もしないようにループを記述する必要があります。図 1.3 の(d)のように、ループ外への飛越しを含むループも、ループの繰返しの実行順序に依存して飛越しのタイミングが変わってしまうので、並列化することはできません。

図 1.4 の二つのループ(a)、(b)は、スカラ変数 t, s のそれぞれを各ループの全繰返しで確定するので、図 1.3 の条件に該当し、一見並列化できないように思えます。しかし(a)の場合、変数 t は、ループの繰返し毎に新たに確定した値を、その繰返しの中だけで引用しています。そのため、ループ実行後に変数 t の値を引用しないのであれば、仕事毎に作業変数を確保して、並列実行中は、変数 t の代わりにその作業変数を参照すれば並列化可能です。また(b)のように、同一の変数に同一の演算を繰返し適用するパターンは集計計算と呼ばれます。集計計算の場合も、やはり仕事毎に作業変数を確保して、並列実行中はその作業変数に各仕事の局所的な計算結果を格納し、並列実行終了時に全仕事の値を集計することにより並列化できます。

多重ループの場合、並列化できるかどうかはループ毎に判断します。図 1.5 の内側の do i のループは並列化可能ですが、外側の do j のループは、左辺の a(i,j)と右辺の a(i,j-1)との間に図 1.3(a)と同様の依存関係があるので並列化できません。

```
integer, parameter :: n=100
real(kind=8), dimension(n) :: a, b
do i=2,n
  a(i) = b(i) + b(i-1)
enddo
```

(a)

```
integer, parameter :: n=100
real(kind=8), dimension(n,n) :: a
do i=2,n
   a(i,1) = a(i,2) + a(i-1,2)
enddo
```

(b)

図 1.2　並列化できるループ

```
integer, parameter :: n=100
real(kind=8), dimension(n) :: a
do i=2,n
  a(i) = a(i) + a(i-1)
enddo
```

(a) 依存

```
integer, parameter :: n=100
real(kind=8), dimension(n) :: a
do i=1,n-1
   a(i) = a(i) + a(i+1)
enddo
```

(b) 逆依存

```
integer, parameter :: n=100
real(kind=8), dimension(n) :: a
integer ifound
do i=1,n
  if(a(i).ge.0) ifound = i
enddo
```

(c) 出力依存

```
integer, parameter :: n=100
real(kind=8), dimension(n) :: a
do i=1,n
  if(a(i).ge.0) goto 100
enddo
100 continue
```

(d) 制御依存

図 1.3　並列化できないループ

```
integer, parameter :: n=100
real(kind=8) :: a(n), b(n), t
do i=1,n-1
  t = a(i) + a(i+1)
  b(i) = t
enddo
```

(a) 繰返し内作業変数

```
integer, parameter :: n=100
real(kind=8) :: a(n), s
do i=1,n
  s = s + a(i)
enddo
```

(b) 集計計算

図 1.4　作業変数の使用により並列化できるループ

```
integer, parameter :: n=100
real(kind=8), dimension(n,n) :: a
do j=2,n
  do i=1,n
    a(i,j) = a(i,j-1) + a(i,j)
  enddo
enddo
```

図 1.5　内側のみ並列化できる多重ループ

1.2 並列化の効果

一つのプログラムを N 個の CPU コア上で並列実行した場合、最大 N 倍のスピードアップ(実行時間の短縮)が期待できます。このスピードアップを最大にするには、まず、どれだけ多くの部分を並列化できたか(並列化率)が重要です。これは、図 1.6 のように逐次実行時間の 10 %分の仕事が並列化できない場合、残りの仕事をどれだけ多くの小さな仕事に分割して並列化しても、スピードアップは 10 倍未満になってしまうことからも分かります。

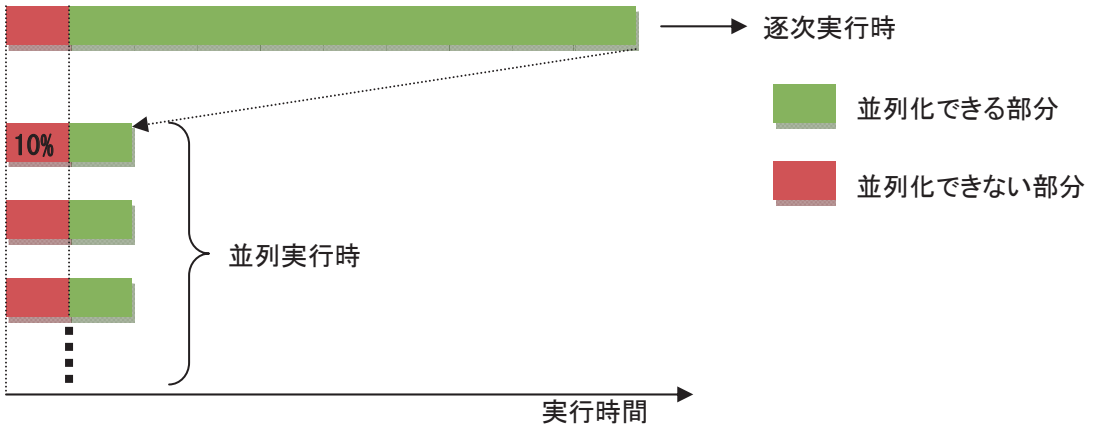


図 1.6　並列化率とスピードアップ(並列化率 90 %・スピードアップは 10 倍未満)

また、仕事を複数の小さな仕事に分割するための処理や、仕事間のデータのやり取り(通信)及び同期といった逐次実行時には必要なかった処理(オーバヘッド)をできるだけ小さくすることも重要です。これは、図 1.7 のように逐次実行時間の 10 %分のオーバヘッドが並列化により発生すると、スピードアップはやはり 10 倍未満になってしまうことからも分かります。特に、実行時間(粒度)の小さなループを並列化するような場合、オーバヘッドの方が大きすぎ、並列化によりかえって遅くなる場合もあります。オーバヘッドを削減するには、できるだけ外側のループで並列化する、といった方法により、各仕事の粒度をできるだけ

大きくすると効果的です。

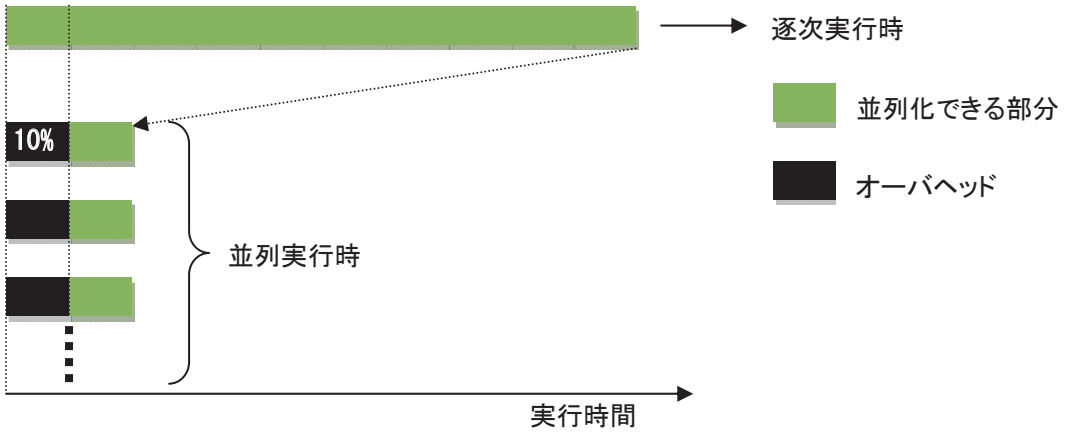


図 1.7　オーバヘッドとスピードアップ（オーバヘッド 10 %・スピードアップは 10 倍未満）

さらに、各仕事の実行時間のバランス（負荷バランス）をできるだけ均等化することも重要です。これは、図 1.8 のように複数の仕事のうち一つの仕事の実行に逐次実行時の 10 %分の時間がかかってしまったとすると、残りの仕事をどれだけ高速化しても、スピードアップは 10 倍以下になってしまうことからも分かります。

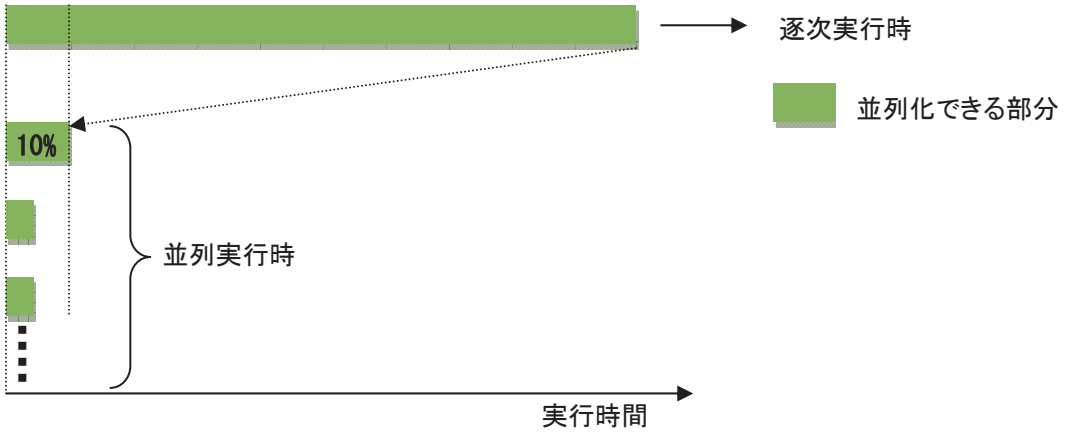


図 1.8　負荷バランスとスピードアップ（スピードアップは 10 倍以下）

このように、並列プログラムにおいて高いスピードアップを達成するには、高い並列化率・低いオーバヘッド・均等な負荷バランスの 3 点が重要です。

1.3　経過時間と CPU 時間

並列処理は、一つの仕事を複数の小さな仕事に分割して、複数の CPU コア上で同時に実行することにより、仕事の実行時間（経過時間）を減らします。一方、全仕事を実行するための全 CPU コアの処理時間の総計（CPU 時間）を減らすことはできません。実際には、一つの仕事を複数の小さな仕事に分割するための処理や仕事間の同期などのオーバヘッドも発生するので、CPU 時間は並列化によりむしろ増加することに注意してください。

1.4　経過時間とチューニング

並列化の主たる目的であるプログラム実行の経過時間短縮のためには、アルゴリズムの工夫、高速な

ライブラリの使用、及びベクトル化の促進などにより、逐次プログラムとしての性能を並列化の前に十分チューニングしておくことが重要です。並列化によるスピードアップは、使用する CPU コアの個数が上限となりますが、逐次プログラムのチューニングによる高速化は、場合により数 10～数 100 倍に達することもあり、さらに CPU 時間も削減できるからです。

2. 共有並列化

一般に、コンピュータのプログラムは、プロセスにより実行されます。実行中のプロセスは、そのメモリ空間にプログラムのデータを格納しており、現在の実行状態をもっています。プロセスは、複数のタスクから構成することもできます。一つのプロセス中の複数のタスクは、それぞれ独自の実行状態をもち、別々の仕事を実行できますが、プロセスのメモリ空間は全タスクが共有します。

共有並列化とは、一つのプロセス中の複数のタスクによる並列処理のことです。SX-ACE の各ノード内では、主記憶装置を共有する 4 個の CPU コア上で実行されるタスクに仕事を割り当て、共有並列化を行うことができます。

例として、図 1.1 の 2 重ループの共有並列化を考えます。第 1 章と同様に、外側の do j のループを四つの仕事に分割して別々のタスクに割り当てます。この場合、データ領域に関しては、図 2.1 のように、2 次元配列 a, b を do j のループに対応する 2 次元目で分割し、分割された各部分領域の処理を各タスクが担当することになります。ここで、例えば j=25 の繰返しを担当するタスク 0 は、タスク 1 の担当領域である配列要素 a(i,26), (i=1,2,…,lx) の値を引用する、といったように、配列の分割境界部分の処理では、自タスクの担当領域外の配列要素を参照する必要があります。しかし、各タスクは、メモリ上に配置された 2 次元配列 a, b の全領域を直接参照できるので、特に問題はありません。一方、DO 変数 i, j については注意が必要です。並列実行中、各タスクは変数 i, j をそれぞれ独自のタイミングで参照します。そのため、配列 a, b と同様に、変数 i, j の領域を全タスクが共有すると、あるタスクが確定した変数 i, j の値を別のタスクが引用してしまう可能性があり、実行が正しく行えません。従って、並列実行中に各タスクが値を確定する変数は、図 2.2 のようにそれぞれのタスクが専用の作業領域を割り付けて参照する必要があります。2 次元配列 a, b のような全タスクがメモリ領域を共有するデータをタスク間共有データ、DO 変数 i, j のような各タスクが専用のメモリ領域を割り付けるデータをタスク固有データと呼びます。

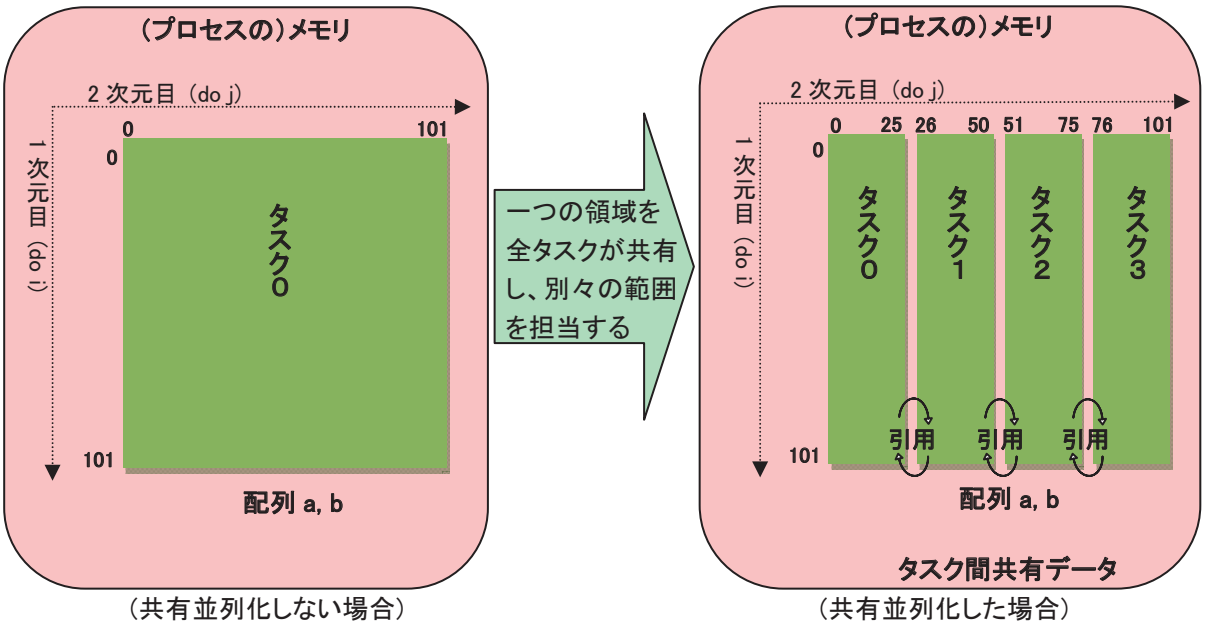


図 2.1　各タスクが担当する配列 a, b の領域

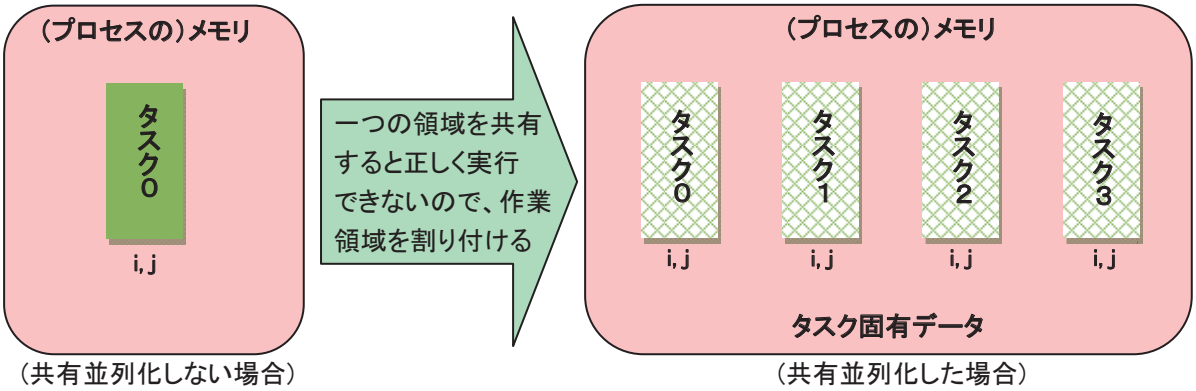


図 2.2　各タスク専用の DO 変数 i, j 用の作業領域

2.1 自動並列化

並列化を行う場合、並列実行しても結果が不正にならないことを保証するために、通常はデータの依存関係の解析を行い、細心の注意を払ってプログラムの変形や指示文の挿入を行う必要があります。しかし、コンパイラの自動並列化機能を使用すると、コンパイラオプション-P auto を指定するだけで、それらの作業をコンパイラが自動的に行います。

コンパイラは、まずプログラムを解析して並列実行可能なループや文の集まりを抽出し、次にそれらを複数の仕事に分割してタスクに割り当て、経過時間を短縮します。また、タスク間共有データ・タスク固有データの割付けも自動的に行います。コンパイラの自動並列化対象は、図 2.3 のとおりです。

対象となる構文	DO ループ、配列式
対象ループ中に書ける文	代入文、IF 構文、GOTO 文、CONTINUE 文、CALL 文、CASE 構文
対象となる演算	四則演算、べき乗演算、論理演算、関係演算、型変換、組込み関数

図 2.3 コンパイラの自動並列化対象

2.2 コンパイラの最適化

ここでは、共有並列化の効果を高めるためにコンパイラが行う最適化をいくつかご紹介します。

2.2.1 ループ変形

ループ融合・ループ一重化などのループ変形による最適化が可能な場合、最適化後に共有並列化を行い、共有並列実行時の仕事の粒度を最大化します。図 2.4 にループ融合の例を示します。

```
integer, parameter :: n=100
real(kind=8), dimension(n) :: a, b, c, d
do i=1,n
  c(i) = a(i) + sqrt(b(i))
enddo
do i=1,n
  d(i) = a(i) + sqrt(b(i))
enddo
```

（ループ融合前）

```
do i=1,n
  c(i) = a(i) + sqrt(b(i))
  d(i) = a(i) + sqrt(b(i))
enddo
```

（ループ融合後）

図 2.4　ループ融合の例

2.2.2 条件並列化

並列化できるかどうか又は並列化により高速化できるかどうかが翻訳時に判断できない場合、実行時に依存関係やループ長を調べて、共有並列化するかどうかを選択できるよう条件並列化を行います。図2.5では、条件並列化後のIF構文の論理式 nx*ny > n によって、共有並列化により高速化するのに十分なループ長があるかどうかを判定しています。この際、ループ中の各演算の演算コストに基づいて、共有並列化の効果が期待できる値(n)を自動的に算出し、条件並列化を行います。また、論理式 id-ic==0 .or. abs(id-ic)>=nx によって、ループ中で確定される配列要素 y(ic+i)と y(id+i)の領域に重なりがなく、並列化可能であることを判定しています。

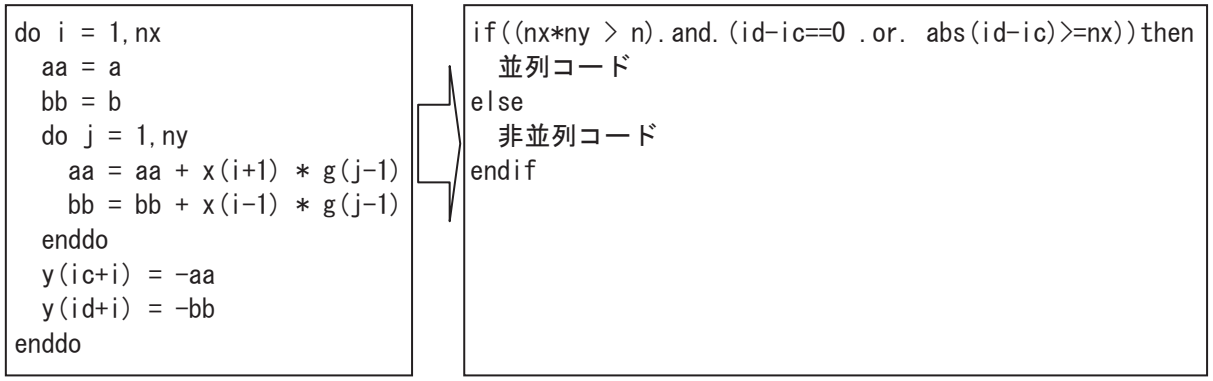


図 2.5　条件並列化の例

2.3 自動並列化促進のためのプログラミング

コンパイラオプション-P auto の指定だけでは自動並列化できない場合でも、コンパイラ指示行の挿入やプログラムの修正により自動並列化できる場合もあります。ここでは、主な並列化用指示行とプログラムの修正例をご紹介します。

2.3.1 並列化指示行

コンパイラ指示行の書式は、図2.6のとおりです。

Fortran の場合：

```
!CDIR コンパイラ指示オプション
```

C・C++の場合：

```
#pragma  cdir コンパイラ指示オプション
```

図 2.6　コンパイラ指示行の書式

自動並列化のために用意されている主なコンパイラ指示オプションは、次のとおりです。

- concur [{ by=整数 | for=整数 }] / noconcur
  直後のループを自動並列化の対象とする / しないを指定します。コンパイラ指示オプション concur に、省略可能なサブオプション by=整数 を指定すると、各タスクに割り当てるループの繰返し回数を指定できます。省略可能なサブオプション for=整数 を指定すると、ループをいくつのタスクに分割するかを指定できます。コンパイラ指示オプション noconcur は、粒度が小さく共有並列化すると

かえって性能が低下するループに指定します。

- inner / noinner
  内側ループ又は一重ループを自動並列化の対象とする / しないを指定します。多重ループの内側ループは、既定値では自動並列化の対象にならないので、共有並列化したい場合、コンパイラ指示オプション inner を指定します。一重ループは、共有並列化の効果が翻訳時に不明の場合、自動並列化の対象になりませんが、コンパイラ指示オプション inner を指定することにより自動並列化の対象とすることができます。

- nosync
  ループ中で参照される配列に依存がないことを指定します。依存関係が翻訳時には分からないので自動並列化できないときでも、依存がないことを利用者が分かっている場合、コンパイラ指示オプション nosync によりそれをコンパイラに教えてやることによって、自動並列化できる場合があります。

- cncall
  Fortran の組込み関数以外の手続引用を含むループは、既定値では自動並列化の対象になりませんが、並列化しても問題ないことを利用者が分かっている場合、コンパイラ指示オプション cncall を指定すると自動並列化の対象とすることができます。自動並列化されたループ中で引用される手続の仮引数は、原則としてタスク間共有データとなるので、そのような仮引数を手続中で確定した結果、図 1.3 のような依存関係が発生すると結果が不正となることに注意してください。

2.3.2 並列化のためのソース変形

図 2.7(a)の 2 重ループは、外側の do j のループには依存があるので並列化できませんが、コンパイラ指示オプション inner を内側の do i のループに指定すると、内側ループで自動並列化されます。ただし、内側ループを並列化すると、並列化のオーバヘッドが外側ループの繰返し回数分発生してしまいます。このような場合、図 2.7(b)のように内側ループと外側ループを利用者が交換すると、外側ループで自動並列化され並列化のオーバヘッドを削減できます。

```
integer, parameter :: n=10000,m=10000
real(kind=8), dimension(n,m) :: a
do j=1,m-1
   do i=1,n
      a(i,j) = a(i,j+1) + a(i,j)
   enddo
enddo
```

(a) ループ交換前

```
do i=1,n
 do j=1,m-1
   a(i,j) = a(i,j+1) + a(i,j)
 enddo
enddo
```

(b) ループ交換後

図 2.7　ループ交換の例

仮引数は原則としてタスク間共有データとなるので、図 2.8(a)のループネストは、タスク間共有データである仮引数 w に関するデータ依存により、外側の do j のループを自動並列化できません。このような場合、図 2.8(b)のように仮配列引数 w を次元拡張し、do j のループの繰返し毎に異なる要素を参照するように修正すれば、自動並列化可能となります。この際、仮引数 w に対応する実引数にも同様の修正が必要となることに注意してください。また、手続 sub の引用元が仮引数 w に対応する実引数を参照しておらず、配列 w は単なる作業変数として利用されている場合は、図 2.8(c)のように仮引数 w を手続 sub の局所変数

に変更すると、コンパイラによりタスク固有データとして割り付けられるので、自動並列化できます。

```
subroutine sub(a,b,w)
parameter(n=100,m=100)
real(kind=8) a(n,m),b(n,m),w(0:n)
do j=1,m
  do i=1,n
    w(i) = a(i,j) / dble(n)
  enddo
  do i=1,n
    b(i,j) = b(i,j) + (w(i-1)+w(i))/2.0
  enddo
enddo
```

(a) 仮引数 w に関する依存

```
subroutine sub(a,b,w)
parameter(n=100,m=100)
real(kind=8) a(n,m),b(n,m),w(0:n,m)
do j=1,m
 do i=1,n
   w(i,j) = a(i,j) / dble(n)
 enddo
 do i=1,n
   b(i,j) = b(i,j)+(w(i-1,j)+w(i,j))/2.0
 enddo
enddo
```

(b) 次元拡張後

```
subroutine sub(a,b)
parameter(n=100,m=100)
real(kind=8) a(n,m),b(n,m),w(0:n)
do j=1,m
  do i=1,n
    w(i) = a(i,j) / dble(n)
  enddo
  do i=1,n
    b(i,j) = b(i,j) + (w(i-1)+w(i))/2.0
  enddo
enddo
```

(c) 局所変数化後

図 2.8　仮引数の修正により自動並列化可能となる例

2.4 自動並列化の際のメモリ消費量

局所変数は、コンパイラによりタスク固有データとして割り付けられます。そのため、巨大な局所配列を宣言すると、その局所配列がタスク数個分割り付けられ、メモリ不足によりプログラム実行が失敗する場合があります。大きな配列は大域変数(Fortranの場合、モジュール変数又は共通ブロック変数)としてください。

クロス環境において、sxsize コマンドを、オプション-l に実行タスク数を指定した上で、実行ファイルに対して適用すると、図 2.9 のように、各タスクに対して確保されるスタックである logical task region のメモリ量を表示させ、確認することができます。

```
%> sxsize -f -l 4 a.out
11256208(.text) + … + 9714896(logical task region) * 4 = …
```

図 2.9　sxsize コマンド実行例 (4 タスク実行の場合)

2.5 OpenMP による共有並列化

コンパイラは、OpenMP による手動並列化も用意しています。OpenMP は、the OpenMP Architecture Review Board (ARB)[1] によって策定された共有並列化のためのオープンな API であり、共有並列化のデファクト標準として広く使用されています。OpenMP では、主として、ソースプログラムに OpenMP 指示文を挿入することによって共有並列化を行います。利用者は、OpenMP 指示文に必要に応じて指示句を追加し、並列化方法を詳細に制御できます。

SX-ACE で OpenMP を使用する場合、コンパイラオプション-P openmp を指定してください。

2.5.1 OpenMP 使用時の注意点

OpenMP と自動並列化は併用できないので、共有並列化したい部分には全て OpenMP 指示文を指定してください。

自動並列化の場合、コンパイラがプログラムを解析し並列ループ間の不必要な同期を自動的に抑制します。一方 OpenMP による共有並列化時には、nowait 指示句を指定しない限り並列ループの直後に毎回同期が生成されます。同期が必要ない場合には、nowait 指示句を指定すると共有並列化のオーバヘッドを削減できます。

また自動並列化の場合、図 1.4 のような作業変数や集計変数は、コンパイラによって自動的に認識され適切に処理されます。一方 OpenMP による共有並列化時には、DO 変数以外の作業変数及び集計変数をそれぞれ private 指示句及び reduction 指示句中に指定する必要があります。

OpenMP の parallel do 指示文をループに指定して共有並列化した例を図 2.10 に示します。繰返し内作業変数 t を private 指示句中に、集計演算子+ 及び集計変数 s を reduction 指示句中に指定していることに注意してください。なお、DO 変数 i は、この場合既定値でタスク固有データとなるので、private 指示句を指定する必要はありません。

```
integer, parameter :: n=100
real(kind=8) :: a(n), t, s
!$omp parallel do private(t) reduction(+: s)
do i=1,n-1
  t = a(i) + a(i+1)
  s = s + t
enddo
```

図 2.10　OpenMP による並列化例

2.6 性能解析

2.6.1 プログラム特性情報（PROGINF）

逐次（ベクトル）プログラムの場合と同様に、共有並列プログラムの場合も、プログラム特性情報（PROGINF）が使用できます。PROGINF は、プログラム実行時に環境変数 F_PROGINF（Fortran の場合）又は C_PROGINF（C・C++の場合）を、YES 又は DETAIL と設定することによって採取できます。図 2.11 に、環境変数 F_PROGINF を DETAIL に設定して共有並列実行した場合の出力例を示します。ここで図中の※は、共有並列実行時に固有の情報です。

共有並列プログラムの性能を分析する場合、PROGINF 中の Conc. Time に着目してください。図 2.11 の場合、プログラムは 4 個の CPU コア上でそれぞれ約 12.6 秒ずつ実行されたことが分かります。Conc. Time が CPU コア間でほぼ均等となっているので、うまく共有並列化されていると判断できます。逆に図 2.12 のように、Conc. Time (>= 1)だけ実行時間が大きく残りが小さい場合は、プログラムが十分共有並列化できていないか、又は負荷バランスが悪い、と判断できます。このようなプログラムは、より多くのループ

を共有並列化できるよう修正するか、又はコンパイラ指示オプション concur by=整数 などを指定して負荷バランスを均等化すると、経過時間を短縮できる可能性があります。

```
    ******  Program Information  ******
Real Time (sec)                :          12.763395
User Time (sec)                :          50.708166
Sys  Time (sec)                :           0.074710
Vector Time (sec)              :          40.632103
Inst. Count                    :       11790881218.
V. Inst. Count                 :        5299726673.
V. Element Count               :     1356647707664.
V. Load Element Count          :       24425310348.
FLOP Count                     :      687380352458.
MOPS                           :       26882.038333
MFLOPS                         :       13555.614543
MOPS   (concurrent)            :      107208.756514  ※
MFLOPS (concurrent)            :       54061.398206  ※
A. V. Length                   :         255.984467
V. Op. Ratio (%)               :          99.523808
Memory Size (MB)               :        4544.000000
Max Concurrent Proc.           :                4.  ※
  Conc. Time(>= 1) (sec)       :          12.714809  ※ 1個以上の CPU コアで実行した時間
  Conc. Time(>= 2) (sec)       :          12.713175  ※ 2個以上の CPU コアで実行した時間
  Conc. Time(>= 3) (sec)       :          12.688490  ※ 3個以上の CPU コアで実行した時間
  Conc. Time(>= 4) (sec)       :          12.591691  ※ 4個以上の CPU コアで実行した時間
Event Busy Count               :                0.  ※
Event Wait (sec)               :           0.000000  ※
Lock Busy Count                :                0.  ※
Lock Wait (sec)                :           0.000000  ※
Barrier Busy Count             :                0.  ※
Barrier Wait (sec)             :           0.000000  ※
MIPS                           :         232.524308
MIPS (concurrent)              :         927.334513  ※
I-Cache (sec)                  :           0.001616
O-Cache (sec)                  :           0.015225
Bank Conflict Time
  CPU Port Conf. (sec)         :           0.005777
  Memory Network Conf. (sec)   :           0.106487
ADB Hit Element Ratio (%)      :          66.689853
```

図 2.11　共有並列実行時の PROGINF

```
Max Concurrent Proc.           :                4.
  Conc. Time(>= 1) (sec)       :          74.154168
  Conc. Time(>= 2) (sec)       :           8.549322
  Conc. Time(>= 3) (sec)       :           8.292376
  Conc. Time(>= 4) (sec)       :           8.071275
```

図 2.12　Conc. Time の例

2.6.2 簡易性能解析機能(ftrace 機能)

逐次(ベクトル)プログラムの場合と同様に、共有並列プログラムの場合も、ftrace 機能によって手続単位又は利用者が指定した区間単位の詳細な性能情報が取得できます。

基本的な使用方法は逐次プログラムの場合と同様ですが、コンパイラは、共有並列化する各ループネストを切り出して、それらを独立した手続に変形します。そのため共有並列化された各ループネストは、ftrace の表示上は独立した一つの手続として表示されることに注意してください。切りだされた各ループネストは、ソースプログラム中の出現順に、元の手続名に加えて$1, $2,…とサフィックスが付いた名前の手続として表示されます。図 2.13 に、共有並列実行時の ftrace の表示例を示します。$のついた手続

gauss$1 及び cauchy$2 は、それぞれ手続 gauss の最初に出現する共有並列化されたループネスト、手続 cauchy の 2 番目に出現する共有並列化されたループネストの性能情報です。-micro1, -micro2, … は、各タスクの性能情報です。gauss のような$のついていない手続名は、手続 gauss 中の共有並列化されなかった部分の性能情報です。

```
PROG.UNIT FREQUENCY EXCLUSIVE      AVER.TIME   MOPS MFLOPS V.OP  AVER.  I-CACHE O-CACHE…
                    TIME[sec](%)   [msec]                 RATIO V.LEN  MISS    MISS  …
gauss$1      7872   2.536( 94.6)     0.322   4487.1 2099.2 99.41 233.7  0.0009 0.2196 …
 -micro1     1968   0.644( 24.8)     0.337   4306.4 2014.4 99.40 233.6  0.0000 0.0574 …
 -micro2     1968   0.616( 23.0)     0.313   4605.5 2154.8 99.42 233.8  0.0009 0.0541 …
 -micro3     1968   0.628( 23.4)     0.319   4522.0 2115.6 99.41 233.8  0.0000 0.0541 …
 -micro4     1968   0.628( 23.4)     0.319   4527.5 2118.2 99.41 233.8  0.0000 0.0540 …
cauchy$2        4   0.104(  3.9)    26.104    230.5   38.4  0.08 228.4  0.0000 0.0000 …
 -micro1        1   0.026(  1.0)    26.138    230.2   38.3  0.08 228.4  0.0000 0.0000 …
 -micro2        1   0.026(  1.0)    26.092    230.6   38.4  0.08 228.4  0.0000 0.0000 …
 -micro3        1   0.026(  1.0)    26.093    230.6   38.4  0.08 228.4  0.0000 0.0000 …
 -micro4        1   0.026(  1.0)    26.093    230.6   38.4  0.08 228.4  0.0000 0.0000 …
gauss           1   0.033(  1.2)    33.191   1246.9  514.9 96.90 226.3  0.0011 0.0030 …
                :
```

図 2.13　共有並列実行時の ftrace

3. 分散並列化

分散並列化とは、それぞれ独自のメモリ空間をもつ複数のプロセスによる並列処理のことです。SX-ACE では、一つ又は複数のノードの CPU コア上で実行されるプロセスに仕事を割り当て、分散並列化を行うことができます。

例として、図 1.1 の 2 重ループの分散並列化を考えます。第 1 章と同様に、外側の do j のループを四つの仕事に分割して別々のプロセスに割り当てます。この場合、データ領域に関しては、図 3.1 のように、2 次元配列 a, b を do j のループに対応する 2 次元目で分割し、分割された各部分領域の処理を各プロセスが担当することになります（分割配置・1 次元分割）。ここで、例えば j=25 の繰返しを担当するプロセス 0 は、プロセス 1 の担当領域である配列要素 a(i,26), (i=1,2,…,lx) の値を引用する、といったように、配列の分割境界部分の処理では、自プロセスの担当領域外の配列要素を参照する必要があります。ところが各プロセスは、他のプロセスに割り付けられた配列の領域を直接参照できないので、通信を行って必要な配列要素の値を取得しなければなりません。このように分散並列化の場合、共有並列化と異なり、他のプロセスの担当範囲のデータを参照するために、必要に応じて通信が必要です。一方 DO 変数 i, j については、図 3.2 のように各プロセスがそれぞれ専用の領域を割り付ける（重複配置）と、並列実行中、各プロセスは変数 i, j をそれぞれ独立に参照できるので通信の必要はありません。すなわち、分散並列実行時にどのような通信が必要となるかは、各データ要素をどのようにプロセスに割り付けるか（データマッピング）及び一つの仕事をどのように複数の小さな仕事へと分割してプロセスに割り当てるか（計算マッピング）に依存します。このように分散並列化においては、データマッピング、計算マッピング、及び通信の 3 点を全て考慮してプログラムする必要があります。

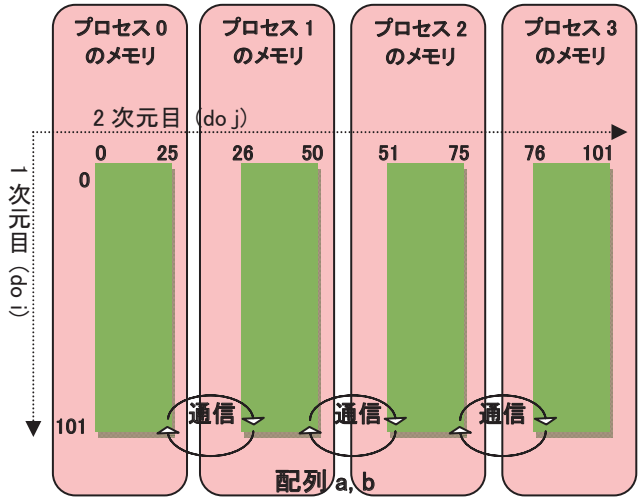


図 3.1　配列 a, b の分割配置（1 次元分割）

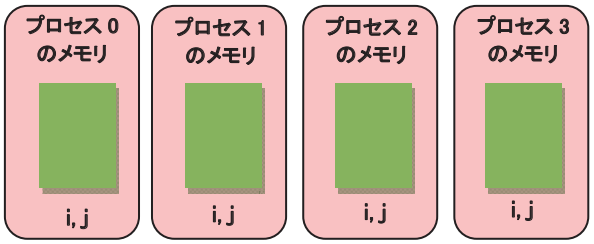


図 3.2　DO 変数 i, j の重複配置

3.1 データマッピングと性能

再び図 1.1 の 2 重ループの分散並列化を考えます。今度は図 3.3 のように、外側・内側のループをそれぞれ二つの仕事に分割してみます。この場合、データ領域に関しては、図 3.4 のように 2 次元配列 a,b を1次元目・2 次元目共に分割し、分割された各部分領域の処理を各プロセスが担当することになります（分割配置・2 次元分割）。ここでも、例えば j=50 の繰返しを担当するプロセス 0 は、プロセス 2 の担当領域である配列要素 a(i,51), (i=1,2,…,50) の値を引用する、といったように、配列の分割境界部分の処理では、自プロセスの担当領域外の配列要素を参照するために通信が必要です。

これを前節の外側ループだけによる分散並列化（1 次元分割）と比較してみます。通信に関しては、1 次元分割の場合、送受信回数はそれぞれ 2 回（2 次元目の上下方向）、送受信量はそれぞれ 1 次元目の寸法である lx*2 となります。一方 2 次元分割の場合、送受信回数はそれぞれ 4 回（1・2 次元目それぞれの上下方向）、送受信量はそれぞれ (lx/1 次元目分割数+ly/2 次元目分割数)*2 となります。従って、1 次元分割と比べると、2 次元分割の方が通信回数は多いですが、通信量はプロセス数が増えるほど少なくなります。1 回の通信時間は、典型的にはセットアップ時間+通信量/通信バンド幅と表せます。そのため、通信量が少なく（配列の寸法が小さく）セットアップ時間の影響が大きい場合には 1 次元分割の方が通信コストは少なく、逆に通信量が多く（配列の寸法が大きく）通信量の影響が大きい場合には 2 次元分割の方が通信コストは少なくなると考えられます。一方、各プロセスの演算性能を比較すると、ベクトル長は内側の do i のループに対応する配列の 1 次元目の寸法に依存します。そのため、1 次元目をプロセス間に分割することによってベクトル長が短くなりすぎると、ベクトル演算の効率が低下します。従って、配列の寸法がそれほど大きくない場合、2 次元目だけを分割したほうが演算時間は短い可能性があります。このように、これら二つの分散並列化方法のどちらが経過時間を短縮できるかは、プロセス数や配列の寸法に依存して変わります。分散並列化時には、通信コストや並列化後の演算性能を考慮して、データマッピングを決定することが重要です。

```
integer, parameter :: lx=100, ly=100
real(kind=8) a(0:lx+1,0:ly+1), b(0:lx+1,0:ly+1), k
do j=1,ly
  do i=1,lx
    b(i,j) = a(i,j) + k * (a(i+1,j) - 4.0D0 * a(i,j) + a(i-1,j) + a(i,j+1) + a(i,j-1))
  enddo
enddo
```

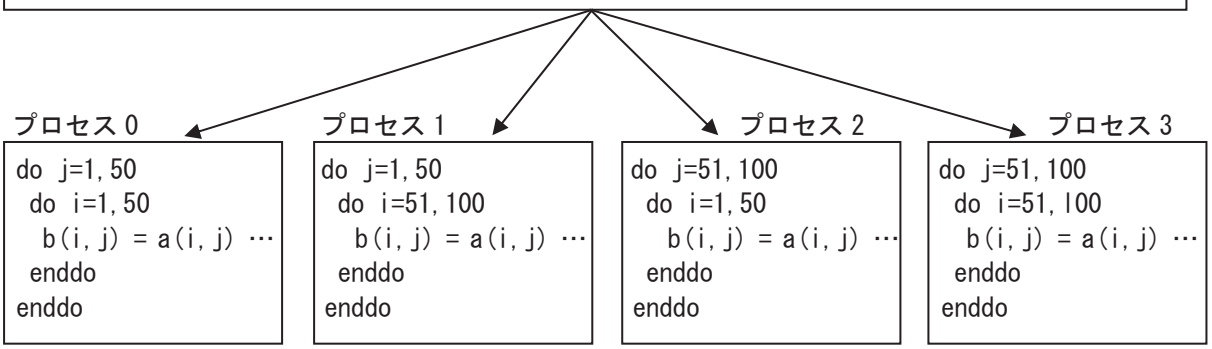


図 3.3　二つのループを共に並列化する例

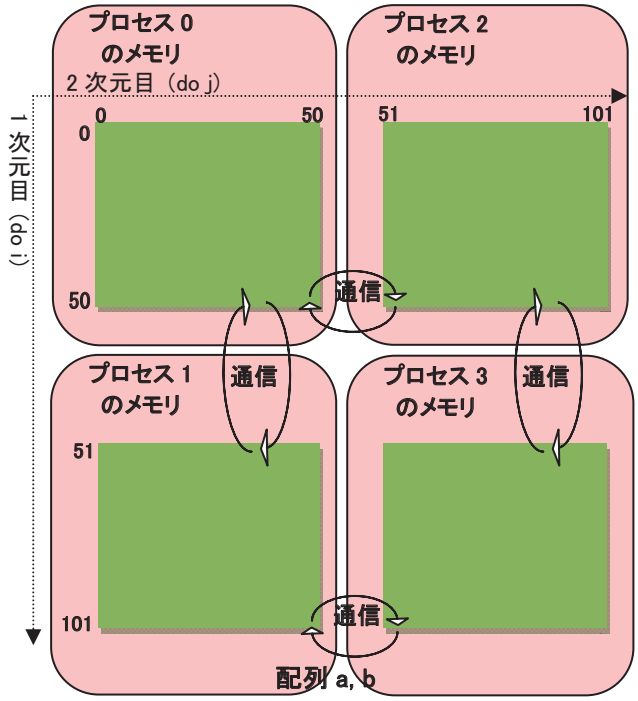


図 3.4　配列 a, b の分割配置(2 次元分割)

3.2 MPI

3.2.1 MPI 概要

MPI (Message Passing Interface) は、Message Passing Interface Forum[2] によって策定された通信ライブラリのオープンな仕様であり、分散並列プログラミングのデファクト標準として広く使用されています。MPI を使用した分散並列化では、図 3.1 や図 3.4 中の通信部分を、MPI 手続の引用によってプログラムすることになります。

分散並列化時に高いスピードアップを達成するには、通信コストの削減が特に重要です。マルチノード

システムの場合、一般に、ノード内のメモリアクセスコストと比較して、ノード間の通信コストは圧倒的に大きいからです。

SX-ACE は、異なるノードのメモリを IXS の命令により直接参照できるハードウェア機能（グローバルメモリ）などを活用した高速な通信環境を提供するため、MPI ライブラリ MPI/SX を用意しています。MPI/SX の 1 対 1 通信は、通信サイズに応じて最適な通信プロトコルを選択し、IXS の性能を最大限引き出しています。また MPI/SX の集団通信は、プロセスのノードへの配置方法や通信サイズに応じて最適な通信アルゴリズムを選択し、各ノードの共有メモリや IXS のバリア同期機構を利用した高速な通信を実現しています。

3.2.2 MPI プログラミングの基本

MPI プログラミングでは、まず MPI 仕様で規定されている各種定数などが定義されたヘッダファイル mpif.h （Fortran の場合） 又は mpi.h （C・C++の場合）をインクルードします（Fortran の場合、ヘッダファイルをインクルードする代わりに、システムモジュール mpi を引用することもできます）。次に、手続 MPI_INIT 又は MPI_INIT_THREAD を引用して初期化を行い、手続 MPI_FINALIZE を引用して MPI の使用を終了します。利用者は、この二つの引用の間で、1 対 1 通信、集団通信、又は単方向通信により通信を行うことができます。

通信を行うプロセスの集合は、コミュニケータと呼ばれる引数を MPI 手続に渡すことにより指定します。プログラム開始時の全プロセスに対応するコミュニケータは、MPI の定数 MPI_COMM_WORLD で指定できます。各プロセスは、ランクと呼ばれるコミュニケータ中で一意な整数値をもちます。ランクの値は、0 以上、プロセス数-1 以下です。手続 MPI_COMM_SIZE 及び MPI_COMM_RANK により、それぞれプロセス数及び自プロセスのランクを取得でき、これらを利用して各プロセスが行う処理をプログラムできます。

MPI プログラムの典型的な例として、図 1.1 の 2 重ループを外側ループで四つの仕事に分割し、分散並列化したプログラムを図 3.5 に示します。MPI プログラミングでは、プログラム全体のデータ・処理ではなく、各プロセス用に分割した後のデータ・処理をプログラムします。そのため、定数 ly の値を 25 に変更して（文番号 100）、配列の宣言（文番号 200）とループ長（文番号 300）を分割後の範囲に縮小しています。この際、通信時に受信したデータを配置する領域が必要なので、図 3.6 のように各プロセス上の配列 a の領域を大きめに割り付けていることに注意してください。このような隣接プロセスとの通信のための領域は袖領域と呼ばれます。並列ループ（文番号 300）実行前に、分割境界の処理に必要な配列 a の要素を、1 対 1 通信手続 MPI_SENDRECV により隣接プロセス間で送受信（文番号 400）し、並列ループ実行中は、隣接プロセス上のデータの代わりに自プロセス上の袖領域を参照しています。両端のプロセスには片側の隣接プロセスが存在しないので、通信相手を設定する IF 構文（文番号 500）において、通信相手を示す変数 men, mep に、該当するプロセスが存在しないことを示す MPI の定数 MPI_PROC_NULL を設定しています。送受信先として定数 MPI_PROC_NULL を指定すると、その通信は実行されません。

```
      include "mpif.h"
100   integer, parameter :: lx=100, ly=25   ! ly は分割後の寸法
200   real(kind=8) a(0:lx+1,0:ly+1),b(0:lx+1,1:ly), k
   ! 初期化
      call mpi_init(ierr)
   ! プロセス数を変数 nproc に取得する
      call mpi_comm_size(mpi_comm_world,nproc,ierr)
   ! 自プロセスのランクを変数 me に取得する。
      call mpi_comm_rank(mpi_comm_world,me,ierr)
                :
   ! 通信相手の設定：両端のプロセスは特別処理を行う。
500   if(me .eq. 0)then
        men = mpi_proc_null  ! 下端のプロセスの下隣は存在しない
      else
        men = me - 1  ! 下隣のプロセスのランク
      endif
```

```
      if(me .eq. nproc-1)then
        mep = mpi_proc_null  ! 上端のプロセスの上隣は存在しない
      else
        mep = me + 1  ! 上隣のプロセスのランク
      endif
  ! 隣接プロセス間の通信：上方向 → 下方向
400   call mpi_sendrecv(a(1,ly),lx,mpi_double_precision,mep,0,                &  ! 上方向
     &     a(1,0),lx,mpi_double_precision,men,0,mpi_comm_world,mpi_status_ignore,ierr)
      call mpi_sendrecv(a(1,1),lx,mpi_double_precision,men,0,                 &  ! 下方向
     &    a(1,ly+1),lx,mpi_double_precision,mep,0,mpi_comm_world,mpi_status_ignore,ierr)
  ! 並列化後のループ
300   do j=1,ly
        do i=1,lx
          b(i,j) = a(i,j) + k * (a(i+1,j)-4.0D0*a(i,j)+a(i-1,j)+a(i,j+1)+a(i,j-1))
        enddo
      enddo
          :
  ! 後始末処理
      call mpi_finalize(ierr)
      end
```

図 3.5　MPI プログラム例

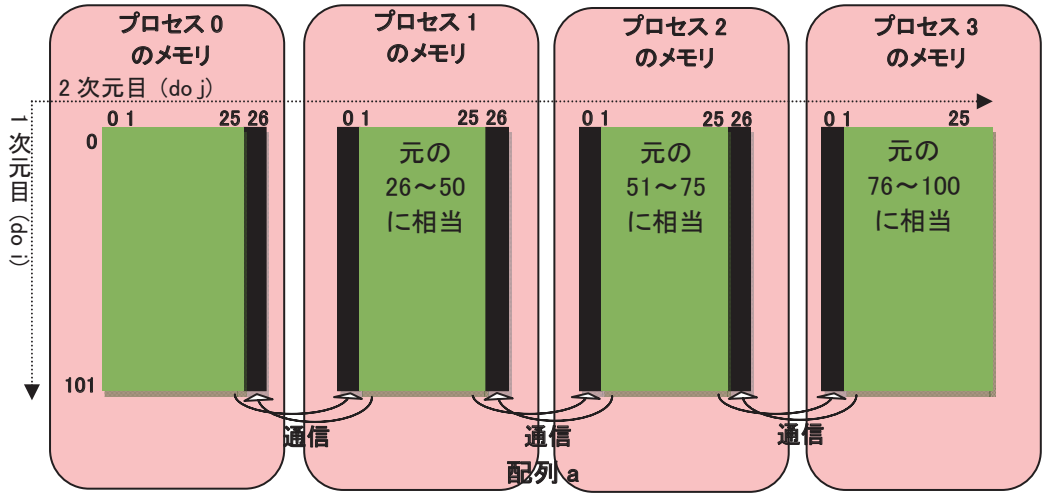


図 3.6　1 次元分割時の配列 a の割付け（黒色は袖領域）

3.3 MPI/SX の活用方法と注意点

3.3.1 MPI プログラムへの標準入力

MPI プログラムに標準入力から入力データを与える場合、図 3.7 のように、実行ファイルに対して直接入力データを与えると、正常に実行できない場合があります。

```
mpirun -np 4 -nn 4 /execdir/a.out < /datadir/inputfile
```

図 3.7　MPI プログラムへの不正な標準入力の例

このような場合、図 3.8 のように、コンパイラの環境変数 F_FF05 に入力ファイル名を設定して事前接続し、さらに MPI の環境変数 MPIEXPORT にその環境変数名 F_FF05 を設定した上で、MPI プログラムを実行してください。なお、MPI プログラム実行時にコンパイラの環境変数を使用するには、このようにコン

パイラの環境変数を設定した上で、環境変数 MPIEXPORT にその環境変数名を設定する必要があります。

```
setenv F_FF05 /datadir/inputfile
setenv MPIEXPORT "F_FF05"

mpirun -np 4 -nn 4 /execdir/a.out
```

図 3.8　MPI プログラムへの標準入力

3.3.2 プロセス毎に別々のファイルを対象にした入出力

環境変数 MPIRANK は、手続 MPI_COMM_RANK に引数 MPI_COMM_WORLD を渡して引用することにより得られるランクと同一の値をもっています。これを利用して、図 3.9 のようなシェルスクリプトを作成し（シェルスクリプトのファイル名を、例えば/execdir/run.sh とします）、図 3.10 のように、mpirun コマンドでそのシェルスクリプトを実行すると、各プロセスのランクと入出力ファイル名とを一対一に対応させ、プロセス毎に別々のファイルを対象にした入出力が可能です。

```
#!/bin/csh
setenv F_FF05 infile.$MPIRANK
setenv F_FF06 outfile.$MPIRANK
setenv MPIEXPORT "F_FF05 F_FF06"
exec /execdir/a.out
```

図 3.9　環境変数 MPIRANK を利用した入出力のためのシェルスクリプト例

```
mpirun -np 4 -nn 4 /execdir/run.sh
```

図 3.10　シェルスクリプトを介した MPI プログラムの実行例

プログラム中で接続したファイルに対して入出力する場合は、手続 MPI_COMM_RANK により取得したランクの値を利用してファイル名を決めると、やはりプロセス毎に別々のファイルを対象にした入出力が可能です

3.3.3 プロセス毎に別々のファイルへの標準出力・標準エラー出力

複数の MPI プロセスが同時に出力を行った場合、それらは入り交じって表示される場合があります。各プロセスの標準出力及び標準エラー出力を別々のファイルに容易に出力できるように、MPI/SX はシェルスクリプト/usr/lib/mpi/mpisep.sh を用意しています。このスクリプトを使用して、図 3.11 のように MPI プログラムを実行すると、環境変数 MPISEPSELECT の値に応じて、次のように各プロセスの標準出力及び標準エラー出力が別々のファイルに出力されます。なお下記の u は通常 0 となり、r は環境変数 MPIRANK の値となります。

- 環境変数 MPISEPSELECT の値が 1 の場合
  各プロセスの標準出力が、ファイル stdout.u:r に出力されます。

- 環境変数 MPISEPSELECT の値が 2 の場合（既定値）
  各プロセスの標準エラー出力が、ファイル stderr.u:r に出力されます。

- 環境変数 MPISEPSELECT の値が 3 の場合
  各プロセスの標準出力及び標準エラー出力が、それぞれファイル stdout.u:r 及び stderr.u:r に出力されます。

- 環境変数 MPISEPSELECT の値が 4 の場合
  各プロセスの標準出力及び標準エラー出力が、共にファイル std.u:r に出力されます。

```
mpirun -np 4 -nn 4 /usr/lib/mpi/mpisep.sh  /execdir/a.out
```

図 3.11　シェルスクリプト /usr/lib/mpi/mpisep.sh を使用した実行例

3.3.4 グローバルメモリを利用した高速化

転送元・転送先のデータを共にグローバルメモリ上に割り付け、MPI の基本データ型を使用することによって、通信を高速化できます。ただし、転送元・転送先データの先頭アドレス及び転送バイト数が、全て 8 の倍数である必要があります。

利用者のデータをグローバルメモリ上に割り付けるには、コンパイラオプション-gmalloc を指定した上で ALLOCATE 文を実行するか（Fortran の場合）、又は手続 MPI_ALLOC_MEM を引用して（C・C++の場合）メモリを割り付けてください。これらの手段で割り付けたメモリ領域の先頭アドレスは、8 の倍数であることが保証されています。

単方向通信（MPI_PUT 又は MPI_GET）において、転送元・転送先のデータが前述の条件を満たす場合、手続 MPI_WIN_FENCE の引数 assert に MPI/SX 独自の最適化情報 MPI_NEC_MODE_GETPUTALIGNED を指定すると、同期処理の高速化が可能です。また手続 MPI_WIN_CREATE によるウィンドウ生成時に、MPI/SX 独自キー"use_gbc"を"true"に設定した info 実体を引数 info に指定すると、同期処理をさらに高速化できます。

3.3.5 MPI プログラムデバッグ機能

MPI/SX は、実行時でなければ見つけにくい MPI プログラム特有のバグを検出する機能を用意しています。コンパイラオプション-mpiverify を指定すると、MPI 手続に対する引数のプロセス間の不整合や、配列引数への領域外アクセスの可能性などの情報が実行時に出力されます。

3.3.6 Fortran からの MPI 使用時の注意点

Fortran プログラム中で引用する非ブロッキングな MPI 手続の実引数として、部分配列、配列式、配列ポインタ、又は形状引継ぎ配列を指定すると、引数がメモリ上連続であることを保証するために、コンパイラが一時配列を割り付けて実引数をコピーする場合があります。このとき、実際の引数としては、その一時配列が MPI 手続に渡されます。しかし、その一時配列は、非ブロッキングな MPI 手続の完了以前に解放されてしまうので、プログラムが正常に実行されない可能性があります。非ブロッキングな MPI 手続の実引数には、メモリ上連続な配列を指定してください。配列ポインタや形状引継ぎ配列のように、翻訳時にはメモリ上連続であることが分からない場合、利用者がメモリ上連続であることを保証できるなら、コンパイラオプション-Wf"-cont"を指定すると、コンパイラによる一時配列の生成を抑制できます。

3.4 ハイブリッド並列化

複数のプロセスだけによるプログラムの分散並列化をフラット並列化と呼びます。SX-ACE では、分散並列化と第 2 章でご説明した共有並列化を併用することもできます。これをハイブリッド並列化と呼びます。

ハイブリッド並列化は、MPI プログラムを自動並列化又は OpenMP により共有並列化するだけで使用可能です。

図 3.5 では、図 1.1 の 2 重ループを外側ループで四つの仕事に分割し、4 プロセスに割り当てフラット並列化しました。これを、それぞれ 2 タスクからなる二つのプロセスに割り当てハイブリッド並列化するには、図 3.5 中の定数 ly の値を 50 に変更した上で、自動並列化のためのコンパイラオプション-P auto 及びタスク数を指定するコンパイラオプション-reserve=2 を指定し共有並列化します。フラット並列化の場合、配列 a を各プロセス上に図 3.6 のように割り付けました。一方、ハイブリッド並列化の場合、配列 a を各プロセス上に図 3.12 のように割り付けることになります。一つのプロセス中の複数のタスクは、そのプロセスの全データ領域を直接参照できるので、タスク間で通信を行う必要はなく、従ってタスク間には袖領域も必要ないことに注意してください。

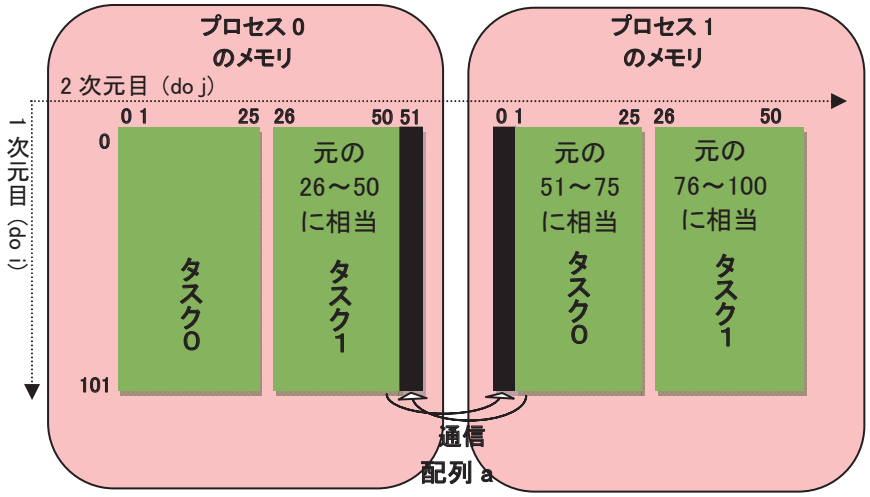


図 3.12　ハイブリッド並列化時の配列 a の割付け（黒色は袖領域）

一つの仕事を複数の小さな仕事に分割して、複数のプロセスに割り当てる場合（フラット並列化）と、各プロセス内の複数のタスクに割り当てる場合（ハイブリッド並列化）とを比較すると、ハイブリッド並列化によりプロセスの総数が削減されます。そのため、プロセス数に依存してコストが増大する通信を含むようなプログラムは、ハイブリッド並列化により経過時間を短縮できる場合があります。その反面、分散並列化と共有並列化の 2 重のオーバヘッドが発生するので、プログラムによってはハイブリッド並列化により経過時間が増加する場合もあります。一方、コンパイラ・MPI の処理系内部のバッファや、全プロセスに重複配置するデータの個数は、プロセス数の減少とともに減少するので、ハイブリッド並列化の方がフラット並列化よりも使用メモリの総量を削減できる場合があります。

3.4.1　ハイブリッド並列化時の注意点

自動並列化の場合、既定値では各プロセスがノード内の全 CPU コアを使用して共有並列実行を行います。そのため、ハイブリッド並列化時に一つのノード上で二つ以上のプロセスを実行すると、CPU コアの個数がタスクの個数より少なくなり、大幅に性能が低下する可能性があります。ハイブリッド並列化時には、各ノードのプロセス数を 1 にするか、共有並列化の並列数を指定する環境変数 F_RSVTASK（Fortran の場合）若しくは C_RSVTASK（C・C++の場合）を指定するか、又はコンパイラオプション-reserve=共有並列数 を指定して、各ノードにおけるプロセス数とタスク数の積が 4 以下となるようにしてください。

MPI/SX のタスクサポートレベルは、MPI_THREAD_SERIALIZED です。従って、全タスクが MPI 手続を引用できますが、手動で共有並列化する場合、同一プロセス中の複数のタスクが同時に MPI 手続を引用しないように利用者がプログラムする必要があります。

3.5 性能解析

3.5.1 MPI プログラム特性情報（MPIPROGINF）

MPI プログラムの実行性能を分析するために、MPI プログラム特性情報（MPIPROGINF）が使用できます。MPIPROGINF は、プログラム実行時に環境変数 MPIPROGINF を、YES，ALL，DETAIL，又は ALL_DETAIL に設定することによって採取できます。図 3.13 に、環境変数 MPIPROGINF を DETAIL に設定して実行した場合の出力例を示します。経過時間が最小のプロセス（Min）、最大のプロセス（Max）、及び平均（Average）の情報が表示されており、プログラム全体の性能を簡便に把握できます。

```
MPI Program Information:
========================
Note: It is measured from MPI_Init till MPI_Finalize.
      [U,R] specifies the Universe and the Process Rank in the Universe.


Global Data of 4 processes  :          Min [U,R]          Max [U,R]      Average
============================
  Real Time (sec)           :       56.829 [0,3]       56.879 [0,0]       56.855
  User Time (sec)           :       15.153 [0,2]       18.308 [0,0]       16.238
  Sys  Time (sec)           :        0.728 [0,3]        0.983 [0,0]        0.850
  Vector Time (sec)         :        3.765 [0,3]       10.689 [0,0]        6.807
  Inst. Count               :    721497042 [0,0]   1091361840 [0,2]    917296494
  V. Inst. Count            :     12671162 [0,3]     48775850 [0,0]     29139941
  V. Element Count          :    537593371 [0,3]   7704138496 [0,0]   4113854075
  V. Load Element Count     :            0 [0,0]            0 [0,0]            0
  FLOP Count                :          737 [0,2]      8400513 [0,0]      2102273
  MOPS                      :      104.355 [0,3]       517.423 [0,1]      296.505
  MFLOPS                    :        0.000 [0,3]        0.459 [0,0]        0.115
  A. V. Length              :       41.177 [0,2]       182.593 [0,1]      106.037
  V. Op. Ratio (%)          :       33.308 [0,2]        91.969 [0,0]       62.558
  Total Memory Size (MB)    :     1344.000 [0,0]     1344.000 [0,1]     1344.000
    Memory Size (MB)        :     1280.000 [0,0]     1280.000 [0,0]     1280.000
    Global Memory Size (MB) :       64.000 [0,0]       64.000 [0,0]       64.000
  MIPS                      :       39.408 [0,0]       72.020 [0,2]       57.512
  I-Cache (sec)             :        0.365 [0,0]        0.906 [0,2]        0.679
  O-Cache (sec)             :        0.539 [0,0]        1.134 [0,3]        0.786
  Bank Conflict Time
    CPU Port Conf. (sec)    :        0.000 [0,0]        0.000 [0,0]        0.000
    Memory Network Conf. (sec):      0.000 [0,0]        0.000 [0,0]        0.000
  ADB Hit Element Ratio (%) :        0.000 [0,0]        0.000 [0,0]        0.000
```

図 3.13　MPIPROGINF

3.5.2 簡易性能解析機能（ftrace 機能・NEC Ftrace Viewer）

コンパイラオプション-ftrace を指定すると、簡易性能解析情報ファイル ftrace.out.u.r（u は通常 0、r は環境変数 MPIRANK の値）が各プロセスから実行時に出力されます。MPI プログラムの場合、コンパイラによる簡易性能解析情報に加えて、通信時間などの情報が表示できます。

SX-ACE が用意する NEC Ftrace Viewer[3] は、並列プログラムの性能を分析する際にも有効です。例として、14 プロセスにより並列実行したある MPI プログラムの負荷バランスを解析してみます。まず、簡易性能解析情報ファイルを読み込んだときに表示される Process Breakdown Chart を見ると、図 3.14 のように手続 iterate 内の実行時間（Exclusive Time）が大きいことが分かります。実行時間が大きな手続 iterate

の負荷バランスを把握するため、左上の「Chart」メニューから Function Statistics Chart を選択します。すると、図 3.15 中の折れ線グラフが示すように、手続 iterate の Exclusive Time の標準偏差が大きく、負荷バランスが悪いことが分かります。状況をより詳細に調査するため、「Chart」メニューから Process Metrics Chart を選択します。次に、右上の「Metrics Selection」メニューから、MPI COMM. TIME（MPI 通信時間）及び MPI IDLE TIME（MPI 待ち時間）を選択します。すると、図 3.16 のように、MPI 通信時間及び MPI 待ち時間を示す二つの折れ線グラフがほぼ重なって見えます。これは、MPI 通信時間のほとんどが MPI 待ち時間であることを意味しています。また、棒グラフで示される手続 iterate の Exclusive Time と、折れ線グラフの MPI 待ち時間とを比較すると、Exclusive Time に占める MPI 待ち時間が、両端のプロセスだけ短いことが分かります。従って、両端のプロセスの MPI 待ち時間以外の実行時間、つまり演算時間が他のプロセスよりも大きく、相対的に演算時間の小さな他のプロセスにおいて、MPI による通信時に待ちが発生していると推定できます。以上のことから、両端のプロセスの演算を他のプロセスに分配すれば、負荷バランスを改善できる可能性があることが分かります。このように、NEC Ftrace Viewer のグラフ機能を使用すると、負荷バランスなど複数の仕事間の関係を容易に把握できます。

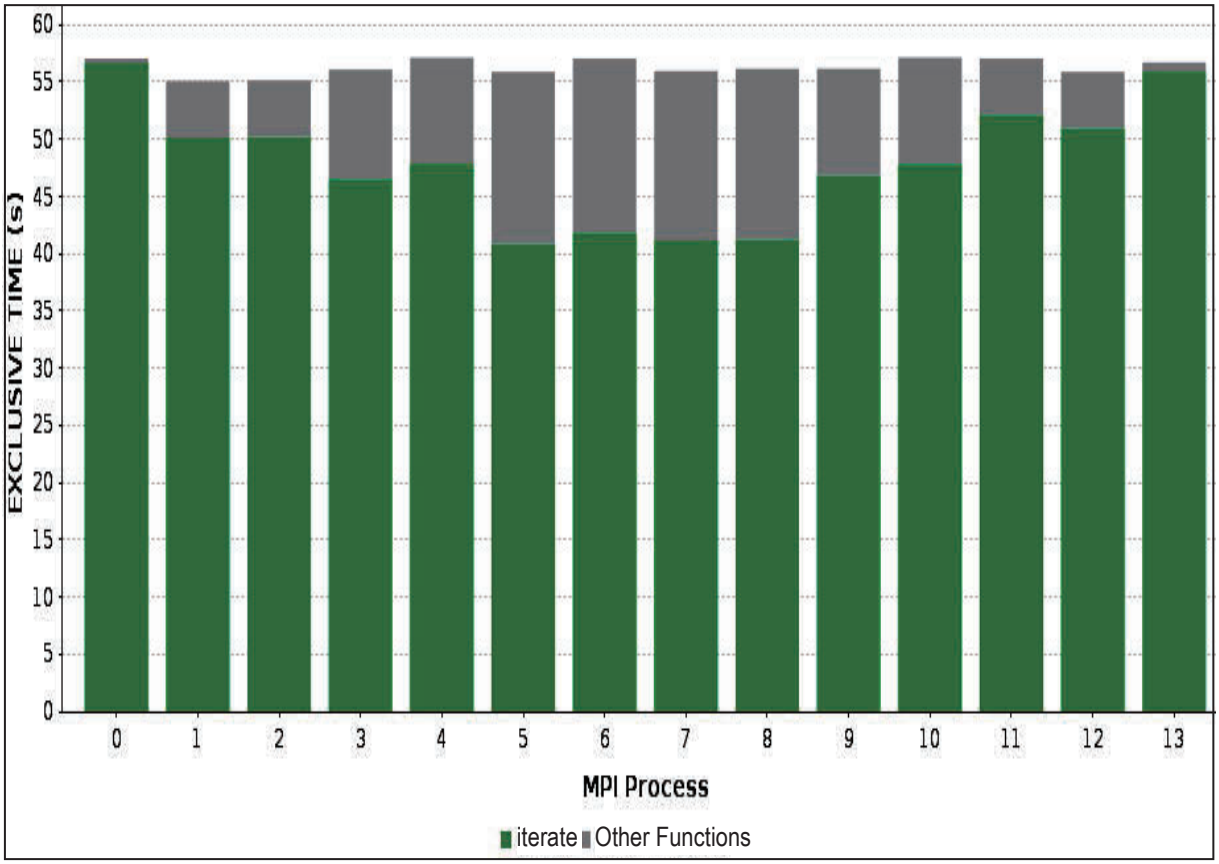


図 3.14　Process Breakdown Chart

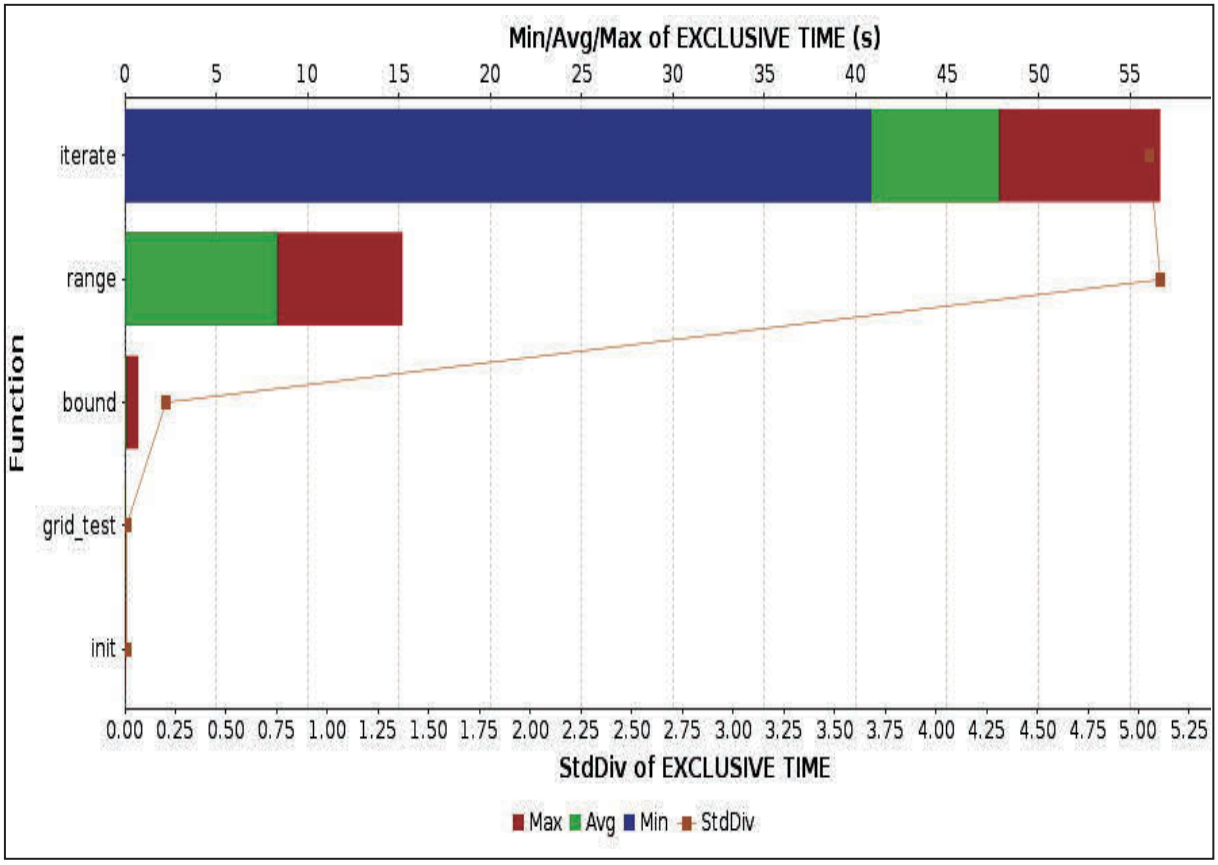


図 3.15　Function Statistics Chart

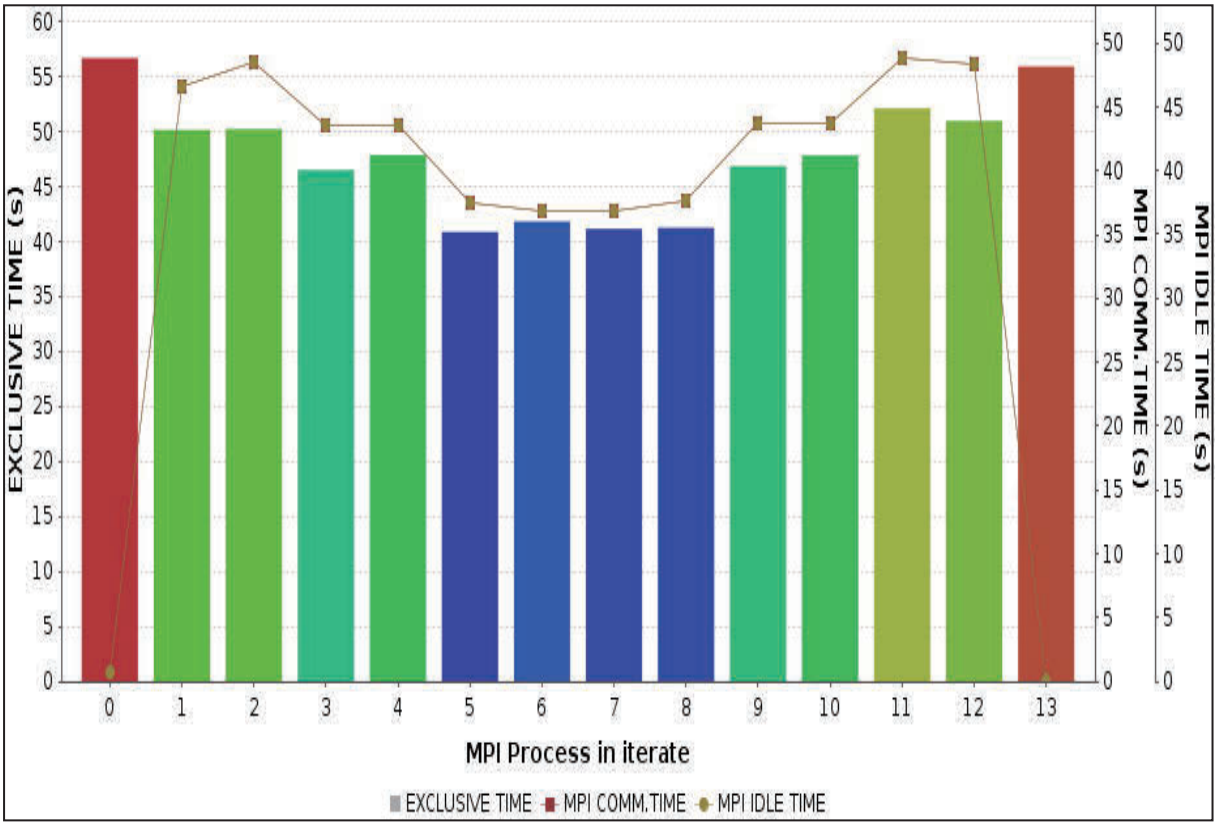


図 3.16　Process Metrics Chart

4. あとがき

本稿では、コンパイラの自動並列化機能及びMPI/SXを使用したプログラム並列化の基本的事項を中心にご紹介しました。SX-ACE が提供する並列処理環境の詳細な使用方法については、「FORTRAN90/SXプログラミングの手引」[4]、「FORTRAN90/SX並列処理機能利用の手引」[5]、「C++/SXプログラミングの手引」[6]、及び「MPI/SX 利用の手引」[7] をご覧ください。SX-ACE をご使用になる上で、本稿がお役に立てば幸いです。

参考文献

[1] The OpenMP Architecture Review Board (http://openmp.org/)
[2] Message Passing Interface Forum (http://www.mpi-forum.org/)
[3] NEC Ftrace Viewer 利用の手引，日本電気株式会社，G1AF33
[4] FORTRAN90/SX プログラミングの手引，日本電気株式会社，G1AF07
[5] FORTRAN90/SX 並列処理機能利用の手引，日本電気株式会社，G1AF08
[6] C++/SX プログラミングの手引，日本電気株式会社，G1AF28
[7] MPI/SX 利用の手引，日本電気株式会社，G1AF09

［大規模科学計算システム］

アプリケーションサービスの紹介

情報部情報基盤課 共同利用支援係

はじめに

本センター大規模科学計算システムでは、分子軌道計算、数式処理、構造解析、データ処理等の各アプリケーションソフトウェアを、利用者の幅広い要望にお応えしてサービスしています。本稿では、並列コンピュータ LX 406Re-2 でサービスを行っているアプリケーションソフトウェアの紹介をします。

表 1. アプリケーションソフトウェアとサービスホスト

アプリケーションソフトウェア		サービスホスト
分子軌道計算ソフトウェア	Gaussian	front.cc.tohoku.ac.jp
反応経路自動探索プログラム	GRRM11	
統合型数値計算ソフトウェア	Mathematica	
汎用構造解析プログラム	Marc/Mentat	
構造解析用汎用プリポストソフトウェア	Patran	
対話型解析ソフトウェア	MATLAB	

アプリケーションソフトウェアの紹介は、以下の URL の本センター大規模科学計算システム Web ページにも掲載しています。

http://www.ss.cc.tohoku.ac.jp/application/index.html

本稿中の内容は 2015 年 3 月現在のものですので、アプリケーションソフトウェアのバージョンアップや利用方法の最新情報については、Web ページを随時ご確認ください。

ご利用の前に

■ リモートログイン

スーパーコンピュータ、並列コンピュータへリモートログインする手順です。SSH（Secure SHell）接続を行います。アプリケーションを利用する際は、並列コンピュータにログインします。GUI アプリケーションを利用する場合は、**GUI アプリケーションを利用する方法**を合わせてご参照ください。

表 2. 計算機システムと日本語環境

システム	ホスト名	OS	日本語環境
並列コンピュータ LX 406Re-2	front.cc.tohoku.ac.jp	Linux	UTF-8

SSH は通信路上のデータを暗号化することで安全性を高めたプログラムです。利用している端末が UNIX, Linux, OS X の場合は SSH クライアントソフトがインストールされています。インストールされていない場合は端末の管理者にご相談ください。

並列コンピュータの OS は Linux です。公開鍵暗号方式による認証のみ利用できます[1]。アカウント希望の場合は、共同利用支援係に利用申請し利用者番号と初期パスワードを発行してもらいます。

[1] パスワード認証方式は 2015 年 4 月 13 日で廃止しました。

並列コンピュータへの初回ログイン時には公開鍵と秘密鍵のペアを作成する必要があります。鍵ペアの作成方法については本誌105ページの「SSH アクセス認証鍵生成サーバの利用方法」をご参照ください。
なお、他人名義の利用者番号でのシステム利用は禁止します。パスワード、秘密鍵、パスフレーズの使い回しは、不正アクセスのリスク(不正ログイン、クライアントのなりすまし、暗号化された通信の暴露、他サーバへの攻撃等)が非常に高く、大変危険です。利用者登録を行うことによる年間維持費等は発生しませんので、利用される方はそれぞれで利用申請をお願いいたします。

【UNIX, Linux からのログイン】

「ターミナル」、「端末」、「terminal」などの SSH クライアントソフトを起動します。コマンドを入力するプロンプトが表示され、コマンドの待ち受け状態になります。

リスト 1. 並列コンピュータへのログイン例

```
localhost$ ssh -i ~/.ssh/id_rsa 利用者番号@front.cc.tohoku.ac.jp
Enter passphrase for key '/home/localname/.ssh/id_rsa':パスフレーズを入力
（初回接続時のメッセージ） : yes を入力
front1 $ （コマンド待ち状態）
```

【OS X からのログイン】

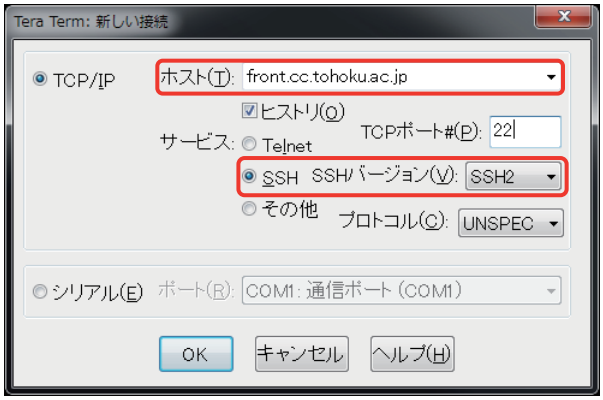
「ターミナル.app」を起動します。接続方法は上記と同じです。

【Windows からのログイン】

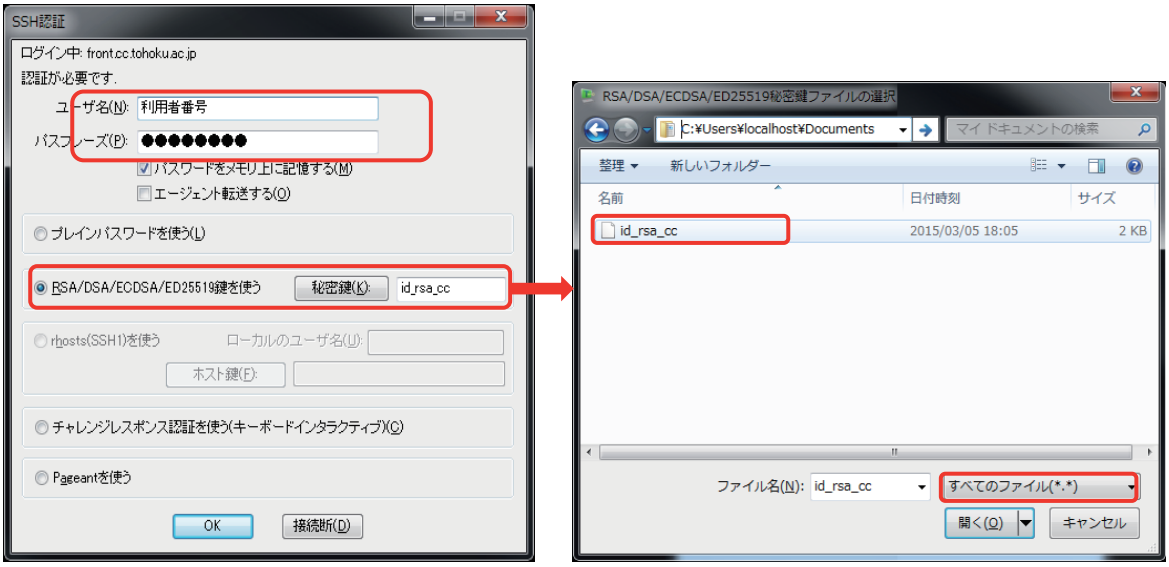
● SSH クライアントソフトのダウンロードとインストール
SSH クライアントソフトの一つである「Tera Term」というフリーソフトをインストールします。以下のページからダウンロードできます。2015 年 3 月現在の最新版は 4.86 です。ダウンロード後インストール作業を行ってください。

Tera Term ダウンロードページ:　　http://sourceforge.jp/projects/ttssh2/

● 並列コンピュータへの接続
「ホスト名」を指定、「サービス」は SSH2 を選択し、[OK]を押下します。



「ユーザ名」に利用者番号、「パスフレーズ」に鍵ペアを作成した際に入力したものを入力、「RSA/DSA 鍵を使う」を選択し、「秘密鍵」に保存した秘密鍵のファイルを指定します。
(秘密鍵ファイルの選択画面では、拡張子「すべてのファイル(*.*)」を選択します)
[OK]を押下すると接続されます。



【シェルの初期設定】

大規模科学計算システムでは、お勧めの初期環境設定を用意しています。これによりパスなどの基本的な設定、また各アプリケーションの環境変数等が自動的に設定されます。これは、利用登録時に個々の ID にあらかじめ行っていますので、通常は作業の必要はありません。

アプリケーションが利用できないという場合には、この設定が変更されていることが考えられます。.cshrc ファイル(csh を利用する場合、センターの規定値) または .login ファイル(sh を利用する場合)に、センターで用意している初期設定ファイル /usr/skel/Cshrc または/usr/skel/Login を読み込む設定となっていることを確認してください。設定を変更した場合は、設定を反映させるためにログインし直してください。

【ファイル転送】

● コマンドラインでのファイル転送

ローカル端末から「scp」、「sftp」コマンドが利用できます。どちらのコマンドも通信経路上は暗号化されていますので安全性の高いファイル転送ができます。利用方法についてはそれぞれのマニュアルをご参照ください。

● アプリケーションを利用したファイル転送

ファイル転送を行う代表的なアプリケーションは Linux では「gftp」、Windows では「WinSCP」、OS X では「Cyberduck」などです。利用方法についてはそれぞれのマニュアルをご参照ください。アプリケーションの設定において、転送プロトコルは SSH2 を選択してください。通信経路上は暗号化されます。

● 入出力端末を利用したファイル転送

センター1F の利用相談室に設置された入出力端末を利用して、USB 接続(USB3.0 対応)の HDD にホームディレクトリのデータをコピーすることができます。センター内ネットワークからのアクセスで、高速なファイルのコピーが可能です。利用方法はセンターまでお問い合わせください。

■ GUI アプリケーションを利用する方法

GUI を用いたアプリケーション（MSC. Mentat, Mathematica, MATLAB）の実行には、ローカルマシンに X Window System 環境の設定が必要です。

【UNIX, Linux からの利用】

標準で X Window System がインストールされています。ローカル端末から以下の様にログインしてください。X Forwarding によりローカル画面にアプリケーション画面が表示されます。

リスト 2. Matlab を起動する場合

```
localhost$ ssh -i ~/.ssh/id_rsa 利用者番号@front.cc.tohoku.ac.jp
Enter passphrase for key '/home/localname/.ssh/id_rsa':パスフレーズを入力
（初回接続時のメッセージ） : yes を入力
front1 $ matlab
```

※1　大文字の“X”です。

【Windows からの利用】

● 商用のアプリケーションを利用する場合
Windows 用 X サーバは、X サーバソフトとしていくつかのメーカから販売されています。

・ASTEC-X（アステック・エックス）
・Exceed（Open Text Exceed　オープンテキスト・エクシード）

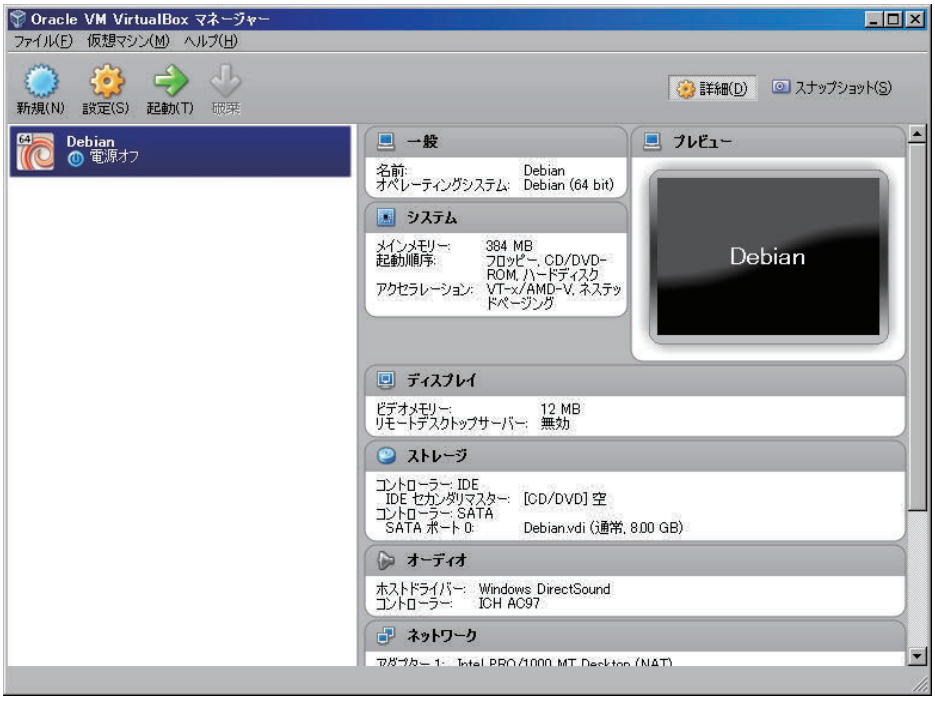
それぞれの利用方法について詳しくは各社の HP をご参照ください。どちらのソフトも無料評価版があります。

● Windows に仮想的な Linux をインストールする場合
Windows に「Oracle VM VirtualBox」（以下「VirtualBox」）という仮想化ソフトウェアをインストールし、その環境に Linux をインストールします。

「VirtualBox」は以下のページからダウンロードできます。「VirtualBox platform packages」（現在使用している OS に合ったもの）と「VirtualBox Extension Pack」の両方をダウンロードし、インストールを行ってください。インストール方法の詳細はマニュアルをご参照ください。2015 年 3 月現在の最新版は 4.3.24 です。

VirtualBox ダウンロード:　　https://www.virtualbox.org/wiki/Downloads

VirtualBox 4.3.24 の起動画面

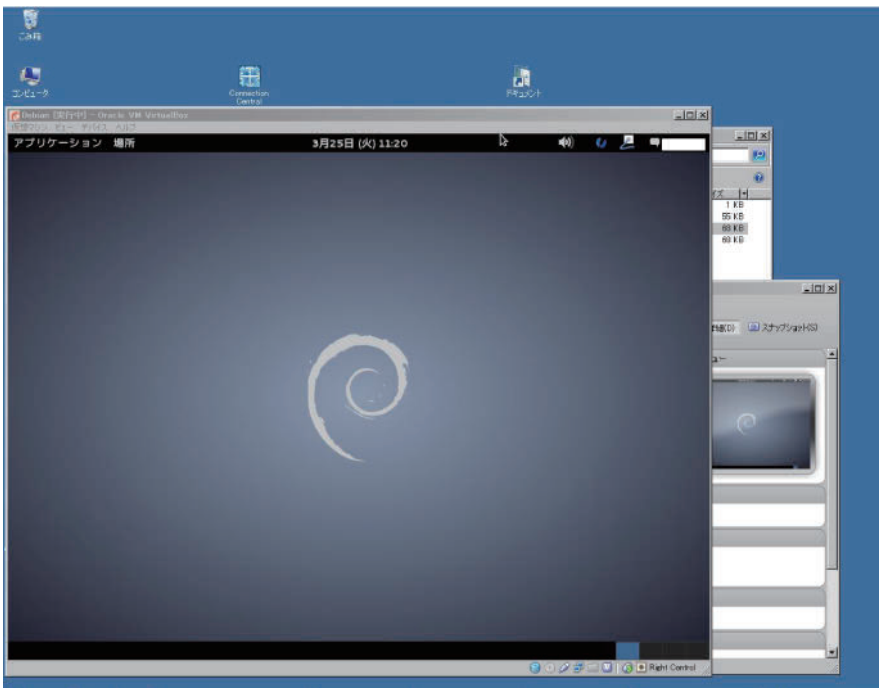


Linux のディストリビューション、バージョンによっては GUI アプリケーションが正しく表示されない場合があります。センターで動作確認を行っているのは、Debian 7.4.0 です。以下のページからダウンロードし、Virtual Box の仮想環境にインストールしてください。X Window System の利用だけならインストール CD（680MB）のインストールで可能です。インストール方法の詳細は各マニュアルをご参照ください。

Debian ダウンロード: http://www.debian.org/CD/

Linux をインストール後 GNOME classic モードで起動し、Linux の端末から【UNIX, Linux からの利用】と同様に利用できます。

Windows 上で動作する仮想Linux（Debian 7.4.0）



【OS X からの利用】

OS X では標準で X Window System 環境の「X11.app」がインストールされていますので、OS X の端末から【UNIX, Linux からの利用】と同様に利用可能ですが、GUI アプリケーションによっては表示の不具合がある場合があります。その場合は、**Windows に仮想的な Linux をインストールする場合** と同様の方法で、Linux をインストールしてご利用ください。

アプリケーションソフトウェア

非経験的分子軌道計算プログラム Gaussian09

Gaussian は、Carnegie-Mellon 大学の Pople を中心として開発された分子軌道計算プログラムパッケージです。広範囲にわたる非経験的モデルおよび半経験的モデルをサポートしています。

本センターの Gaussian には、以下のような特長があります。

- 最大 24 並列までの並列処理が行え、実行時間の短縮が可能です。
- スクラッチファイル(テンポラリファイル)を高速な SSD ディスクに置くことにより、ファイル入出力時間が短縮されます。

■ サービスホスト・バージョン

front.cc.tohoku.ac.jp・Gaussian09 D.01

■ 利用方法

以下は Gaussian 利用方法の概要です。

【実行コマンド】

Gaussian のインプットファイルは、拡張子を .com とします。(例： e2-01.com)

インプットファイルを Windows のエディタで作成した場合、拡張子.com のファイルは Windows では実行ファイルと認識されるため、誤ってダブルクリックなどでインプットファイルを実行しないようご注意ください。また、ファイル転送ソフトで front に転送する際にはアスキーモードを指定し、転送してください。

front.cc.tohoku.ac.jp にログイン後、subg09 コマンドにキュー名と入力プログラム名を指定することにより、バッチリクエストとして実行されます。リクエストはアプリケーション用の利用形態（経過時間無制限、最大並列数 24、最大メモリ 128GB）に投入します。

リスト 3. e2-01.com を解析するコマンド例

(subg09 コマンドに入力ファイルを指定する際は拡張子 .com を省きます)

```
[front1 ~]$ subg09 -q lx -b a e2-01
```

【12 および 24 並列実行の指定】

本センターでサービスしている Gaussian では、12 および 24 並列での並列処理が可能です。大きな分子の解析にぜひご活用ください。

12 または 24 並列で実行するには、ルートセクションに Link 0 コマンドの%NProc=並列数を追加します。手入力の場合は、テキストエディタで先頭行に追加、GaussView 等ではインプットファイル作成画面の Link 0 section の項に追加してください。

【使用メモリ量の指定】

実行して「メモリ量が足りない」というエラーになった場合は、Link 0 コマンド %Mem= で使用メモリ量を増やしてください。

リスト 4. 24 並列、メモリ 16GB の設定をしたインプットファイル e2-01.com を実行する例

```
[front1 ~]$ cat e2-01.com ← インプットファイルの内容を表示

 %NProc=24     ← 並列数
 %Mem=16Gb     ← メモリ量
 # RHF/6-31G(d) Pop=Full Test

 Formaldehyde Single Point

 0 1
 C   0.   0.   0.
 O   0.  1.22  0.
 H  .94  -.54  0.
 H -.94  -.54  0.

[front1 ~]$ subg09 -q lx -b a e2-01
```

【実行結果の確認】

計算が終了すると、インプットファイル名に拡張子.log がつけられた結果ファイル (例: e2-01.log)が作成されます。計算結果をはじめ、CPU 時間などの計算機使用量に関する情報もここに含まれます。

正常終了ならば、このファイルの末尾に「Normal termination of Gaussian 09.」というメッセージが出力されます。ファイルの末尾を表示する tail コマンドで確認できます。

リスト 5. 実行結果の確認

```
[front1 ~]$ tail e2-01.log
  :
 Job cpu time:  0 days  0 hours  0 minutes 30.7 seconds.
 File lengths (MBytes):  RWF=   11 Int=    0 D2E=    0 Chk=    8 Scr=    1
 Normal termination of Gaussian 09 at Mon Nov 1 12:00:00 2006.
```

・ 結果ファイルの詳細な見方は、マニュアル等をご参照ください。

【チェックポイントファイル】
チェックポイントファイルは、デフォルトで作成される結果ファイル（.log ファイル）より詳細な結果が出力され、計算のやり直しや結果を画像表示するためなどに使用されます。チェックポイントファイルを出力するには、ルートセクションに Link 0 コマンドの %Chk=*チェックポイントファイル名* を追加します。

■ マニュアル

本センター本館 1 階 利用相談室に以下の資料を備えてあります。

- 電子構造論による化学の探求 第二版,ガウシアン社,1998
- Gaussian 09 User's Reference
- Gaussian 09 IOps Reference
- Gaussian 09 Online Manual, http://www.gaussian.com/
- Gaussian プログラムによる量子化学計算マニュアル ： 堀憲次, 丸善出版
- すぐできる量子化学計算ビギナーズマニュアル ： 武次鉄也, 講談社
- すぐできる分子シミュレーションビギナーズマニュアル ： 長岡正隆, 講談社
- Gaussian プログラムで学ぶ情報化学・計算化学実験 ： 堀憲次, 丸善出版

Gaussian プリポストシステム GaussView

GaussView は、分子軌道計算プログラム Gaussian のプリポストシステムです。Windows 8/7/Vista, Linux 搭載のパソコンなどで動作し、入力データの作成、計算結果の可視化を 3 次元的に行うことができます。

■ バージョン

5.0.9

■ お申し込み

利用ご希望の方に、GaussView の CD-ROM を貸し出しいたします。

利用条件
・東北大学内の方

CD-ROM は、お手数ですが Gaussian 利用申請書をホームページよりダウンロードしてご記入の上、当センターまで直接お越しください。

■ 利用方法

インストール方法、データ作成方法などについては同梱マニュアルまたは以下のHPをご参照ください。

ヒューリンクス Gauss View 5：　http://www.hulinks.co.jp/software/gaussview/

並列コンピュータ front.cc.tohoku.ac.jp の Gaussian で解析を実行する手順

1. 入力データ作成後、Gaussian のインプットファイル「.com」としてエクスポートします。
2. インプットファイルを front.cc.tohoku.ac.jp に転送します。
3. front.cc.tohoku.ac.jp にログインします。
4. subg09 コマンドにより解析を実行します。
5. 結果ファイルを転送し GaussView で表示します。

チェックポイントファイル(.chk)は、Gaussian のユーティリティコマンド formchk により書式付(.fchk)に変換後転送してください。

反応経路自動探索プログラム GRRM11

GRRM は、2002 年に東北大学（教授：大野公一、修士1年：前田理、当時）で制作が開始され、その後開発が進められて、2011 年には GRRM11（開発者： 前田理、長田有人、諸熊奎治、大野公一）として広く利用されるようになりました。GRRM には、以下のような特長があります。

- g09 や g03 プログラムなどを用いた非経験的量子化学計算に基づいて、各化学式で表される構造や反応経路を自動的に探索します。
- 平衡構造から出発して、その周囲に存在する反応経路を、ポテンシャルの非調和下方歪みを検出して、系統的に調べ上げる超球面探索アルゴリズムが搭載されており、従来不可能であった5原子以上の反応経路自動探索を行うことができます。

■ サービスホスト・バージョン

front.cc.tohoku.ac.jp・11.01

■ 利用方法

利用方法はセンターのホームページ（http://www.ss.cc.tohoku.ac.jp/application/grrm11.html）をご覧ください。

■ GRRM プログラムの詳細

GRRM の詳細については、NPO 法人 量子化学探索研究所（http://iqce.jp/）、化学反応経路自動探索の Web ページ(http://grrm.chem.tohoku.ac.jp/GRRM/)を参照してください。また、GRRM プログラムは現在さら

に開発が進められています。利用法の詳細や新しい情報を得るには、開発者と連絡をとることをお勧めします。(連絡先アドレス:ohnok@m.tohoku.ac.jp)

■ GRRM プログラムの文献と研究成果発表時の引用義務

GRRM11 を用いて得た成果を公表するときは、次のような形式で、著者名, プログラム名, version 名(GRRM 出力の log ファイル参照)を引用文献として記載してください。

Satoshi Maeda, Yuto Osada, Keiji Morokuma, and Koichi Ohno, GRRM11, Version 11.01, 2011.

また、GRRM プログラムに搭載されたオプションの詳細については、それぞれ下記の文献を参照してください。これらのオプションを利用して得た研究成果を公表する際には、次に示す GRRM に関する3つの基本文献(1)-(3)および、下に示された各オプションに対応する文献を引用しなければなりません。

● GRRM:
(1) K. Ohno, S. Maeda, A Scaled Hypersphere Search Method for the Topography of Reaction Pathways on the Potential Energy Surface., Chem. Phys. Lett., 2004, 384, 277-282.; (2) S. Maeda, K. Ohno, Global Mapping of Equilibrium and Transition Structures on Potential Energy Surfaces by the Scaled Hypersphere Search Method: Applications to Ab Initio Surfaces of Formaldehyde and Propyne Molecules., J. Phys. Chem. A, 2005, 109, 5742-5753.; (3) K. Ohno, S. Maeda, Global Reaction Route Mapping on Potential Energy Surfaces of Formaldehyde, Formic Acid, and their Metal Substituted Analogues., J. Phys. Chem. A, 2006, 110, 8933-8941.

● 2PSHS:
S. Maeda, K. Ohno, A New Approach for Finding a Transition State Connecting a Reactant and a Product without Initial Guess: Applications of the Scaled Hypersphere Search Method to Isomerization Reactions of HCN, (H2O)2, and Alanine Dipeptide., Chem. Phys. Lett., 2005, 404, 95-99.

● SCW:
S. Maeda, K. Ohno, Conversion Pathways between a Fullerene and a Ring among C20 Clusters by a Sphere Contracting Walk Method: Remarkable Difference in Local Potential Energy Landscapes around the Fullerene and the Ring., J. Chem. Phys., 2006, 124, 174306/1-7.

● LADD, NLowest, NRUN:
S. Maeda, K. Ohno, Structures of Water Octamers (H2O)8: Exploration on Ab Initio Potential Energy Surfaces by the Scaled Hypersphere Search Method., J. Phys. Chem. A, 2007, 111, 4527-4534.

● Frozen Atom:
S. Maeda, K. Ohno, Lowest Transition State for the Chirality-Determining Step in Ru{(R)-BINAP}-Catalyzed Asymmetric Hydrogenation of Methyl-3-Oxobutanoate., J. Am. Chem. Soc., 2008, 130, 17228-17229.

● External Atom:
S. Maeda, K. Ohno, K. Morokuma, An Automated and Systematic Transition Structure Explorer in Large Flexible Molecular Systems Based on Combined Global Reaction Route Mapping and Microiteration Methods., J. Chem. Theory Comput., 2009, 5, 2734-2743.

● OptX:
S. Maeda, K. Ohno, K. Morokuma, Updated Branching Plane for Finding Conical Intersections without Coupling Derivative Vectors., J. Chem. Theory Comput., 2010, 6, 1538-1545.

● ModelF:
S. Maeda, K. Ohno, K. Morokuma, Automated Global Mapping of Minimum Energy Points on Seams of Crossing by the Anharmonic Downward Distortion Following Method: A Case Study on H2CO., J. Phys.

Chem. A, 2009, 113, 1704-1710.; S. Maeda, K. Ohno, K. Morokuma, Exploring Multiple Potential Energy Surfaces: Photochemistry of Small Carbonyl Compounds, Adv. Phys. Chem. 2012, 2012, 268124.

■ マニュアル

PDF 形式のマニュアルがセンターのホームページから参照できます。

- GRRM プログラム利用ガイド
- GRRM の実行方法（東北大学サイバーサイエンスセンター編）

本センター本館 1 階 利用相談室に以下の資料を備えてあります。

- GRRM11 User Manual（英語版）

汎用構造解析プログラム MSC.Marc / MSC.Marc Mentat

MSC.Marc は有限要素法による非線形汎用構造解析プログラムです。世界中で広く利用され最も評価を受けているプログラムの一つで、その扱える解析は以下の通り非常に広範囲にわたっています。
非線形／大変形／接触／弾塑性／剛塑性／破壊／熱伝導／動的非線形／境界非線形流体と固体の連成／電気伝導と熱伝導の連成／熱と応力の連成
MSC.Marc Mentat は、汎用構造解析プログラム Marc の会話型プリ／ポストプロセッサとして、有限要素モデルの作成および解析結果の表示が行えます。

■ サービスホスト・バージョン

front.cc.tohoku.ac.jp・MSC.Marc /Mentat　2013

■ 利用方法

Marc のプリポストプロセッサとして、Mentat の他に MSC.Patran も提供しています。

【run_marc コマンドでの解析実行】

● 実行コマンド
Marc の入力ファイルは、拡張子を .dat とします。(例： job-name.dat)
front.cc.tohoku.ac.jp にログイン後、 run_marc コマンドに入力ファイル名を指定し実行することにより、バッチリクエストとして解析が行われます。ジョブクラスの指定は必要ありません。自動的にアプリケーション専用の利用形態（経過時間無制限、最大メモリ 128GB）に投入されます。

リスト 6. job-name.dat を解析するコマンド例

(run_marc コマンドに入力ファイルを指定する際は拡張子 .dat を省きます)

```
[front1 ~]$ run_marc -jid job-name -v n
```

表 3. run_marc の入力オプション

オプション	説明
-jid (-j) *job-name* (必須)	入力ファイル名 *job-name*.dat を指定
-cpu *秒数*	cpu 時間の制限
-ver (-v) yes(デフォルト) no	バッチリクエスト投入前に確認する。 バッチリクエストをただちに投入する。
-user (-u) *user_name*	ユーザサブルーチン *user_name*.f を指定

- その他のオプションは、「マニュアル C 編 プログラム入力 付録 B 表 B-2」をご参照ください。

● 解析結果

バッチリクエストが終了すると、主に以下のようなファイルが作成されます。

job-name.out	(解析結果)
job-name.log	(解析ログ)
job-name.t16	(ポストファイル)
job-name.sts	(ステータスレポートファイル)
job-name.batch_err_log	(エラーログ)

解析時の指定によって、この他にもファイルが作成されます。それらのファイルの概要は、「マニュアル C 編プログラム入力 付録 B 表 B-1」をご参照ください。

● 終了番号 (exit number)

解析結果ファイル(job-name.out)の末尾にある marc exit number により、正常に終了したか、エラー終了か、またエラー終了の場合はその原因がわかります。

リスト 7. 終了番号を確認する

(tail コマンドで job-name.out の末尾を表示)

```
[front1 ~]$ tail job-name.out

************************************************************

MSC.Marc Exit number 3004

check marc exit passed
[front1 ~]$
```

表 4. 終了番号

終了番号	説明
3004	正常終了
13	入力データにデータエラーが検出された。
2004	剛体変位が発生している、または全体剛性マトリクスが非正定マトリクスになっている。
3002	指定したリサイクル数内で収束しない。

・　この他の番号については、「マニュアル C 編 プログラム入力 付録 A」をご参照ください。

【プリポストプロセッサ Mentat からの解析実行】

● Mentat の起動

Mentat の起動には、並列コンピュータに接続する際に X forwarding の設定を行う必要があります。Mentat2013 では新 GUI を採用しています。従来の Classic GUI は mentat.classic というコマンドでご利用できます。

リスト 8. mentat の起動方法

```
yourhost$ ssh -X front.cc.tohoku.ac.jp -l 利用者番号
   :
 [front1 ~]$ mentat                （新 GUI 版）

 [front1 ~]$ mentat.classic        （Classic 版）
```

● 解析実行 （新 GUI 版）

Mentat 上でモデルを作成し、解析のための設定を行った後、

タブメニュー [解析ジョブ] -> [新規] -> 解析タイプを選択 -> [実行] -> [実行(1)]

という操作をすることで、バッチリクエストとして解析を実行します。

ツールバーメニュー [ファイル(F)] -> 書き出し -> [Marc 入力...]

とすることで、run_marc コマンド用入力ファイル(.dat ファイル)を作成することができます。

● 解析実行 （Classic 版）

Mentat 上でモデルを作成し、解析のための設定を行った後、

メインメニュー [JOBS] -> [RUN] -> [submit1]

という操作をすることで、バッチリクエストとして解析を実行します。

スタティックメニュー [FILES] -> MARC INPUT FILE [WRITE]

とすることで、run_marc コマンド用入力ファイル(.dat ファイル)を作成することができます。

■ サンプルプログラム

【Marc】

マニュアル E 編に掲載されている例題が、並列コンピュータ front.cc.tohoku.ac.jp の /usr/ap/MSC2013/marc2013/demo/にあります。コピーしてご利用ください

【Mentat】

マニュアル「ユーザガイド」に掲載されている例題ファイルが、並列コンピュータ front.cc.tohoku.ac.jp の /usr/ap/MSC2013/mentat2013/examples/marc_ug/にあります。コピーしてご利用ください。

■ マニュアル

PDF 形式のマニュアルを提供しています。

各マニュアルは、並列コンピュータ（front.cc.tohoku.ac.jp）の以下のディレクトリにあります。

/usr/ap/MSC2013/mentat2013/doc/

release_guide.pdf	:	Release Guide（2013 英語版）
release_guide.pdf	:	What's new（2013 英語版）
mt_help_ref.pdf	:	MSC.Marc Mentat Help Reference（2013 英語版）

英文マニュアル　/usr/ap/MSC2013/mentat2013/doc/(vola～vole)

vola.pdf	:	Volume A：Theory and User Information
volb.pdf	:	Volume B：Element Library
volc.pdf	:	Volume C：Program Input
vold.pdf	:	Volume D：User Subroutines and Special Routines
vole.pdf	:	Volume E：Demonstration Problems

和文マニュアル (MSC.Marc2003 版)　/usr/ap/MSC2013/mentat2013/doc/japanese/

vola.pdf	:	A 編 理論およびユーザー情報
volb.pdf	:	B 編 要素ライブラリ
volc.pdf	:	C 編 プログラム入力
vold.pdf	:	D 編 ユーザサブルーチンおよび特別ルーチン
vole.pdf	:	E 編 例題集
new_features.pdf	:	新機能ガイド
marc_ug.pdf	:	ユーザガイド
mt_help_ref.pdf	:	Mentat 2003 ヘルプリファレンス
xsec_adden.pdf	:	ドキュメント補足資料

有限要素法プログラム汎用プリポストソフトウェア MSC.Patran

MSC.Patran は、有限要素法構造解析プログラム MSC.Nastran 用として開発されたプリポストソフトウェアです。本センターでは Marc の利用をサポートするためにサービスしています。

MSC.Patran は多くの CAD に対応するダイレクトインターフェースを介して、正確で迅速な CAD 形状のインポートが可能です。さらに優れた特長として、高水準のメッシュ作成機能や可視化機能に加え、Marc との親和性が高いことが挙げられます。

■ バージョン

MSC.Patran2014 Windows 版, Linux 版

■ お申し込み

利用条件（以下の条件をすべて満たしている方）

・大規模科学計算システムの利用者番号を持っている方
・本センターでサービスしている Marc のプリポストとして利用する方

利用ご希望の方は、共同利用支援係までお問い合わせください。

数式処理プログラム Mathematica

Mathematica は Stephen Wolfram によって作られた、プログラミング言語を備えた数式処理システムです。Mathematica の機能は、数値計算、記号計算、グラフィックスという 3 つに大別でき、この 3 つが一体となって使いやすいインタフェースを提供しています。

■ サービスホスト・バージョン

front.cc.tohoku.ac.jp・version 9.0.1

■ 利用方法

【Mathematica の起動】

● GUI 版

GUI 版の Mathematica の起動には、並列コンピュータに接続する際に X forwarding の設定を行う必要があります。

リスト 9. GUI 版の起動方法

```
yourhost$ ssh -X front.cc.tohoku.ac.jp -l 利用者番号
   :
[front1 ~]$ mathematica
```

リスト 10. テキスト版の起動方法

```
yourhost$ ssh  front.cc.tohoku.ac.jp -l 利用者番号
    :
[front1 ~]$ math
```

・　Mathematica の基本的な使い方は、マニュアル・参考資料 や、Web などをご参照ください。

■ マニュアル・参考資料

参考資料

本センター本館1階 利用相談室に、以下の資料を備えてあります。

- スティーブンウルフラム Mathematica ブック (日本語版)：トッパン
- Mathematica 方法と応用：J.W. グレイ, サイエンティスト社
- Mathematica プログラミング技法：R. メーダー, トッパン
- 入門 Mathematica ： 日本 Mathematica ユーザー会, 東京電機大学出版局
- はやわかり Mathematica ： 榊原進, 共立出版
- もっと Mathematica で数学を ： 吉田孝之, 培風館

科学技術計算言語 MATLAB

MATLAB は高機能な数値計算機能と多彩な可視化機能を備えた技術計算ソフトウェアです。科学的、工学的分野の様々な数値計算(特に行列演算)、データ解析、シミュレーション、およびビジュアライゼーションのための統合環境を提供しています。

■ サービスホスト・バージョン

front.cc.tohoku.ac.jp ・ R2014a (8.3.0)

■ Toolbox

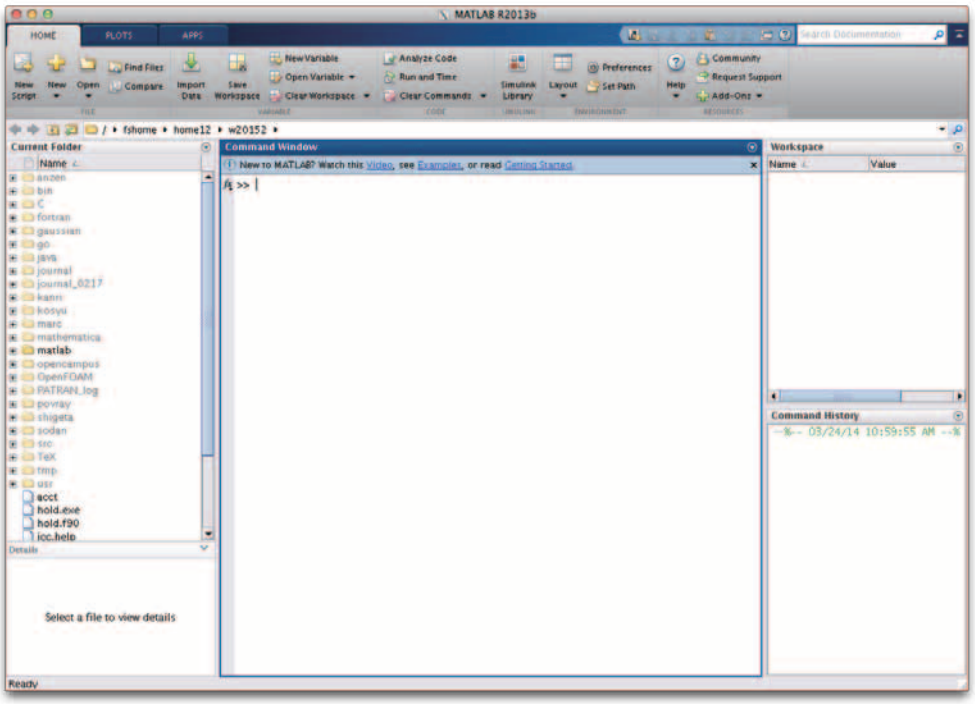
センターで導入している Toolbox です。
MATLAB
Simulink
Curve Fitting Toolbox
Communications System Toolbox
MATLAB Compiler
Control System Toolbox
DSP System Toolbox
Fuzzy Logic Toolbox
System Identification Toolbox
Image Processing Toolbox
MATLAB Corder
Model Predictive Control Toolbox
Neural Network Toolbox
Optimization Toolbox
Partial Differential Eauation Toolbox
Fixed-Point Toolbox
Robust Control Toolbox
Simulink Corder
Simulink Control Design
Signal Processing Toolbox
Symbolic Math Toolbox
Simulink Design Optimization
Statistics Toolbox
Simulink Verification and Validation
Wavelet Toolbox

■ 利用方法

【MATLAB の起動】
● GUI 版
GUI 版 MATLAB の起動には、並列コンピュータに接続する際に X forwarding の設定を行う必要があります。

リスト 11. GUI 版 MATLAB の起動

```
yourhost$ ssh -X front.cc.tohoku.ac.jp -l 利用者番号
   :
 [front1 ~]$ matlab
```



● テキスト版
GUI を使用せず、コマンドライン上で起動することもできます。

リスト 12. テキスト版 MATLAB の起動

```
(yourhost)$ ssh front.cc.tohoku.ac.jp -l 利用者番号
   :
 [front1 ~]$ matlab -nojvm -nosplash -nodesktop -nodisplay

                    < M A T L A B (R) >
          Copyright 1984-2014 The MathWorks, Inc.
            R2014a (8.3.0.532) 64-bit (glnxa64)
                      February 11, 2014

  To get started, type one of these: helpwin, helpdesk, or demo.
  For product information, visit www.mathworks.com.
>>
```

● バッチ処理

MATLAB の組み込み並列処理機能を使用し、24 並列までの処理が可能です。最大メモリも 128GB まで利用可能です。大規模な計算にご利用ください。ただし、バッチ処理ではグラフ描画など画面出力のあるプログラムや、対話的な処理は行えません。

function として作成した test を実行するためには以下の様なバッチリクエスト用シェルスクリプトファイルを作成します。リクエストはアプリケーション専用の利用形態に投入します。

リスト 13. バッチリクエストファイル

```
[front1 ~] cat job-m     ←バッチリクエストファイルの中身を表示

#PBS -q lx -b a  ←アプリケーション専用の利用形態を指定
cd $PBS_O_WORKDIR
matlab -nojvm -nosplash -nodesktop -nodisplay -r test
```

以下のコマンドでリクエストを投入します。

リスト 14. リクエストの投入方法

```
[front1 ~]$ qsub job-m
Request 1234.job submitted to queue: ap.
```

MATLAB の基本的な使い方は、マニュアル・参考資料などをご参照ください。

■ サンプルプログラム

MATLAB には豊富なデモがありますので、ご利用ください。MATLAB 上で、demo コマンドを実行すると、デモ画面が開きます。

■ マニュアル・参考資料

【マニュアル】

日本語オンラインマニュアルが公開されています。以下のページをご参照ください。

http://www.mathworks.co.jp/help/ja_JP/techdoc/index.html

【参考資料】

本センター本館1階 利用相談室に、以下の資料を備えてあります。

MATLAB による制御理論の基礎：野波健蔵, 東京電機大学出版局
MATLAB による制御のためのシステム同定：足立修一, 東京電機大学出版局
だれでもわかる MATLAB ： 池原雅章, 培風館
はやわかり MATLAB 第 2 版：芦野隆一, 共立出版
最新 MATLAB ハンドブック第 3 版：小林一行, 秀和システム

MATLAB グラフィックス集：小国 力, 朝倉書店
MATLAB と利用の実際：小国 力, サイエンス社
MATLAB の総合応用 ： 高谷邦夫, 森北出版
最新使える！MATLAB ： 青山貴伸, 講談社
使える！MATLAB/Simulink プログラミング ： 青山貴伸, 講談社
MATLAB による画像＆映像信号処理 ： 村松正吾, CQ 出版

Matlab によるグラフ描画 ： 西村竜一 (広報誌 SENAC Vol.37 No.1 (2004-1))
高機能数値計算・可視化機能ソフト MATLAB の基本的な使い方 ： 陳国曜 他
(広報誌 SENAC Vol.46 No.3 (2013-7))

［お知らせ］

SSH アクセス認証鍵生成サーバの利用方法

共同利用支援係　共同研究支援係

1. はじめに

2015 年 4 月、大規模科学計算システムのセキュリティ強化のため、パスワード認証によるログインを廃止※1 し、公開鍵暗号方式に統一※2 します。これに伴い、SSH アクセス認証鍵生成サーバの運用を開始します。SSH アクセス認証鍵生成サーバ（以下、鍵サーバ）はセンターに SSH アクセスするために必要な公開鍵と秘密鍵のペアを生成し、ユーザのホームディレクトリに公開鍵を自動登録するサーバです。本稿では、その利用方法についてご紹介します。

表 1　各ホストのログイン認証方式

ログインホスト名	認証方式	利用システム
front.cc.tohoku.ac.jp	公開鍵	スーパーコンピュータ SX-ACE 並列コンピュータ LX 406Re-2
file.cc.tohoku.ac.jp		データ転送サーバ
―	パスワード※3	利用者端末 大判カラープリンタ 三次元可視化システム

※1：パスワード認証でのログインは、2015 年 4 月 13 日 10:00 を以って廃止します。
※2：HPCI 課題、JHPCN-HPCI 課題で利用する場合は認証方式が異なります（GSI 認証のみ可）詳しくは、以下のリンク先の「HPCI ログインマニュアル」をご覧ください。
http://www.hpci-office.jp/pages/hpci_manuals
※3：センター内施設（利用者端末・大判カラープリンタ・三次元可視化システム）は、ローカルログインのため、今までどおりパスワード認証でご利用いただけます。利用にあたり、秘密鍵を持参する必要はありません。

2. 公開鍵暗号方式を使用する上での注意事項

以下のような行為は、不正アクセスのリスク（不正ログイン、クライアントのなりすまし、暗号化された通信の暴露、他サーバへの攻撃等）が非常に高く、大変危険です。ご注意願います。

- パスフレーズなしの秘密鍵を使用
- 秘密鍵、パスフレーズの使い回し
- 秘密鍵のメールへの添付、USB メモリやホームディレクトリへの保存
- 公開鍵と秘密鍵のペアを同一ノード上に保存

3. SSH アクセス認証鍵の生成

鍵を生成すると、鍵サーバへのログインが自動的にロックされます。一度ログアウトすると、以降は鍵サーバにはログインできなくなりますのでご注意ください。鍵の再登録が必要になった場合は共同利用支援係までご連絡下さい。本人確認の上、ロックを解除します。

(1) 鍵サーバに利用者番号と初期パスワード（変更している場合は変更後のパスワード）で SSH 接続します。

SSH アクセス認証鍵生成サーバ

```
key.cc.tohoku.ac.jp
```

リスト 1　鍵サーバへの SSH 接続例

```
localhost$ ssh 利用者番号@key.cc.tohoku.ac.jp
Password? : パスワードを入力
（初回接続時のメッセージ） : yes を入力

key$（コマンド待ち状態）
```

(2) 以下のコマンド（cckey-gen）を実行し、メッセージに従って公開鍵と暗号鍵の鍵ペアを作成します。必ずパスフレーズ（8 文字以上）を設定して鍵を作成してください。

リスト 2　公開鍵と暗号鍵の作成方法

```
key$ cckey-gen
Enter passphrase(8 or more characters) : パスフレーズの入力
Confirm : 同じパスフレーズを再度入力　　↑必ず設定してください

（生成された秘密鍵の表示）
-----BEGIN RSA PRIVATE KEY-----
gp5U3M6wVIvuGLX8OtYBAWC3WwNzX9TPu8eOCA9Pd/i6ijSNcVKp7lGJtuRzjfXV
  （中略）
FSwfyL63gRqxPZEmlcZzfDnhyX7ezdNNveZu37U/nq4TQj9+Q+RWHhjF9jwnuW6F
-----END RSA PRIVATE KEY-----
```

(3) 画面に表示された秘密鍵（---BEGIN RSA PRIVATE KEY--- から ---END RSA PRIVATE KEY--- まで）をコピー&ペーストし、ローカル PC にテキストファイルとして保存します。公開鍵は自動的にユーザのホームディレクトリに登録されます。秘密鍵はセキュリティを考慮して消去されます。

4.　公開鍵暗号方式によるログイン方法

4.1　Linux/OS X のターミナルソフトから接続する方法

生成された秘密鍵をファイル名「id_rsa_cc」として「~/.ssh/」以下に保存した場合

(1) パーミッションを 600 に変更します。(初回のみ)

リスト 3　パーミッションの変更

```
localhost$ chmod 600 ~/.ssh/id_rsa_cc
```

(2) i オプションで使用する秘密鍵を指定して SSH 接続を行います。
(i オプションを省略した場合は ~/.ssh/id_rsa あるいは ~/.ssh/id_dsa が利用されます)

リスト 4　ログインホストへの SSH 接続例

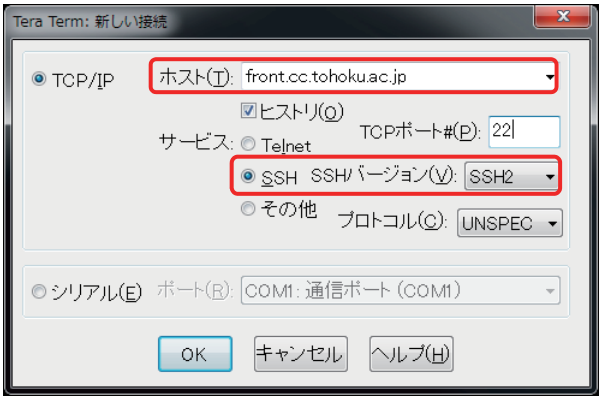
```
localhost$ ssh -i ~/.ssh/id_rsa_cc 利用者番号@front.cc.tohoku.ac.jp
Enter passphrase for key '/home/localname/.ssh/id_rsa_cc':パスフレーズを入力
（初回接続時のメッセージ） : yes を入力
```

```
front$ （コマンド待ち状態）
```

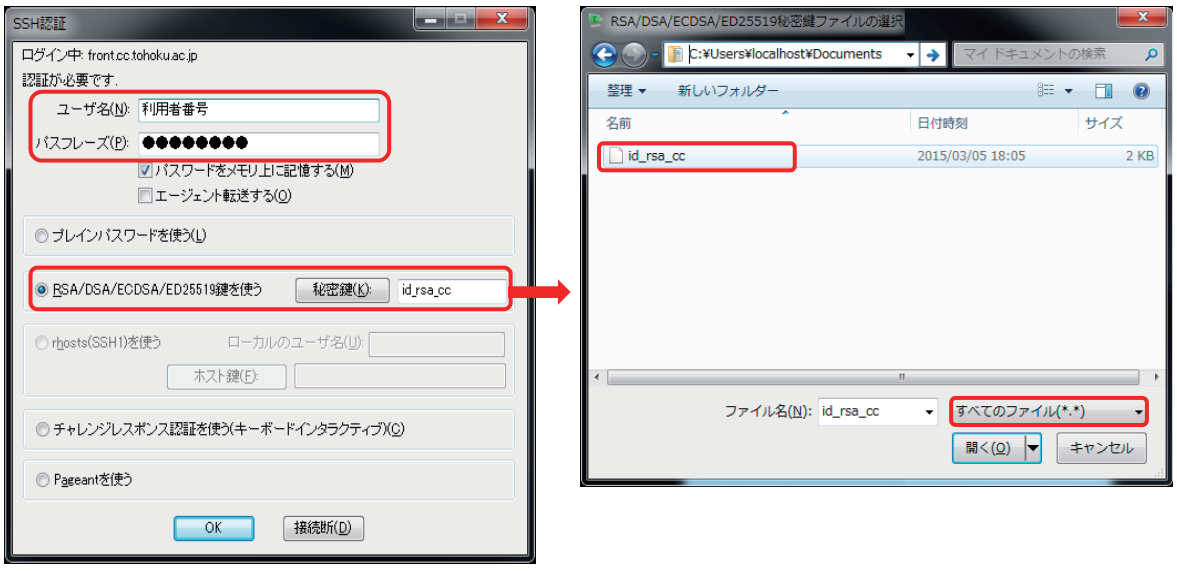
4.2 Windows の Tera Term から接続する方法

生成された秘密鍵をファイル名「id_rsa_cc」として「ドキュメント」以下に保存した場合

(1) 「ホスト名」を指定、「サービス」は SSH2 を選択し、[OK]を押下します。



(2) 「ユーザ名」に利用者番号、「パスフレーズ」に鍵ペアを作成した際に入力したものを入力、「RSA/DSA 鍵を使う」を選択し、「秘密鍵」に保存した秘密鍵のファイルを指定します。(秘密鍵ファイルの選択画面では、拡張子「すべてのファイル(*.*)」を選択します) [OK]を押下すると接続されます。



4.3 その他の OS／アプリケーションから接続する場合

各アプリケーションのヘルプを参照ください。

5. おわりに

本稿では、SSH アクセス認証鍵生成サーバの利用方法を紹介しました。公開鍵暗号方式により、より安全にシステムをご利用いただけるようになります。今後とも、研究の強力なツールとしてセンターのシステムをご活用いただければ幸いです。

［共同研究成果］

東北地方の農業における温暖化適応策と気象情報の高度利用

岩崎俊樹[1]、大久保さゆり[2]、菅野洋光[3]、紺野祥平[2]、島田照久[1,4]
福井真[1]、南野謙一[5]、宮脇祥一郎[6]、吉田龍平[1]

1 東北大学 大学院理学研究科
2 東北農業研究センター
3 中央農業総合研究センター
4 弘前大学 大学院理工学研究科
5 岩手県立大学 ソフトウェア情報学部
6 気象庁気候情報課

要旨

地球温暖化が顕在化し、適応策の立案が急務となっています。そこで、高解像度の気候モデルの予測結果を農業モデルに入力し、東北地方の農業における適応策について検討しました。温室効果ガスの排出がこのまま続けば、21 世紀末には、20 世紀末と比べて、東北の夏季の気候は

（1） 気温は約 3℃程度上昇する
（2） （気温の変動幅はあまり変わらず）温暖化後もヤマセによる冷夏が間欠的に発生する
（3） 短時間強雨が増加する

ことが予想されています。地球温暖化に対する第 1 の適応策は適地適作です。今後の気温上昇に適応するために高温障害に強い耐高温品種の導入が考えられます。しかし、一般に耐高温品種はヤマセなどの低温には脆弱です。そこで、年々の変動に対処するために、第 2 の適応策として、農業分野における、日々の気象予測の高度利用を提案します。アンサンブルダウンスケールという予測手法を用いると、ヤマセに特徴的な気温の地域特性も 5 日程度先まで精度よく予測できます。この気象予測の結果を農業モデルに入力し、確率的な農業気象情報を作成します。また、高解像度気象予測は農作物や農業施設への気象災害対策のためにも有効です。官・学・農業事業者で協力し、気象予測の利用技術の高度化に取り組むことが望まれます。

1. 本稿の目的

地球温暖化については、二酸化炭素をはじめとする温室効果気体の増加を抑えるために、さまざまな緩和策が検討されています。しかしながら、地球温暖化を完全に抑えることは難しく、温暖化時代を乗り切るための適応策の検討が急がれています。本稿では、文部科学省の気候変動適応研究推進プログラムの一環として実施した「東北地域のヤマセと冬季モンスーンの先進的ダウ

ンスケール研究（http://www.mext-isacc.jp/staticpages/index.php/report_iwasaki_j)」（2010 年 10 月-2015 年 3 月）に基づき、東北農業の温暖化対策について考察します。東北農業の温暖化対策に関心を寄せる多くの皆様にお読みいただき議論していただくことを期待しています。

我々は、21 世紀末を見据えた東北農業には、２つの温暖化適応策が必要だと考えています。第１の適応策は、適地適作や水資源管理の最適化などの長期的な対策です。温暖化に伴い、気温や降水量、日射量などが変化します。そのため、温暖化後の気候に適応した作物品種への変更や既存の品種の改良が必要となります。また、降水特性の変化や生育作物に合わせ、水資源の管理手法と防災対策を変える必要があります。地域の詳細な温暖化予測に基づく長期的な対策について、第２節にまとめます。第２の適応策は、気象予測情報の高度利用です。本研究においては、実際の気象予報に用いて地域気象を詳細に表現する、アンサンブルダウンスケール予報（詳しくは 2.1 節および 3.1 節を参照）という技術による気象予測と農業モデルを組み合わせ、高解像度の確率的な農業気象情報作成の実験を行いました。ここでは、その手法を紹介し、実用化に向けた課題について議論します。また、本研究の実施期間中、4 年間にわたり、農業事業者等に向けて試験的に農業気象情報を提供し、ニーズを把握するためのアンケート調査を実施しました。以上の研究・調査に基づき、気象予測の高度利用に関する提案を行います。これらの内容は第 4 節にまとめます。

温暖化予測には不確実性がつきものですし、気象予測は完全ではありません。適応策を検討する際には、温暖化予測の不確実性を考慮することが必要です。高度化が進む温暖化予測を常に検証し、適応策を改訂していく必要があります。また、気象予測は将来改善が見込まれます。予測精度を考慮した確率情報の利用を推進することが今後はますます望まれます。

当該研究の参加機関は、東北大学、弘前大学、岩手県立大学および、（独）農研機構東北農業研究センターで、協力機関は、気象庁気候情報課、気象庁気象研究所、仙台管区気象台、岩手大学です。また、東北各県の農業試験場（青森県産業技術センター、岩手県農業研究センター、宮城県古川農業試験場、秋田県農業試験場、山形県農業総合研究センター、福島県農業総合センター）の関係者とも年２回開催する研究会等を通じて議論を行いました。本稿の直接の執筆者は標記の通りですが、取りまとめた内容は、当該研究への参加・協力機関および、連携する気候変動適応研究推進プログラムとの共同研究に基づくものです。現在気候および地球温暖化予測のダウンスケールには東北大学サイバーサイエンスセンターのスーパーコンピュータを利用しました。

2. 東北の夏の温暖化

2.1. 東北の夏への温暖化影響（ヤマセ、異常高温、短時間強雨と無降水日）

2.1.1. 地球温暖化の影響は地域スケールではどのように現れるのか？

地球温暖化の影響が顕在化するにつれて、温暖化適応策や温暖化に起因する様々なリスク管理を推進していくことが重要な課題になっています。このような社会的要請に応えるためには、世界の気象機関・研究機関が実施している全球気候モデル（用語解説参照）による地球規模の温暖化予測（起こりうる社会の変化–シナリオ（用語解説参照）–を想定して、それが気候にどのような影響を与えるかをシミュレーションによって予測すること）から、地域レベルでの気候変化を明らかにする必要があります。このような社会的要請に応えるための方法の一つが、力学的ダウンスケーリング（領域気象モデルを用いた再計算によるデータの詳細化のこと（用語解説参照）。以下、単にダウンスケーリング）です。温暖化予測を行うほとんどの全球気候モデルの空間解像度は 100km より粗く、このままでは東北地方の気候にとって重要な役割を果たす脊梁山脈を十分に解像することはできません（図 1 左）。つまり、東北地方の詳細な気候は再現できないことになります。これを解決するために、全球の温暖化予測の結果を反映させながら、着目する地域について、高解像度の地形（ここでは例として 10km 格子の地形を利用、図 1 右）を入力してシミュレーションをやり直します。こうして、全球的な温暖化による気候変化が、地域スケールでどのように現れるかが明らかになります。

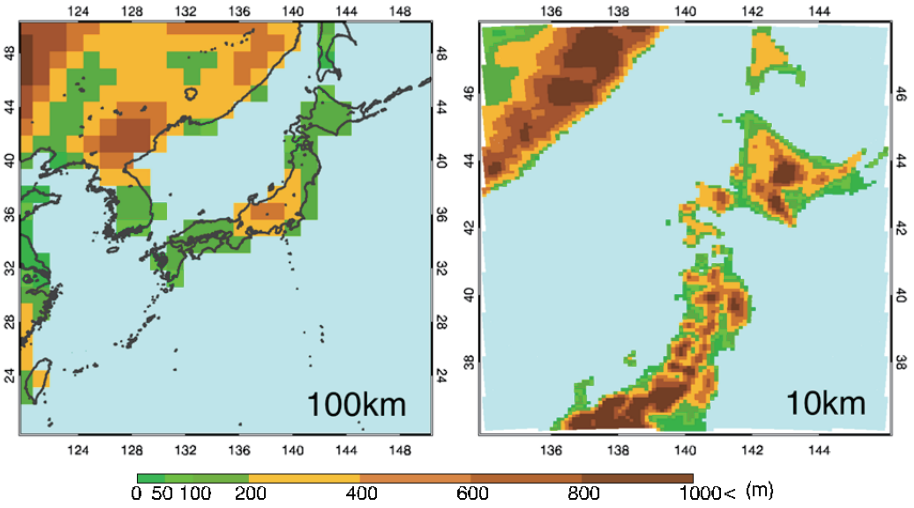


図 1　（左）100km と（右）10km の格子間隔で表現される地形。

気象庁では、全球気候モデルによる実験結果を「地球温暖化予測情報」として取りまとめて公開しています[1]。これは、全球気候モデルによる地球全体の温暖化予測を、日本周辺に絞ってダウンスケーリングして、地球温暖化が日本の気候にどのように影響するのかを調べたものです。最新の「地球温暖化予測情報第 8 巻」では、21 世紀末に大気中の二酸化炭素が約 700ppm に達す

るとしたシナリオ（SRES A1B: 用語解説参照）を考えています。このシナリオでは、21 世紀末の日本付近で、20 世紀末に比べて、年平均気温が約 3℃上昇し、短時間強雨や無降水日が増加する可能性が示されています。さらに、季節と大まかな気候区分（北日本の太平洋側と日本海側等）に分けて、今後 100 年間に起こりうる気候の変化が示されています。

我々のグループでは、東北地方における地域のニーズ（地域の産業や生活スタイルへの適応策、防災・減災対策）に応えるための温暖化予測情報を提示するために、気象庁の「地球温暖化予測情報第 8 巻」に用いられている全球気候モデルを用いて、独自に力学的ダウンスケーリングを行い詳細に解析しています。ここでは、東北地方の夏季気候に着目し、ヤマセによる低温、異常高温、短時間強雨と無降水日という観点から、気候の将来変化の特徴を示します。なお、解析期間は、「地球温暖化予測情報第 8 巻」に対応させて、現在気候を 1980-1999 年、将来気候を 2076-2095 年としています。

2.1.2. 低温をもたらすヤマセは将来も発生するのか？

東北地方の夏季気候は、間欠的に発達するオホーツク海高気圧から北日本の太平洋側に吹きつける冷たい東風（ヤマセ）の影響を大きく受けます。この東風とともに、背の低い（<1000m）下層の冷気が北日本のオホーツク海側と太平洋側に押し寄せて低温をもたらします。さらに、下層雲や霧が低地を覆い、放射冷却により低温を維持します。ヤマセの発生頻度や低温の程度は、オホーツク海高気圧の消長に合わせて経年変動が顕著で、卓越したヤマセは、夏季の異常気象（冷夏）と言われてきました。一方で、ヤマセの持続期間や低温の程度に経年変動があるとはいえ、毎年繰り返し発生しているという事実も重要です（例えば、2013 年は、7 月 17–23 日にかけて東北地方の太平洋側で、顕著な低温低日射の状態が続きました）。ヤマセのこのような特徴は、地域気象・気候、そして社会活動（農業、海洋安全、航空安全等）に影響します。例えば、農業にとっては、毎年発生する数日程度のヤマセによる低温寡照期が、農作物の重要な成長期に当たるかどうかが大きなリスクとなっています[2]。このように、ヤマセの理解は、特に東北地方の冷害の歴史[3]を背景に、社会的に重要な課題でありつづけています。

では、地球温暖化が進んだ将来、ヤマセの発生はどのように変化するのでしょうか。将来気候においても、ヤマセは発生することがこれまでの研究で示されています。ここでは、オホーツク海高気圧が発達し、東北地方の太平洋岸に向かって東風（ヤマセ）が吹いているときの平均的な状況を考えます。現在気候について、夏季の平均気温と比べると、北日本のオホーツク海沿岸と太平洋沿岸は低温になり、日本海側はやや高温になります（図 2 左）。ヤマセによる低温が顕著な地域は、日高山脈、八甲田山、北上山地、蔵王連峰等の山地の東側に見られ、下層の冷たい空気が山地にせき止められることを示しています。将来気候においても、同じ状況では、現在気候とほぼ同様の気温の分布が見られます（図 2 中）。この気温分布は、将来気候の夏季平均からの差であることに注意してください。ヤマセが卓越しているとはいえ、気温は温暖化に伴って、現在気

候のヤマセ卓越時より約 3℃高くなっています（図 2 右）。ただし、現在気候と同じ条件の事例発生数は、将来気候では若干（約 2%）減少しています。まとめると、温暖化によって気温上昇した後も、将来気候の平年値を基準に考えると、ヤマセが北日本に向かって吹き、太平洋側と日本海側の気温差を大きくする状況は、将来も発生すると言えます。

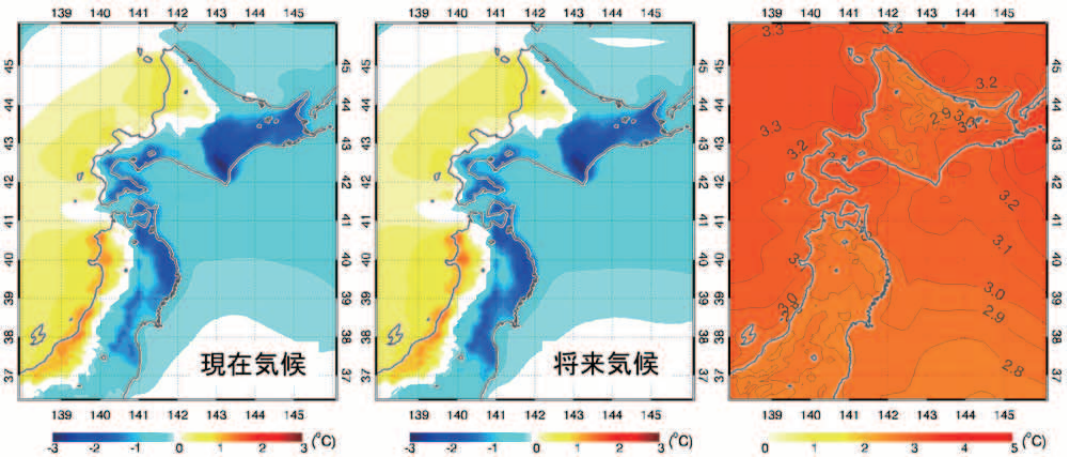


図 2　（左）現在気候と（中）将来気候の 6-8 月のヤマセ卓越時について、気温の平年値からの差。現在気候と将来気候の平年値は、現在気候と将来気候それぞれの夏季平均である。（右）ヤマセ卓越時の気温の将来気候と現在気候の差。

2.1.3 異常高温は東北地方全体で顕著になる

地球温暖化による平均的な気温上昇とともに、猛暑が増加しています（2010 年は、日本各地で記録的な高温が観測されました）。猛暑による異常高温は、健康被害（熱中症、睡眠障害等）を引き起こし、冷房のための電力や都市ガスの供給に影響を与えます。また、農作物の高温障害が発生します[7]。「地球温暖化予測情報第 8 巻」[1]によると、21 世紀末の北日本では、夏季の平均気温は約 2.9℃上昇し、一年で最も気温が高くなる時期が約一月早まる傾向が予測されています。また、夏季の極端な高温の日の最高気温についても、将来は 2〜3℃程度の上昇が予測されています。

ここでは、夏季の極端な高温の日の最高気温の地理的分布を詳細に見てみます。「地球温暖化予測情報第 8 巻」[1]と同じく、日最高気温の 95 パーセンタイル値を指標とします。95 パーセンタイル値とは、標本の各データを値の小さい順に並べた時、値の小さい方から数えてデータ数の 95 パーセントに当たる値のことです。つまり、日最高気温の 95 パーセンタイル値は、日最高気温の低い方から 95%の値、逆に言うと日最高気温のデータ中で上位 5%に位置する値となります。このため、特に高い日最高気温を表す統計値となり、極端現象の一つの指標となります。例えば、現在気候における 7–8 月の仙台の日最高気温の 95 パーセンタイル値は、32.1℃です（図 3 左）。7–8 月について、日最高気温の 95 パーセンタイル値の将来変化をみると（図 3 右）、東北地方の

太平洋沿岸で約 3℃の上昇が見られる一方、日本海側では約 2℃です。これまでは太平洋側よりも日本海側の平野部で顕著な高温が観測されることが多かったのですが、将来気候では太平洋側での異常高温が増え、北日本全体で同じ程度の異常高温が発生することになります。

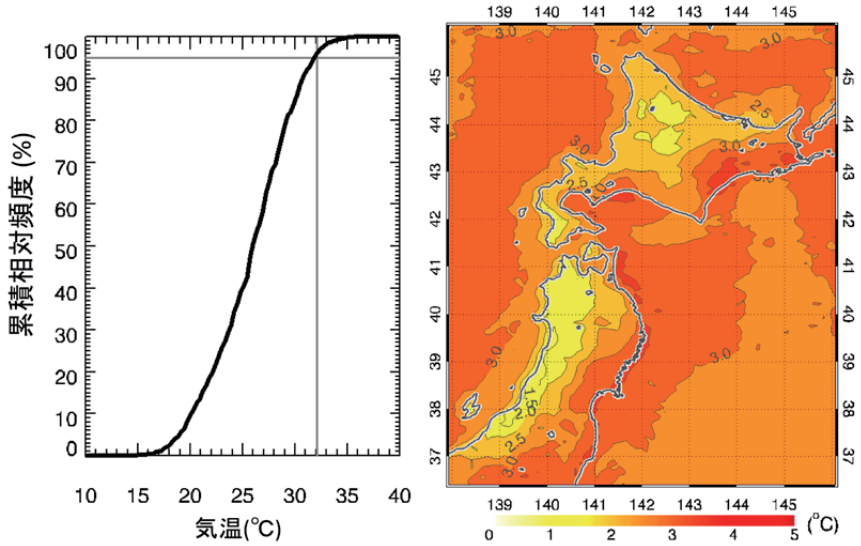


図 3　（左）仙台における日最高気温の累積頻度分布。灰色の線の交点が 95 パーセンタイル値を表す。（右）日最高気温の 95 パーセンタイル値の将来変化。

2.1.4. 短時間強雨の少なかった東北地方でも将来は増加

近年、大雨に伴う水害や土砂災害が多く報告されるようになり、地球温暖化との関連性が注目されています。東北地方では、2013 年 7 月に山形県および福島県で梅雨前線に伴う激しい雨が発生し大きな被害が出ました。現在のところ、日本全体の降水量の長期的な変化傾向は見られないものの、日降水量が 100 mm あるいは 200mm 以上となる大雨の年間日数は増加傾向にあることが示されています[5]。その一方で、降水日数は減少する傾向にあります[5]。これらの傾向は、「地球温暖化予測情報第 8 巻」[1]で示された将来変化と一致しています。つまり、現在観測されている傾向は今後も継続し、将来は大雨の発生回数は多くの地域で増加し、無降水日（日降水量が 1 ミリ未満の日）数も多くの地域で増加すると見込まれています。また、年降水量は、全国的に増加することが示されています。

現在気候の東北地方について、我々のデータを用いて詳しく見てみます。夏季（6–8 月）の降水量の地理的分布をみると（図 4 左）、山地に沿った地域で特に降水量が多くなっています。将来気候では、夏季降水量は、平均して約 12%増加すると予測されています。 1 時間降水量が 50mm を越える短時間強雨については、現在のところ、東北地方の発生頻度は全国平均の約 20%であり、他の地域と比べて発生頻度は少ない状況です[1]。しかし、東北地方においても短時間強雨が将来増加する傾向が現れており（図 4 右）、全国的な特徴[1]と一致しています。今後、大雨や短時間

強雨の頻度が増加するにつれて、水害や土砂災害に対する東北地方の脆弱性が明らかになる可能性があります。将来の大雨や短時間強雨の増加を想定して、防災情報等に関わる降水量の指標を見直すとともに、新たな防災減災対策が必要になってくると考えられます。なお、無降水日は東北地方の夏季に限っては、やや減少傾向になっています。

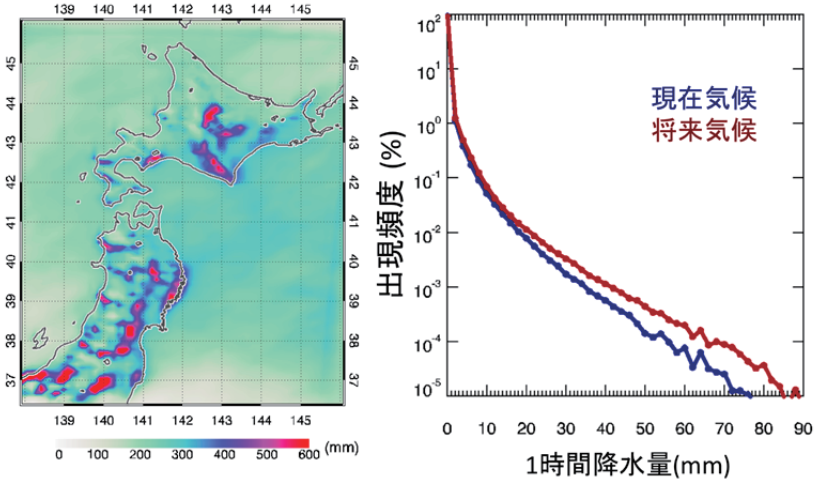


図 4　（左）現在気候における 6-8 月の降水量。（右）現在気候と将来気候の 1 時間降水量の発生頻度分布。

2.1.5. まとめ

気候変動の影響が顕在化する中、温暖化に対する緩和策とともに、適応策や気候変動に起因する様々なリスク管理を推進して行く必要があります。これを地域の実情とニーズに合わせて行っていくためには、全球の気候予測が地域スケールでどのように現れるのかを明らかにしなければなりません。東北大学や東北農業研究センターでは、ダウンスケーリングという手法を用いて、東北地方に特化した将来気候と温暖化の影響評価の研究を進めています。

2.2. 東北農業の将来

2.2.1. はじめに：現在の品種で安定したコメ生産は可能か？

温暖化が水稲生育に与える影響は、現在各地がおかれる状況によって様々です。九州や四国といった温暖な環境では、近年の昇温によってコメの品質や収量の低下が起こっています[6]。一方で、もともと冷涼な環境の東北では増収傾向であると報告されています。そのため、温暖化の進行に伴って、東北地方はコメ生産のより重要な地域となっていくと考えられます[7]。我々のグループは（独）農業環境技術研究所と共同で、東日本における温暖化の水稲生育に与える影響を調べました。

2.2.2. 現行品種を維持した場合、増収が見込まれるが高温障害リスクも増加する

気温や日射量といった気象条件から日々の生長量を計算する水稲生育モデル Hasegawa/Horie を用いて、水稲生長のシミュレーションを行いました。気象条件には前節の温暖化シナリオを用いて、温暖化が水稲に与える影響を算出しました。

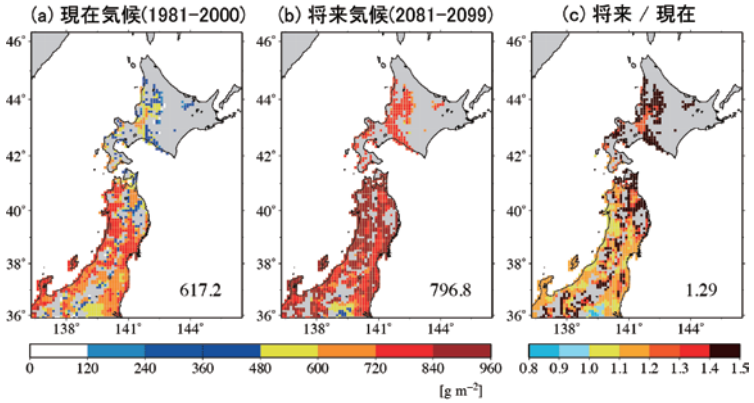


図 5　各県で最大の作付面積を持つ品種を将来も維持した場合の収量変化。(a) 現在気候、(b) 将来気候、(c) 将来気候値の現在気候値に対する比。各図右下の値は領域平均値で、灰色の部分は 10km メッシュ内の水田面積率が 1%未満の地点を表す。

図 5 は、各県で栽培されている主要な品種（現行品種と定義します）を将来もそのまま維持した場合の収量分布です。現在気候では 720-840 (t m^{-2})は高収量といえますが、平均気温の上昇と CO_2 の施肥効果によって、将来気候ではそういった値やそれ以上の値が本州の多くの地域で見られるようになりました。また、増収率は本州の山岳部や北海道で高く、現在は十分にとれなかった地域での増加が顕著でした。現在報告されている増収傾向は今後も続くと見込まれます。

つづいて、水稲生育における高温・低温による不稔率（＝実がつかない割合）を推定しました（図 6)。現在の東北地方では、ヤマセによる冷害が水稲生育に深刻な問題です（高温の不稔率を 1 としたとき、低温の不稔率は 72.3 倍)。温暖化に伴って、現在冷涼な東北地方においても高温による不稔率の増加が予測されますが、依然として低温による不稔率の方が高いと推定されまし

た（同 3.3 倍）。将来気候においては、高温による不稔率は低温による不稔率の約 30%まで増加するため、将来は冷害への継続した警戒と顕在化する高温障害への対応の両面が必要になると言えます。

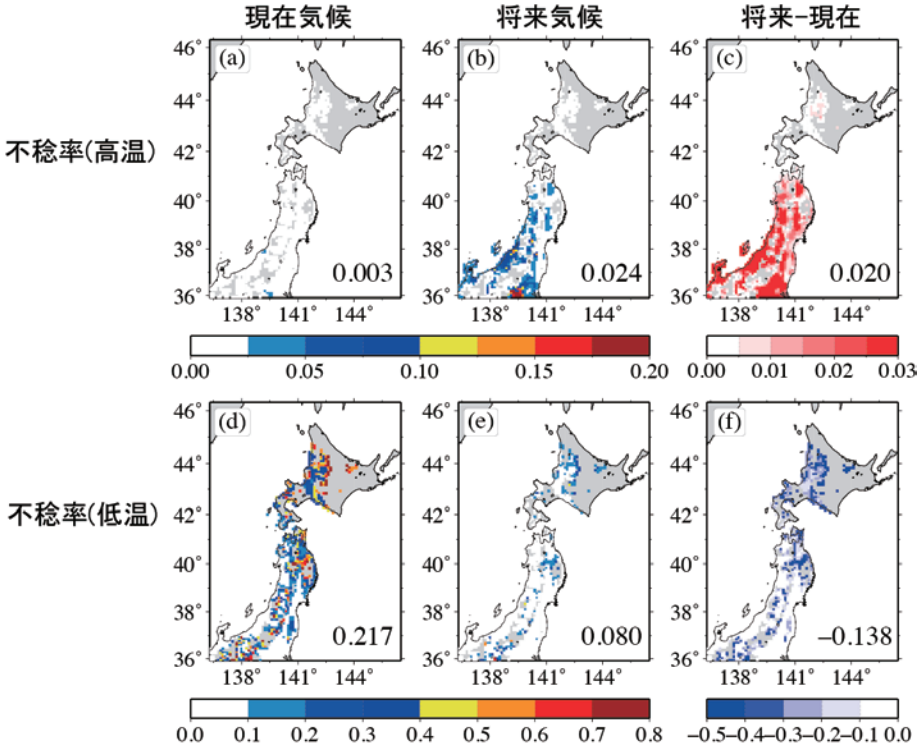


図 6　図 5 と同様。ただし、(a-c)は高温による不稔率、(d-f)は低温による不稔率。

コメ生産では、品質も重要な要素です。出穂したあとの生殖生長期に高温になると、充分に登熟できず、白く濁った粒が増えます（白未熟粒（しろみじゅくりゅう）と呼ばれます)。白未熟粒の増加により、一等米の比率が下がります。登熟期の日平均気温が 26℃を越えた分を足し合わせた積算気温が、品質のよい指標として用いられるため、先の水稲生育モデルで算出しました（図 7)。

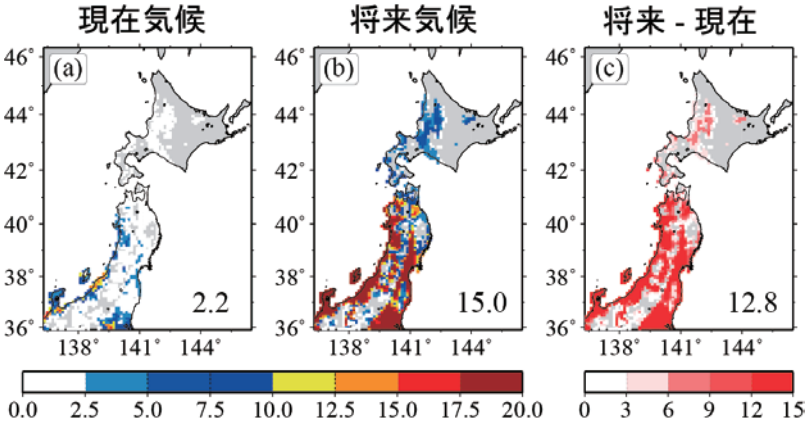


図 7　登熟期の日平均気温が 26℃を越えた分を積算した気温(℃)。

収量の場合と同様に、品質についても登熟期の高温は現在ほぼ見られませんが、将来気候では東北の平野部で多く見られるようになりました。ごくわずかではありますが、北海道でも登熟期の高温が推定されています。東日本全体では、登熟期の”積算”気温は将来気候で 10℃以上増加すると推定されるため、品質の面からも高温障害への警戒が必要になっていくといえます。

2.2.3. 安定したコメ生産には耐高温品種の導入と現行品種の維持が大きな柱

再び収量の話にもどります。冷害と高温障害の影響を考慮して、現行品種よりも高収量・不稔率低下を実現できる品種を検討しました。検証に使用した品種は、現在日本で主に栽培されている 10 種です（表 1)。現行品種よりも収量あるいは不稔率のいずれかの点で優れる品種は多くありましたが、複合条件では低温による不稔率低下の実現が厳しいために品種の選択肢が狭まり、中京以南で栽培されている数種類に限られました（図 8)。一方で、半分ほどの地点では現行品種が最適と選定され、安定した食糧供給のためには、耐高温品種の導入と現行品種の維持が大きな柱といえそうです。

表 1　水稲生育モデル Hasegawa/Horie で使用した品種リスト。

1.	ひとめぼれ	2.	きらら 397	3.	ヒノヒカリ
4.	あさひの夢	5.	あきたこまち	6.	あいちのかおり
7.	はえぬき	8.	こしいぶき	9.	コシヒカリ
10.	キヌヒカリ				

2.2.4. 最後に：結果の解釈について

本課題によって得られた結果は 1 モデル・1 シナリオに限られており、将来気候において必ずこのようになるというものではないことに注意が必要です。さらに、コメ生産での適応策として栽培品種の変更に着目していますが、品種改良や水田への移植日変更といった他の様々な方策も考えられます。そのため、得られた結果は数ある可能性のうちの 1 つとして捉えるとともに、他の全球気候モデルや排出シナリオによるデータも利用した、大気側・水稲側のそれぞれ多様な方策を考慮した、より包括的な解析が今後必要です。

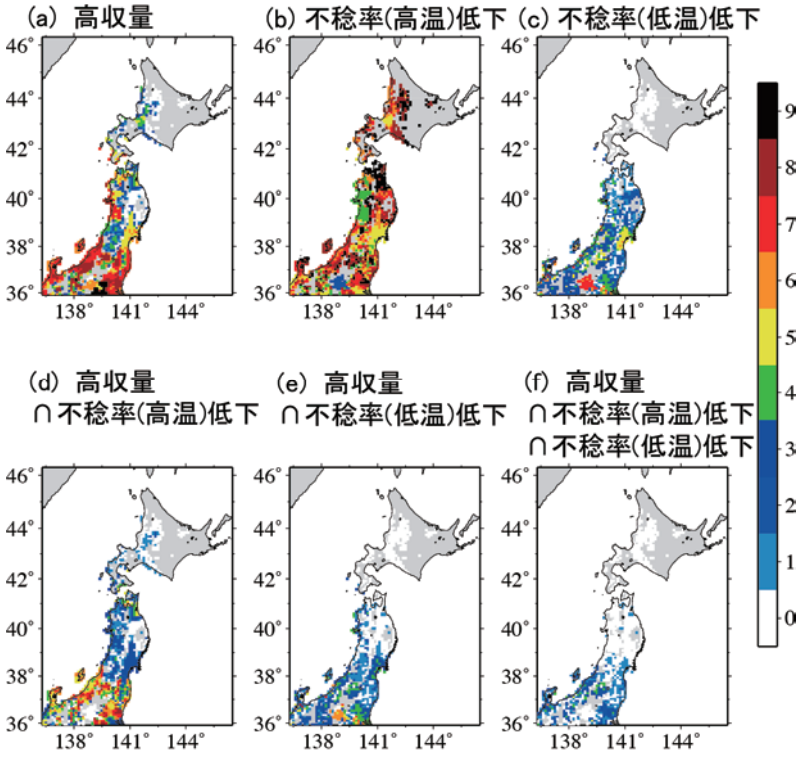


図 8　将来気候において、現行品種よりも好ましい条件となる品種数の空間分布。(a)高収量、(b)不稔率(高温)低下、(c)不稔率(低温)低下、(d)高収量∩(かつ)不稔率(高温)低下、(e)高収量∩不稔率(低温)低下、(f)高収量∩不稔率(高温)低下∩不稔率(低温)低下。

3. 気象予測情報の高度利用の手引き

3.1. アンサンブルダウンスケール予報

3.1.1. はじめに

100 年後の地域の気温は、地球温暖化により、およそ 3℃上昇することが予測されています。他方、年ごとの気温の変動（自然変動）の大きさについて、例えば、ヤマセの顕著な八戸の 7 月の場合で、月平均気温の標準偏差は 1.89℃です。暑夏と冷夏では、3℃以上（1.89℃の 2 倍）の気温差があることになり、今後 100 年間の地球温暖化による上昇に、およそ匹敵します。地球温暖化の長期的な適応策が機能するためには、暑夏年や冷夏年などの自然変動（年々変動）に対しても適応できなければなりません。

年々の変動に対処する一つの方法は、今後さらに精度向上が期待できる気象予測を高度利用することです。気象予測の農業利用では、地上気象要素（気温、湿度、風、日照）を 1 ㎞程度の高解像度で、かつできるだけ先まで予測することが望まれます。アンサンブルダウンスケール（図 9）という手法を用いれば、5 日程度先まで、有効な高解像度予測情報を作成できます。

3.1.2. アンサンブル予報

今日の天気予報は、数値気象予報モデルの予測に基づいて作成されています。数値気象予報モデルは、現在の大気の状態（実況値）を初期条件とし、物理法則に基づいて時間積分することで、将来の大気の状態を予測します。しかしながら、大気の状態は、常に完全に把握されている訳ではありません。観測を実施している場所は相当に限られます。例えば、現状の観測システムでは、大気の鉛直構造を知るためのラジオゾンデ観測は、およそ 300 ㎞間隔で実施されており、大気状態を観測だけで実況値を細かく解析することはできません。また、観測誤差をなくすことも不可能です。このため、初期条件（実況値）には必ず誤差が含まれます。誤差は予測時間とともに成長し、やがて、予測は誤差に凌駕されて利用できなくなります。

気象庁の発表する週間予報とそれより長い予報では、不確実性の影響を抑え、信頼性の高い予測情報を作成するために、アンサンブル予報という手法が採用されています。アンサンブル予報では、観測誤差に見合うだけ少しずつ異なるたくさんの初期条件を準備し、それぞれの初期条件から予測（メンバー予報）を実行し、その結果を統計処理して予測情報を作成します。

図 10 は、およそ 1500m 上空の仙台付近の気温に対する全球アンサンブル予報の 1 例です。細い実線は各メンバーの予報を表し、太い実線はすべてのメンバー予報の平均値を表します。各メンバーの予報は成績の良いものも悪いものもあります。太い実線の予報の平均値は大きく外すことは少なくなります。予報メンバー間の相違（バラツキ）は、予報の不確実性を表しており、その標準偏差はスプレッドと呼ばれ、予報の信頼性の指標とされます。このケースでは、10 日予報でもスプレッドは比較的小さいままですが、通常は、スプレッドは予報時間とともに大きくなり、予報精度が低下することを示唆します。また、指定した気温の範囲に入る予報の数と全体の予報

の数の比をとれば、予報を確率で表すことも可能です。例えば、図 10 で 5 日後に気温が 12.5 ± 1℃の範囲に入る予報は 5 本で、その確率はおよそ 60%となります。このように、アンサンブル平均をとることによって、誤差が極端に大きくなることを防ぎ、予測の確からしさを知り、確率予報を行うことができます。

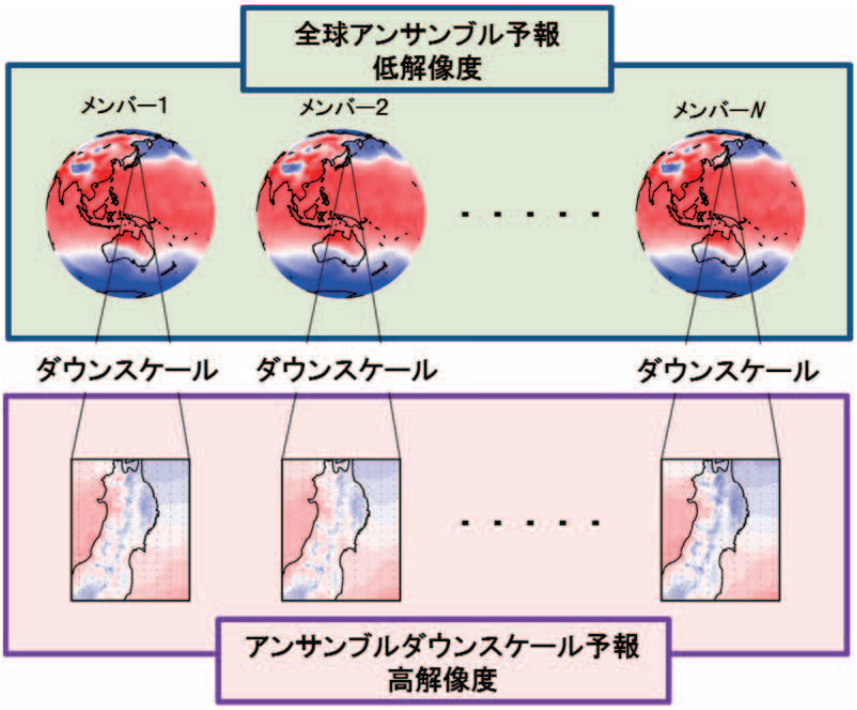


図 9　アンサンブルダウンスケール予報の模式図。

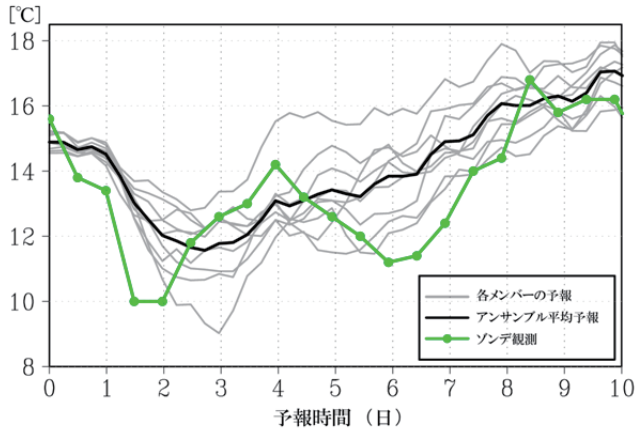


図 10　全球モデルによるアンサンブル予報の 1 例。仙台上空の 850hPa（およそ 1500m）の気温の 10 日予報。細線は 9 メンバーのそれぞれの予報結果で、太い実線はアンサンブル平均値、緑は観測値。

3.1.3. アンサンブルダウンスケール予報の予測可能性

農業などでは、地上気象要素に関する面的な高解像度の情報を必要としています。しかし、全球大気を数kmの格子間隔で予報するには大変な計算コストがかかります。実際に、気象庁で週間アンサンブル予報に用いられている全球数値気象予報モデルの格子間隔はおよそ 40 kmで、個々の水田・畑は解像できません。そこで利用するのが 2.1 節で説明したダウンスケールという技術です。

特に、東北地方で問題となるヤマセにおいては、冷気は谷間に向かって侵入するので、細かな地形の効果が大きな影響を与えます。このような現象の予測には、高解像度の数値予報モデルの利用が必要です。ダウンスケール予報では、比較的広い領域の予報を低解像度モデル用いて計算し、高解像度モデルのための時間的に変化する側面境界条件を作成します。この側面境界条件を利用し、高解像度モデルにより、狭い領域を高解像度で計算します。

図 11 はそのようなアンサンブルダウンスケール予報の例です。このダウンスケール予報実験では、全球数値予報モデル(格子間隔 125km)による週間アンサンブル予報の結果と、日本全体（25 km格子)、東北地方（5 km格子）の２つの解像度の異なる領域数値予報モデルを利用します。まず、全球モデルによる週間アンサンブル予報の各メンバーの予報を、それぞれ初期条件および側面境界条件として、25 km格子の領域数値予報モデルによるアンサンブルダウンスケール予報を行います。次に、25 km格子の領域数値予報モデルの予報結果を初期値・境界値として 5 km格子の数値気象予報モデルによりアンサンブルダウンスケール予報を行います。図 11 にはそれぞれの数値気象モデルによる 66 時間予報のアンサンブル平均を表しています。全球予報の結果では、ヤマセの事例であるにもかかわらず、太平洋側の気温が低くならず、東北がほぼ一様な気温となっています。これがダウンスケール予報では、太平洋沿岸地域と仙台平野の低温が明瞭に示され、地域の特徴が良く表現されています。

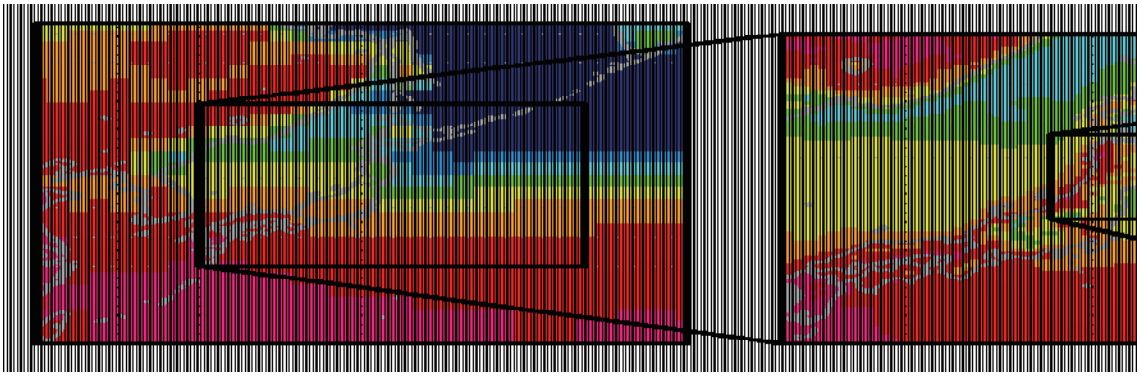


図 11　実験システムで用いたアンサンブルダウンスケール予報の計算領域と気温の予報例。　左より全球予報（125 km格子で表現)、25 km格子、5 km格子モデルの 66 時間予報の結果（9 メンバーの平均値）で、枠線はそれぞれ 25 km、5 kmの計算領域を表す。

アンサンブルダウンスケール予報の場合、局地循環が良く表現される反面、その誤差は大きく

なります。低解像度モデルの誤差に、高解像度モデルの誤差（局地循環の誤差）が加わるためです。各メンバー単独の予報を利用した場合は、低解像度モデルの誤差が高解像度モデルで増幅され、利用者には使いにくい情報となってしまいます。これに対して、アンサンブル平均をとるとノイズが相殺されて消え、シグナルが残り、実効的な予測可能時間を延ばすことができます。

図 12 は、ヤマセ型の気温分布パターンの時間的な変動について、40 事例のアンサンブル平均予報の成績を、観測値に対する相関係数と回帰係数で表したものです。大まかには、相関係数は時間的な変動パターンの類似性を表し、回帰係数はその変動の大きさの再現性を表します。ヤマセ型の気温分布パターンの変動の類似性（相関係数）は、アンサンブルダウンスケールを行ったかどうかにあまり依存せず、5 日程度予測可能であることを示しています（注: 相関係数が 0.5 を切るまでの時間で判定)。他方、変動の大きさ（回帰係数）は、アンサンブルダウンスケールを行なっていない全球アンサンブル予報では実際の 2 割以下ですが、25 ㎞格子にアンサンブルダウンスケールした場合ではおよそ 7 割、5 ㎞格子にアンサンブルダウンスケールした場合では 8 割以上再現されています。すなわち、アンサンブルダウンスケールにより解像度を高くすることで、小さなスケールの現象の予測が有効となることがわかります。ただし、解像度を高くすると計算量が著しく増えるので、費用対効果をよく考えて解像度を選ぶ必要があります。将来的に全球数値予報の改善や、メンバー数の増加によって、アンサンブルダウンスケール予報の精度向上を図ることができます。

最後にアンサンブルダウンスケール予報の農業利用について整理します。アンサンブルダウンスケール予報の各予報（メンバー）によって、起こりうる気象状態を確率的に予報することができます。各予報の結果を、数値農業モデルに入力し、得られた結果を統計処理して不確実性を考慮した農業気象情報を作成することが可能です。例えば、いもち病の予察の場合は、いもち病発生のケースと発生しないケースを分けて、加算し、確率情報として利用者に提供することが可能です。詳しい説明は次節で行います。

3.1.4. まとめ

温暖化後も予想される間欠的なヤマセによる冷夏などの自然変動に対応するためには、日々の気象予測の高度利用が重要です。アンサンブルダウンスケール予報は、より小さなスケールの現象を考慮しながら、ヤマセ型の気温分布を 5 日程度先まで予測することが可能です。アンサンブルダウンスケール予報を、農業の数値モデルと組み合わせれば、確率的な農業気象情報を作成することができます。

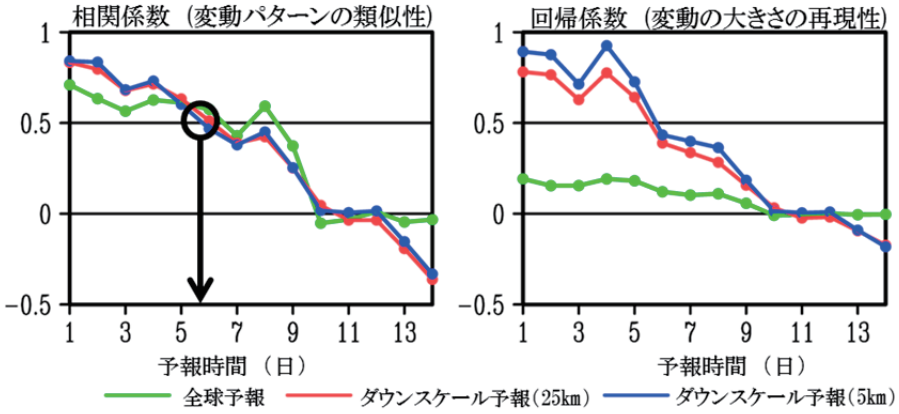


図 12　アンサンブルダウンスケール予報実験におけるアンサンブル平均値予報の、ヤマセ型の気温パターンに対する相関係数と回帰係数。40 事例の平均。

3.2. 農業気象情報へのアンサンブルダウンスケールデータの利用
–葉いもち予察モデルを例として

3.2.1. はじめに

アンサンブル予測や気象データのダウンスケールを用いることで、高解像度の気象予測情報を作成することができます。これをうまく利用することで、農業へ役立てられる可能性があります。ここでは、葉いもちの予察を例に、気象データの農業への利用について紹介します。

3.2.2 イネいもち病について

いもち病は、イネがいもち病菌に感染して起きる病気で、蔓延すると大幅な減収を招くため、昔から恐れられてきました。特に冷夏で低温や低日照が続く場合は、稲の抵抗力も弱まり、低温による冷害とあわせて病害も発生し被害の拡大をもたらします。いもち病を防ぐには、予防的な農薬とともに、発生が予測された際の殺菌剤の使用が効果的です。そこで、東北農業試験場（現農研機構東北農業研究センター）では、アメダス気象観測所の気象データを用いて、いもち病のうち、葉いもちの発生予察を行うコンピュータープログラム「BLASTAM」を 1988 年に開発しました[8、9]。これは、アメダスで観測される気象 4 要素（気温、雨量、風、日照時間）を用いて、イネの葉の濡れ時間を推定し、連続濡れ時間がしきい値を超えると「感染可能性あり」の数字（1,2,3,4,10）で示すものです。

3.2.3 アンサンブルダウンスケール予報の農業モデルへの適用

アンサンブルダウンスケール予報（3.1 節）とは、アンサンブル気象予測という手法の予測結果を、細かい空間解像度に展開（ダウンスケール）したものです。本節では、アンサンブルダウンスケールデータを農業気象情報として利用する場合の利用可能性について、葉いもち予察を例に紹介します（図 13）。

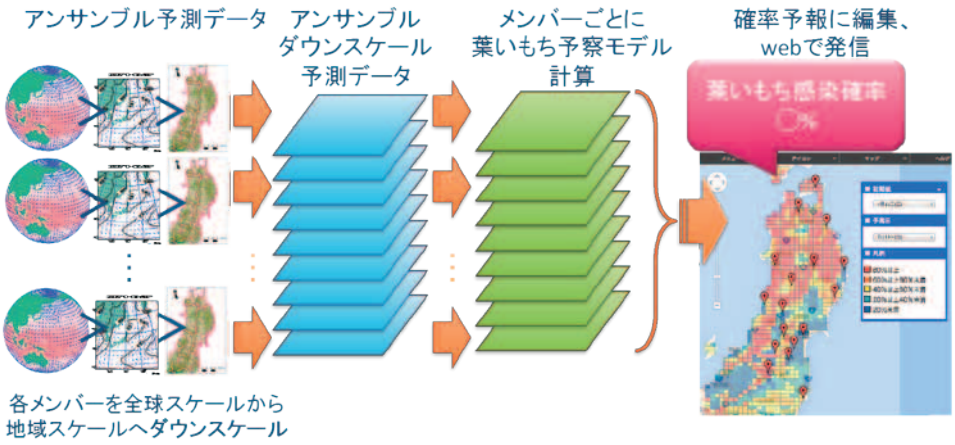


図 13　アンサンブル予測結果による葉いもち病発生確率予測の概念図。
1 つの気象予測計算（アンサンブルメンバー）毎にダウンスケール計算、農業モデル計算を行う。

ここでは、近年の東北地方での代表的な冷夏年である 2003 年夏季を対象としたアンサンブル予測データを使用して、BLASTAM の計算を行いました。実験対象期間は 2003 年夏季の、それぞれ 6 月 20 日、6 月 30 日、7 月 10 日、7 月 20 日を初期値とする 4 期間×各 2 週間としました。予測実験の結果を、力学的ダウンスケールによって空間解像度を 1.25°（約 110km）間隔から 5km 間隔にしたデータを用いて、BLASTAM で葉いもち病の感染好適条件を求めました。9 つの予測結果それぞれを用いて BLASTAM を計算し、メッシュ毎に「感染可能性あり」の出現割合を求め、葉いもち発生確率としました（図 13）。なお、ここでは BLASTAM の結果が 1, 2, 3, 4, 10 のいずれかであれば「感染可能性あり」としました。

対照的な事例となった 6 月 20 日と 6 月 30 日を初期値とする期間を例に結果を示します。2003 年 6 月 20 日初期値による結果では、感染好適条件の出現確率の分布が、アメダス観測値の統計的ダウンスケールによる 1km メッシュ気象データを用いて求めた感染好適条件の分布と期間を通じておおむね一致していました（例として、図 14a）。それに対して、2003 年 6 月 30 日を初期値とした期間は、特に期間の後半で予測自体が大きく外れたため、感染好適の分布も計算によるものと観測値によるものとで大きく異なる日が多くなりました（例として、図 14b）。

（a） 2003 年 6 月 20 日初期値、5 日目の例　　（b）2003 年 6 月 30 日初期値、7 日目の例

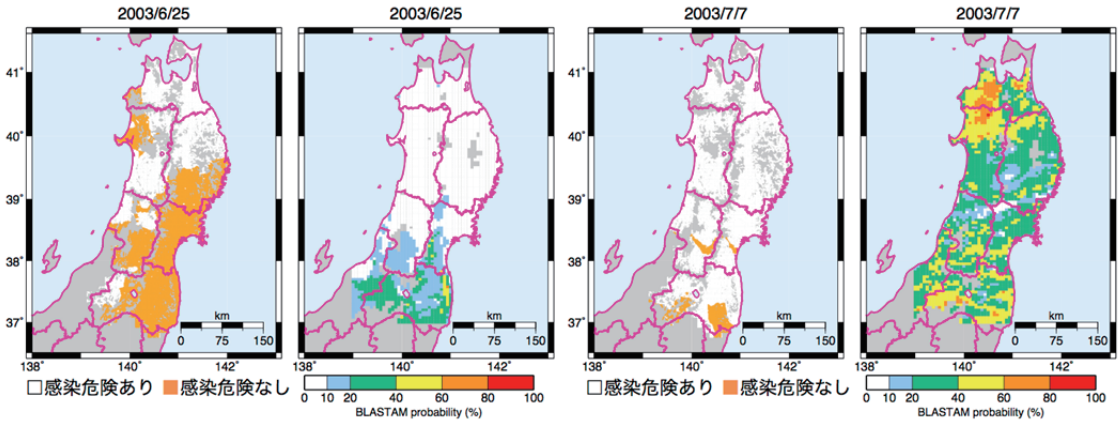


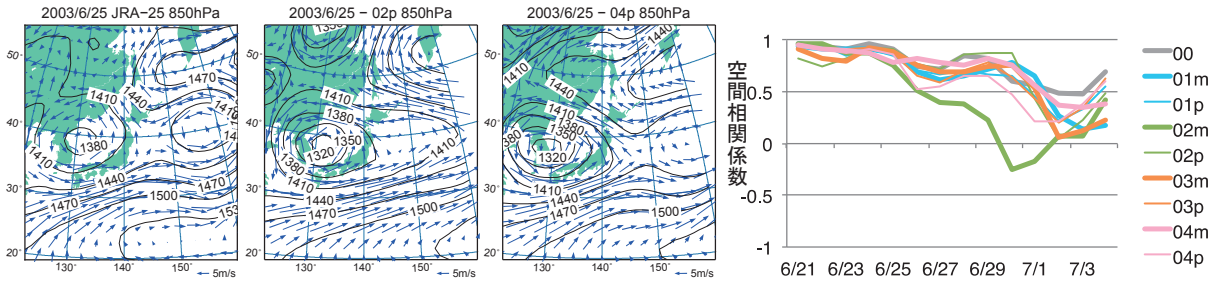
図 14　アメダス観測値により求めた感染好適条件（a）とアンサンブル予測データによる葉いもち感染確率の分布（b）

これらの原因は、気圧配置の予測精度と関係しています。5km 間隔にダウンスケールを実施する前の予測データの気圧配置を、再解析データによる実際の気圧配置と比較しました。比較にはアノマリ相関による空間相関係数（実際の気圧配置と予測された気圧配置とを比較するために、その時期の平均的な気圧配置からの両者の差で求めた相関係数）を用いて、気圧配置の類似度としました。実際と予測の気圧配置の例と、両者の空間相関係数の推移を図 15 に示します。

2003 年 6 月 20 日初期値による気圧配置の例（図 15a）では、再解析データとアンサンブル予測データとでよく似た気圧配置を示しています。両者の気圧配置の類似度を示す空間相関係数も、

ほとんどのメンバーが再解析データと 0.6 程度の相関係数を保って推移しており、期間を通じて気圧配置の予測精度が比較的よかったことを示していました。その一方で、2003 年 6 月 30 日初期値の例（図 15b）では、多くのメンバーで 5 日目以降に空間相関係数が大きく下がり、気圧配置が再現されていないことを示していました。BLASTAM の分布も、気圧配置の相関が低下した時期に観測値との一致度が下がっていました。これらのことから、予測の精度が高く、気圧配置型の再現性が高い時期には、BLASTAM の結果もアメダス観測値と近い空間分布になることがわかりました。

（a）2003 年 6 月 20 日初期値、気圧配置は 5 日目の例



（b）2003 年 6 月 30 日初期値、気圧配置は 7 日目の例

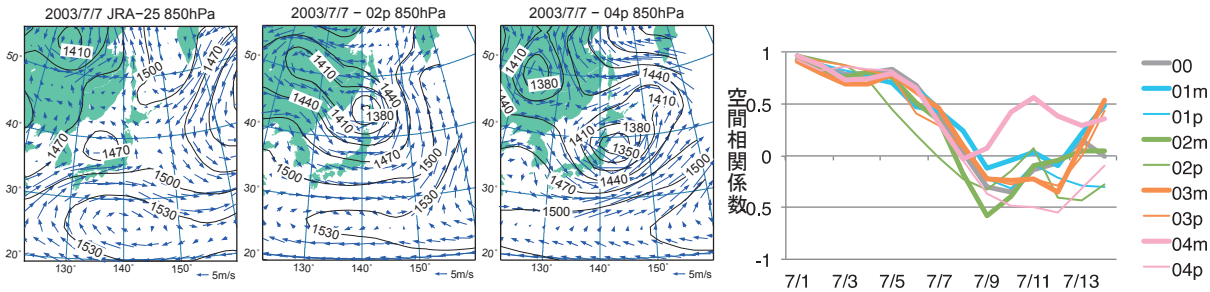


図 15　再解析データとアンサンブル予測実験による気圧配置の比較。

a, b ともに、左図が再解析データによるもの、中・右図がアンサンブルメンバー（9 メンバーより 2 つを抜粋）によるものを示す。グラフは再解析データとアンサンブル予測 9 メンバーのアノマリ相関による空間相関係数の推移を示す。

3. 2. 4 おわりに

ここで示したアンサンブルダウンスケールデータによる「葉いもち感染確率」の予測は、実験段階にあるものです。今後、アンサンブルダウンスケールデータの実利用が始まると、どのくらいの確かさであるかを示した新たな形での情報提供（例として、図 16）が、BLASTAM をはじめとする農業気象分野へも活用されることが期待されます。

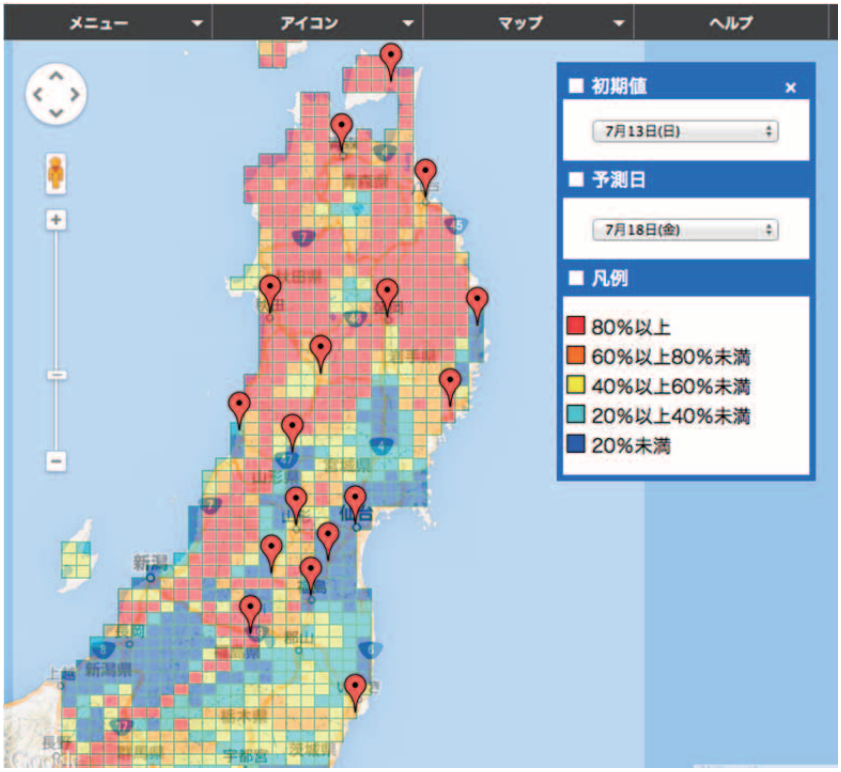


図 16　葉いもち感染危険度予測マップの web 上での表示例。
アンサンブルダウンスケール気象予測データ（27 メンバー、10km 間隔）を BLASTAM に適用し、感染危険ありを示したメンバーの割合をメッシュごとに示している。

3.3. 農業支援システム

3.3.1. はじめに：2 週間先の低温・高温リスクの高まりを把握する技術とは？

これまでに東北農業研究センターと岩手県立大学で共同研究を行い、東北地方の農業従事者を対象に、ウェブサイトを通じて 1 週間先までの気温の予測等を用いた「水稲栽培管理警戒情報」を提供してきました（2010 年より農業試験システムを一般公開しております。ウェブサイト http://map2.wat.soft.iwate-pu.ac.jp/）。

深水管理による冷害対策などには一定の準備期間を要することから、さらに早い時期から低温・高温のリスクの高まりを把握できるように、2011 年より気象庁の協力のもと、2 週間先までの予測情報による警戒情報を試行提供しています。2 週間先までの予測情報は、アンサンブルダウンスケール予報（確率予報）であり、一般にあまり馴染みのないものであるので、警戒情報を分かりやすい言葉で表現し提供しています（警戒情報はメール配信もしています）。

3.3.2. 2 週間先までの気温の予測情報を提供しています

2 週間先までの予測情報には、東北地方全体を 1km メッシュに区切って示した気温予測（7 日平均気温）と確率情報（低温、高温確率）があります（東北農業研究センターと気象庁の共同研究により作成しています）。

（1）気温予測

気温予測（7 日平均気温）は、東北農業研究センターが作成した 1km メッシュ気温平年値・標準偏差（対象期間は 1981～2010 年の 30 年）と気象庁の「異常天候早期警戒情報（以下、早警）」のガイダンス（アンサンブル予報から地域毎の気温平年差や降水量平年比を予測したデータ）を用いて作成しています（図 17）。

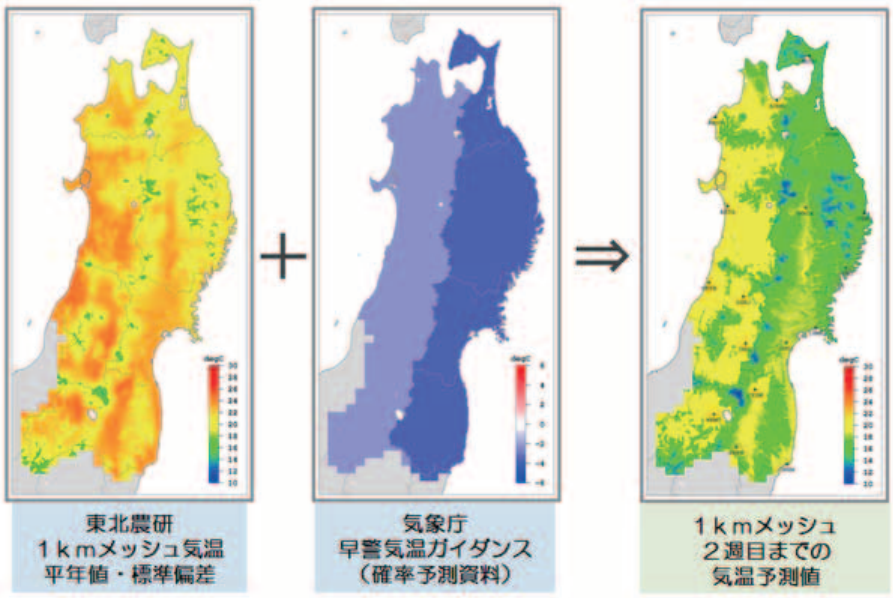


図 17　東北地方 1km メッシュ気温予測値の作成方法。

2 週間先の予測では、明日、明後日の天気予報のような精度のよい予測は困難ですが、気温の大まかな傾向などを把握することは可能です。図 18 は、低温が顕著であった、1993 年の 7 月末を対象として、モデルによる予測実験を行った結果です。右の予測結果をみると、2 週間前の時点で、左の解析値ほどではありませんが、北日本を中心とした低温が予測されています。このような情報により、いち早く低温などの兆候が分かれば、冷害対策に生かすことが期待できます。

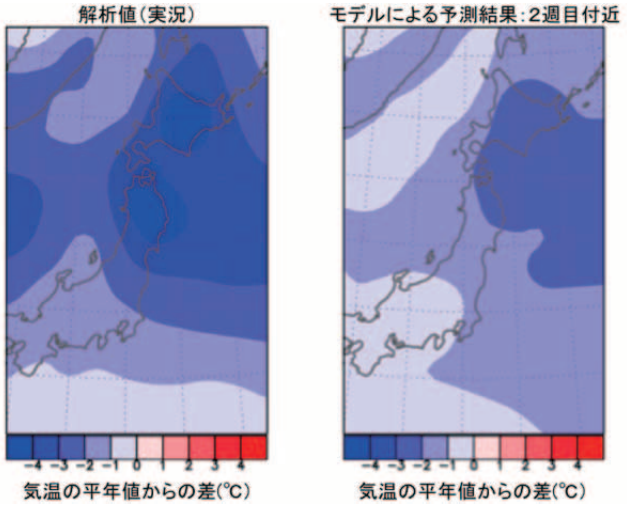


図 18　1993 年 7 月末のヤマセ時を対象とした、2 週間先の予測実験結果（7 日平均気温）。

（2）確率情報

確率情報（低温、高温確率）は、各メッシュについて、水稲の低温・高温への影響が懸念される気温の確率です。表 2 に示すように、低温の基準は 20℃以下としています。7 月中旬から 8 月上旬にかけて気温が 20℃より下がると、花粉の発育障害が起こり、受粉・受精が正常に行われないなどの理由から、収量が著しく下がる危険性が高まります。また、高温の基準は 27℃以上としています。8 月上旬から 8 月下旬にかけて気温が 27℃以上になる場合、高温登熟障害により品質が低下する危険性が高まります。

表 2　水稲の警戒気温。

時期	警戒気温（7 日間平均）	懸念される症状	対策
7 月中旬から 8 月上旬（幼穂形成期～出穂期前）	20℃以下	障害不稔発生	深水管理など
8 月上旬（出穂期）	20℃以下	障害不稔発生	深水管理など
8 月上旬から 8 月下旬（出穂期～登熟初期）	27℃以上	高温登熟障害	掛け流し灌漑など

図 19 は、ヤマセによる低温が顕著であった、2003 年の 7 月末を対象として予測実験を行った結果です。左の平年値と比較すると、低温確率が大きく、低温のリスクが平年より高まっていることがわかります。右の実況値はこの時期に対応した実際の気温の分布です。水稲の低温障害が懸念される 20℃以下の所を、青系の色で示しています。この事例では、中央の確率メッシュの 20%以上に着目すると、実況で 20℃以下の地域がかなり捉えられ、10%以上に着目すればほぼ網羅されます。このように平年の確率と比べて低温（高温）確率が大きい場合には、10%や 20%のような小さい確率でも注意が必要な場合がありますので、この先の状況に注目しつつ、いざという時に深水管理などの対策ができるように準備するのがよいと考えられます。

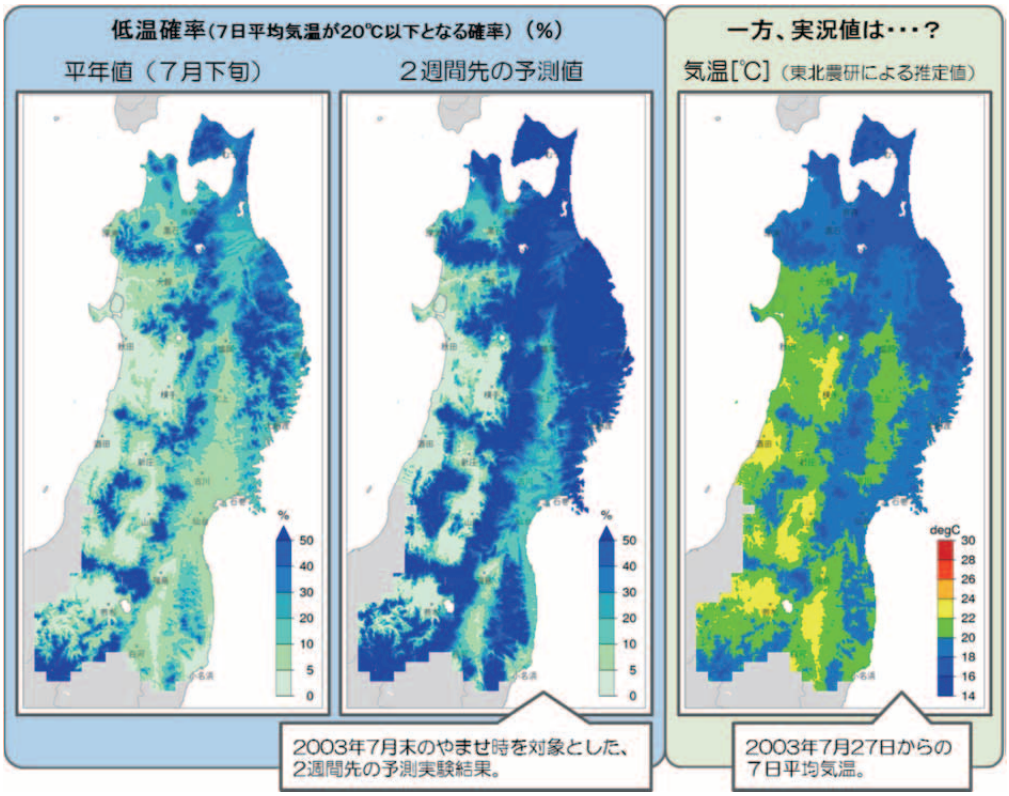


図 19　1993 年 7 月末のヤマセ時を対象とした、2 週間先の予測実験結果（警戒確率）。

3.3.3. 低温リスクと高温リスクの警戒情報を配信しています

2 週間先までの予測情報による警戒情報は、2 週間先までの予測情報の特性を考慮して、1 週間先までの警戒情報（栽培管理）の補助的な情報と 2 週間先までの警戒情報（低温、高温確率）のみの単独の情報の 2 つを提供しています。

（1）1 週間先までの警戒情報（栽培管理）の補助的な情報

1 週間先までの予測情報を用いた栽培管理の警戒情報では、生育モデルからユーザの圃場での生育段階を予測し、各生育段階での警戒温度をもとに低温、高温のリスクがある場合に警戒情報を提供しています。1 週間先までに低温、高温のリスクがある場合には、さらに 2 週間先までの

予測情報から低温、高温のリスクがあるかを警戒情報として提供します。すなわち、低温、高温のリスクが 2 週間先まで継続する場合に注意を促し、早目の対策を立てる判断材料として活用できるようにしています。

図 20 に示すように、低温（20℃以下）、高温（27℃以上）の警戒情報は、低温、高温確率が 30%以上の場合に提供します（確率と実況出現頻度から基準を定めています）。ただし、高温の警戒情報においては、8 月下旬から 9 月には 27℃以上の確率が 30%となることがほとんどないため、刈り取り適期等に活用することを考え、アンサンブル平均の平年差が+1.5℃以上となる場合に警戒情報を提供しています（気象庁から高温の異常天候早期警戒情報が発表されるレベルに近い条件で警戒情報を提供しています）。

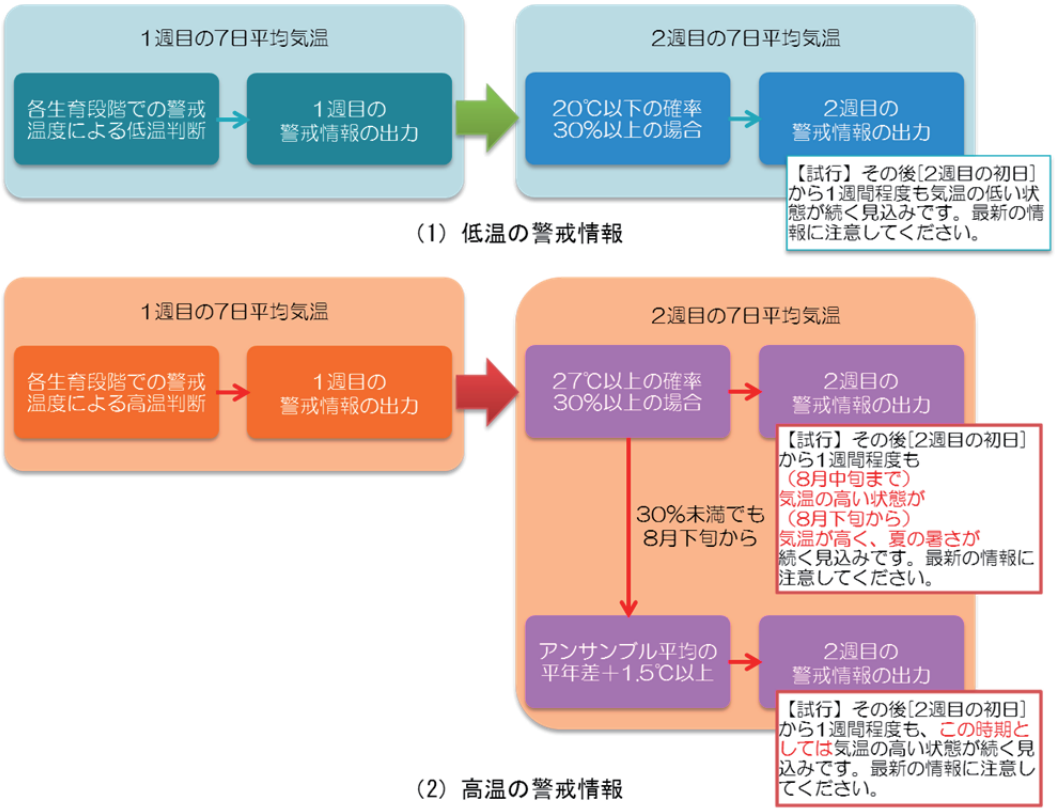


図 20　1 週間先までの警戒情報（栽培管理）の補助的な情報。

（2）2 週間先までの警戒情報（低温、高温確率）のみの単独の情報

2 週間先までの警戒情報のみの単独の情報では、低温、高温確率を分かりやすい言葉で表現し提供します（表 1 の低温、高温障害の危険期に警戒情報をメールで配信しています）。

図 21 に示すように、低温（20℃以下）、高温（27℃以上）の警戒情報は、低温、高温確率が 20%以上の場合で、なおかつ、低温、高温確率が平年の出現確率より大きい場合に提供します（確率と実況出現頻度から基準を定めています）。20%の小さい確率でも平年の確率と比べて低温、高温確率が大きい場合に注意を促し、早目の対策の判断材料として活用できるようにしています。

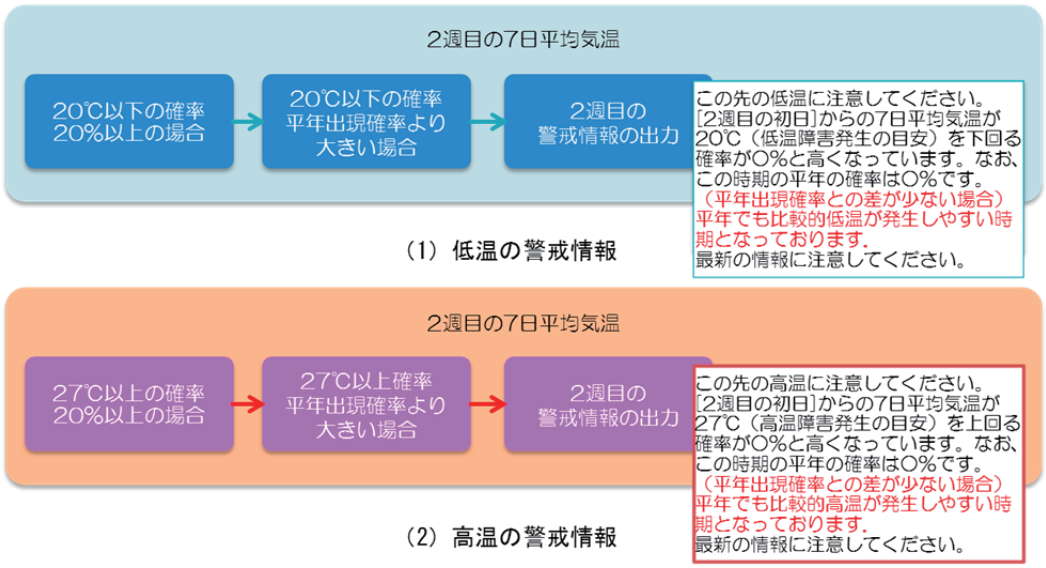


図 21　2 週間先までの警戒情報（低温、高温確率）のみの単独の情報。

3.3.4. まとめ

本節で紹介しました農業支援システム（図 22）は，ユーザ登録をすれば，どなたでも無料で利用できます（ウェブサイト http://map2.wat.soft.iwate-pu.ac.jp/ からユーザ登録ができます)。早い時期から低温・高温のリスクの高まりを把握できるように、ウェブサイトおよびメールにより情報提供を行っていますので、深水管理などの対策を実施する際の判断材料の 1 つとしてご利用して頂ければと思います。

図 22　農業支援システム（ウェブサイト http://map2.wat.soft.iwate-pu.ac.jp/)。

4. 終わりに

本稿は、大学、農業研究機関、気象機関が協力し、東北の農業（稲作）の地球温暖化適応策を取りまとめたものです。地球温暖化予測実験結果をダウンスケールし、地域の温暖化予測データを作成し、適応策を検討しました。内容は、気候の平均的状態の変化への対応と、気候変動に対処するための農業気象情報の高度利用の 2 点です。

1 節でも書いた通り、地球温暖化予測には、様々な不確実性が残されています。また、温暖化による影響は地域によって異なり、地域の温暖化予測は一層難しく不確実性も大きい、と考えなければなりません。温暖化予測の精密化をはかり、適応策も随時修正していく必要があります。

地球温暖化が地域社会に与える影響はたいへん複雑です。このため、温暖化適応策も多岐にわたります。地域の問題は地域で考えていくことが大切で、そのためには、地域における横の連携が大事です。有効な適応策を提案するためには、産官学や学際、異業種間での議論を一層深める必要があります。

地球温暖化予測情報を地域の発展に役立てるためには、そもそも、日々の気象予測を有効活用できる基盤が必要です。気象予測技術の発展は顕著であり、予報を活用できる範囲は確実に広がります。確率表現が可能となるアンサンブル予報も徐々に公開が始まりつつあります。気象データはビッグデータの時代を迎えます。膨大な気象予測データから地域に役立つ情報を引き出すための高度利用技術がますます重要になります。

謝辞

本稿は、気候変動適応研究推進プログラム「ヤマセと冬季モンスーンに関する先進的ダウンスケール研究」の研究成果と様々な議論に基づいています。また、全球気候モデルによる温暖化データのダウンスケールには、東北大学サイバーサイエンスセンターのスーパーコンピュータを利用しました。同センター関係各位の有益なご指導とご協力に感謝いたします。本研究で使用した気象庁データ「週間予報データ」は気象庁と（社）日本気象学会の研究協力の枠組みである「気象研究コンソーシアム」を通じて提供されました。

用語解説

- 全球気候モデル

全球気候モデルは、大気・海洋・陸面・雪氷などの変化を、流体力学・力学・化学・物理学・生物学などの方程式を用いて表現し、地球規模での気候をシミュレーションする数理モデルの総称。地球温暖化の予測や原因の推定などに用いられ、気候変動に関する政府間パネル（IPCC）の報告書の根拠となる研究成果にも数多く利用されています。

- 領域気象モデル

領域気象モデルは、大気・海洋・陸面・雪氷などの変化を、流体力学・力学・化学・物理学・生物学などの方程式を用いて表現し、着目する領域内の気象を詳細にシミュレーションする数理モデルの総称。防災気象情報等を提供するために用いられています。

- シナリオ

将来の気候を予測するために用いるための社会の変化の想定のこと。特に、温室効果ガスの排出量についてのシナリオを、排出シナリオと言います。排出シナリオの中で、特に IPCC の「排出シナリオに関する特別報告 (Special Report on Emission Scenarios)」が提案したものを SRES シナリオと言います。気象庁の「地球温暖化予測情報第 8 巻」に用いられている SRES シナリオは、その中の一つ SRES A1B シナリオです。このシナリオは、高度経済成長が続き、地域間格差が縮小し、新しい技術が急速に導入される未来社会で、全てのエネルギー源のバランスを重視するという、社会変化を想定したものです。このシナリオの他に、二酸化炭素の排出を減少させていく社会を想定したシナリオ、あるいは、二酸化炭素排出量が増えていく社会を想定したシナリオも存在します。

- 力学的ダウンスケーリング

数値気象モデルを用いた再計算によるデータの詳細化のこと。低解像度の全球気候モデルによる大規模場の計算結果をもとに、ある領域について、高解像度の地形等を入力して再計算することを言います。これにより、指定した領域について、下部境界の状態に大きく依存した小さな現象を解像することができます。温暖化予測については、全球で計算された気候の変化が、着目する地域ではどのように現れるかがわかるようになります。本研究では、東北大学サイバーサイエンスセンターのスーパーコンピュータを利用して、計算を実施した。

参考文献

[1] 気象庁 地球温暖化予測情報 第 8 巻, 2013.
http://www.data.jma.go.jp/cpdinfo/GWP/index.html
[2] 下野裕之, 2012: 地球温暖化でも冷害はなくならない―そのメカニズムと対策, 農山漁村文化協会.
[3] 卜蔵建治, 2001: ヤマセと冷害: 東北稲作のあゆみ, 成山堂書店.
[4] 松村伸二, 2011: 2010 年夏季の異常高温と農業被害 -水稲を中心として-, 自然災害科学, 30(2), 169-192.
[5] 気象庁 気象庁気候変動監視レポート 2013, 2014.
http://www.data.jma.go.jp/cpdinfo/monitor/index.html
[6] 林陽生, 2009: 地球温暖化で日本農業はどう変わる, 家の光協会.
[7] 渡邉紹裕, 2008: 地球温暖化と農業–地域の食料生産はどうなるのか？, 昭和堂.
[8] 越水幸男, 1988: アメダス資料による葉いもち発生予察法. 東北農業試験場研究報告, (78), 67-121.
[9] 林孝・越水幸男, 1988: 葉いもち発生予察のコンピュータプログラム（BLASTAM）の開発. 東北農業試験場研究報告, (78), 123-138.

［式典報告］

東北大学サイバーサイエンスセンター新棟竣工および新スーパーコンピュータシステム導入披露式典を開催しました

平成 27 年 2 月 20 日（金）、東北大学サイバーサイエンスセンター（以下、本センター）において、新棟竣工および新スーパーコンピュータシステム導入披露式典を開催しました。これは、新スーパーコンピュータシステム導入に合わせて建設が進められていた新棟の竣工および新スーパーコンピュータシステムを正式に運用を開始したことを記念して開催されたものです。

本センターは、これまで、学術界および産業界で年々大規模・高精度化するシミュレーション解析に対応するため、最先端コンピュータシステムを整備し、優れた利用環境の開発に努力してまいりました。この度、更なる処理能力の要求に応えるべく、これまで運用していた SX-9 の 20 倍以上の性能を有する新型ベクトル型スーパーコンピュータ「SX-ACE」（NEC 製）を導入し、平成 27 年 2 月 20 日から全国の研究者・技術者の皆さまへの提供を開始しました。

サイバーサイエンスセンター新棟（左）

新スーパーコンピュータシステム

この新スーパーコンピュータシステムは、我が国をはじめ、米国、中国などのフラグシップスーパーコンピュータの 3 倍～10 倍となる世界最高の性能効率を、極めて省電力で達成できることが明らかになっています。

また、本センターは被災地にある拠点大学として防災・減災の取り組みにも力を入れており、災害科学国際研究所等と共同し、この新システムを利用することにより、大規模地震に伴って発生する津波浸水被害をリアルタイムで評価・分析する「リアルタイム津波浸水予測システム」の研究開発にも取り組んでいます。

式典では、榎本剛文部科学省研究振興局参事官（情報担当）をはじめ、里見進総長、各大学の基盤センターや関連する機関、NEC の来賓の方々ら約 100 名にご出席いただき、式典後には新棟に設置された新スーパーコンピュータシステムを見学いただきました。

続いて開催された記念講演会では、宇宙航空研究開発機構教授・一般社団法人 HPCI コンソーシアム理事長の藤井孝藏氏、ドイツ シュトゥットガルト大学高性能計算センター長の Michael M. Resch 氏、および NEC 執行役員常務の庄司信一氏より、それぞれご講演いただきました。

引き続き、講演会の後に開催された祝賀会では、青木孝文副学長のご発声で乾杯を行いました。終始和やかな雰囲気の中で、学内外からスピーチをいただき、盛況のうちに閉会しました。

里見進総長の挨拶

見学会の様子

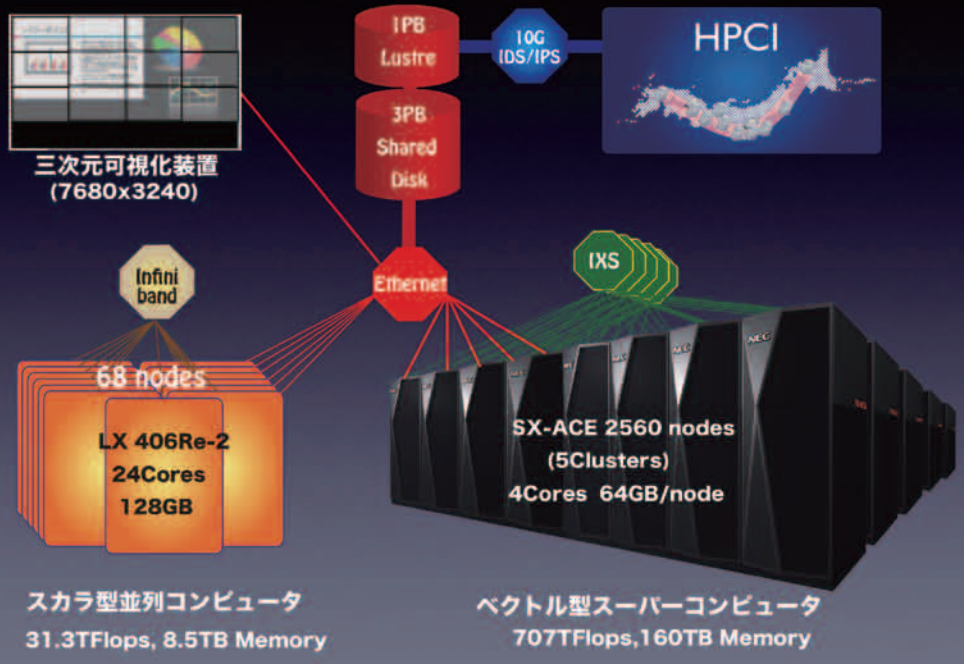
［式典報告］


東北大学
Cyberscience Center
東北大学サイバーサイエンスセンター
新スーパーコンピュータの紹介と
高性能計算に関する研究開発活動
小林広明
センター長・教授
東北大学サイバーサイエンスセンター
koba@cc.tohoku.ac.jp
2015年2月20日


TOHOKU
Hiroaki Kobayashi, Tohoku University
東北大学サイバーサイエンスセンターの歩み
★ 1969年に日本で2番目の全国共同利用型大型計算機センターとして設立
• 汎用大型計算機の運用
★ 1985年よりスーパーコンピュータセンターとして活動
• 大規模科学計算システムとしてベクトル型のスーパーコンピュータを運用
★ 2008年に情報シナジーセンターからサイバーサイエンスセンターへ改組
• 2010年より「学際大規模情報基盤共同利用・共同研究拠点」として文科省より認定を受け，2012年よりHPCIに資源提供を開始し，HPCに関する共同利用・共同研究活動を一層強化
新システム設計
運用/利用支援/教育
人材育成
研究開発
SENAC-1 in 1958
First Computer in 1969
ACOS 1000 in 1982
SX-1 in 1985
SX-2 in 1989
SX-3 in 1994
SX-4 in 1998
SX-7 in 2003
SX-9 in 2008
新スーパーコンピュータ導入披露式典
2
平成27年2月20日

Hiroaki Kobayashi, Tohoku University

新スーパーコンピュータシステムの構成



Hiroaki Kobayashi, Tohoku University

新システムの特徴

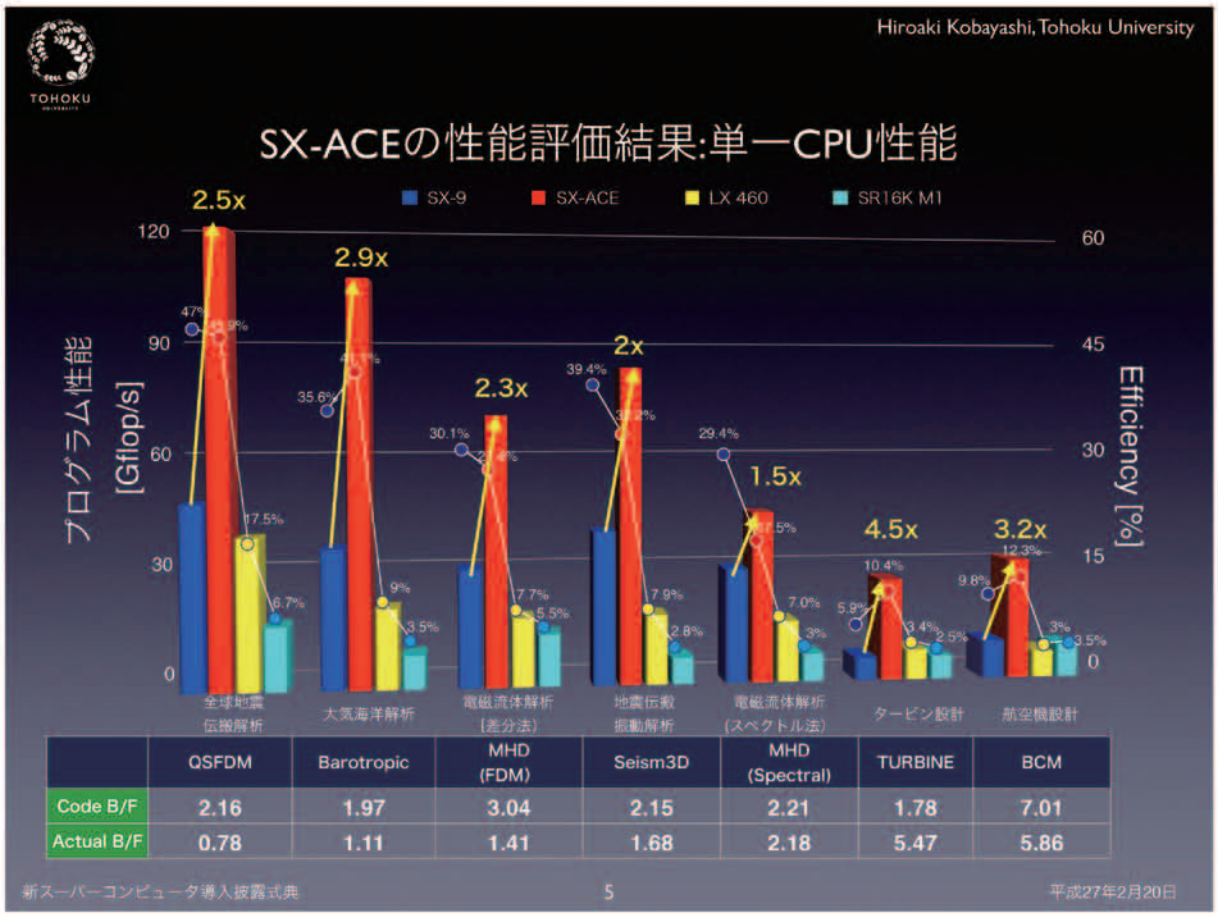
大幅な性能強化を省スペース・省電力で実現

		SX-9 (2008)	SX-ACE (2014)	性能向上比
	コア数	1	4	4x
CPU性能	理論最大演算性能	118.4Gflop/s	276Gflop/s	2.3x
	メモリバンド幅	256GB/sec	256GB/sec	1
	ADB容量	256KB	4MB	16x
システム性能・規模・消費電力	理論最大演算性能	34.1Tfop/s	706.6Tflop/s	20.7x
	総メモリバンド幅	73.7TB/s	655TB/s	8.9x
	総メモリ容量	18TB	160TB	8.9x
	最大消費電力	590kVA	1,080kVA	1.8x
	計算機室床面積	293平米	430平米	1.5x

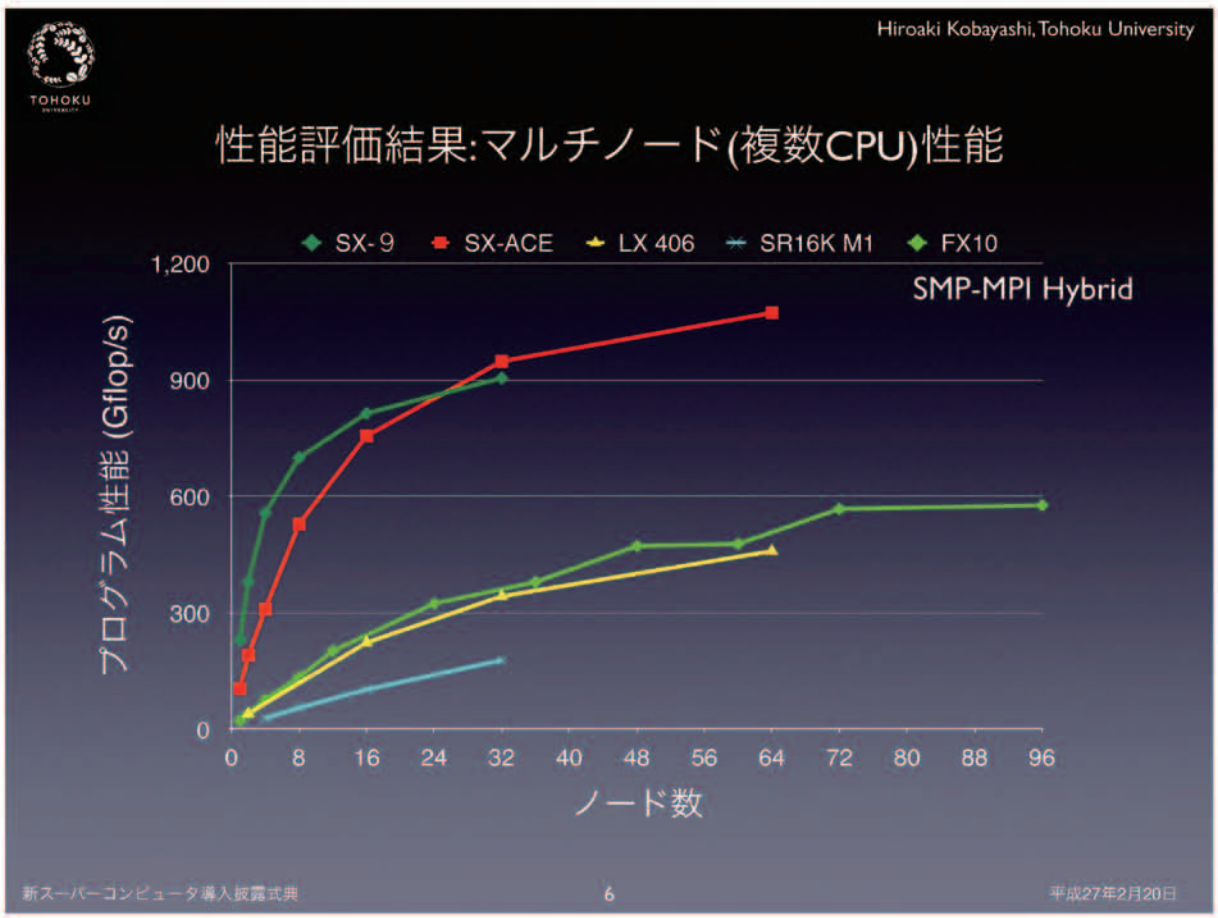
量より質へのこだわり

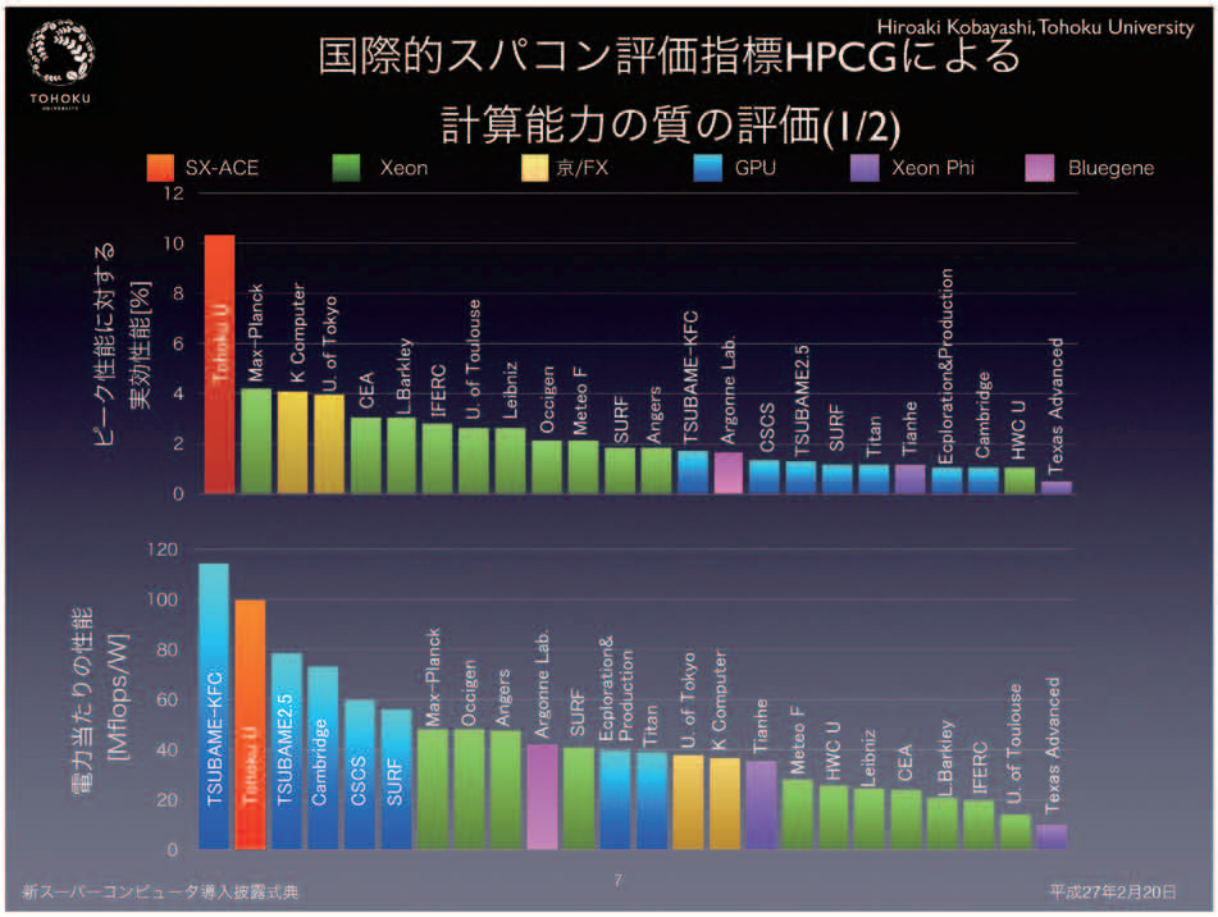
		SX-ACE(2014)	京(2011)	性能比
CPU (ノード) 性能	クロック周波数	1GHz	2GHz	0.5x
	コア性能	64Gflop/s	16Gflop/s	4x
	コア数/CPU	4	8	0.5x
	積和演算性能	256Gflop/s	128Gflop/s	2x
	メモリバンド幅	256GB/s	64GB/s	4x
	メモリ容量	64GB	16GB	4x

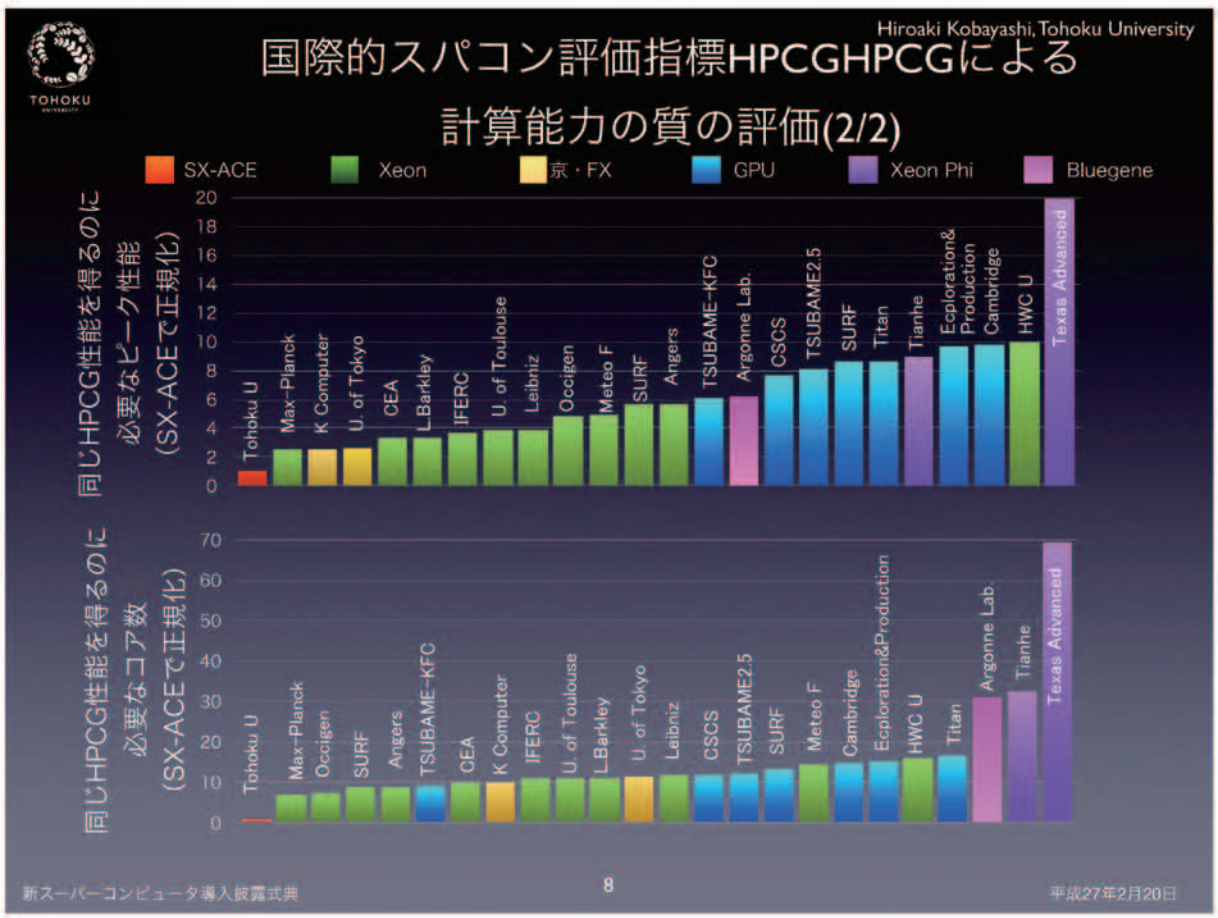
計算科学の多様な分野において萌芽研究から実用研究に対し
生産性の高いシミュレーション実行環境の構築をめざし，全国の利用者に提供



	QSFDM	Barotropic	MHD (FDM)	Seism3D	MHD (Spectral)	TURBINE	BCM
Code B/F	2.16	1.97	3.04	2.15	2.21	1.78	7.01
Actual B/F	0.78	1.11	1.41	1.68	2.18	5.47	5.86

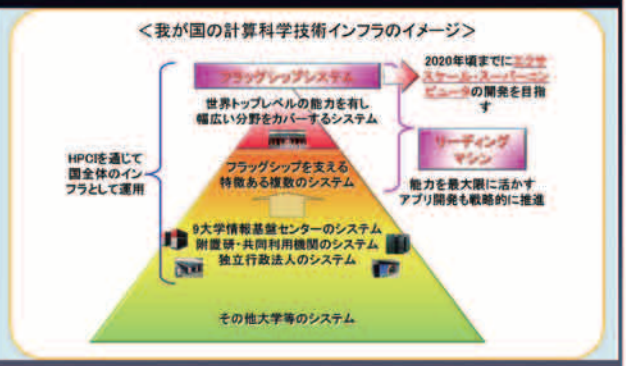



Hiroaki Kobayashi, Tohoku University
国際的スパコン評価指標HPCGによる
計算能力の質の評価(1/2)
SX-ACE
Xeon
京/FX
GPU
Xeon Phi
Bluegene
ピーク性能に対する実効性能[%]
Tohoku U
Max-Planck
K Computer
U. of Tokyo
CEA
L.Barkley
IFERC
U. of Toulouse
Leibniz
Occigen
Meteo F
SURF
Angers
TSUBAME-KFC
Argonne Lab.
CSCS
TSUBAME2.5
SURF
Titan
Tianhe
Ecploration&Production
Cambridge
HWC U
Texas Advanced
電力当たりの性能 [Mflops/W]
TSUBAME-KFC
Tohoku U
TSUBAME2.5
Cambridge
CSCS
SURF
Max-Planck
Occigen
Angers
Argonne Lab.
SURF
Ecploration&Production
Titan
U. of Tokyo
K Computer
Tianhe
Meteo F
HWC U
Leibniz
CEA
L.Barkley
IFERC
U. of Toulouse
Texas Advanced
新スーパーコンピュータ導入披露式典
7
平成27年2月20日


Hiroaki Kobayashi, Tohoku University
国際的スパコン評価指標HPCGHPCGによる
計算能力の質の評価(2/2)
SX-ACE
Xeon
京・FX
GPU
Xeon Phi
Bluegene
同じHPCG性能を得るのに必要なピーク性能 (SX-ACEで正規化)
Tohoku U
Max-Planck
K Computer
U. of Tokyo
CEA
L.Barkley
IFERC
U. of Toulouse
Leibniz
Occigen
Meteo F
SURF
Angers
TSUBAME-KFC
Argonne Lab.
CSCS
TSUBAME2.5
SURF
Titan
Tianhe
Ecploration&Production
Cambridge
HWC U
Texas Advanced
同じHPCG性能を得るのに必要なコア数 (SX-ACEで正規化)
Tohoku U
Max-Planck
Occigen
SURF
Angers
TSUBAME-KFC
CEA
K Computer
IFERC
U. of Toulouse
L.Barkley
U. of Tokyo
Leibniz
CSCS
TSUBAME2.5
SURF
Meteo F
Cambridge
Ecploration&Production
HWC U
Titan
Argonne Lab.
Tianhe
Texas Advanced
新スーパーコンピュータ導入披露式典
8
平成27年2月20日

Hiroaki Kobayashi, Tohoku University

東北大学サイバーサイエンスセンターの目指す役割



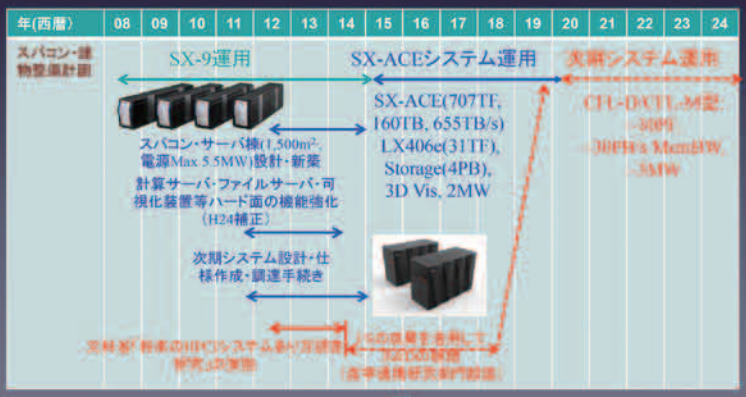
現在のHPCIシステムの安定運用・高度化への貢献　　将来のHPCIシステム研究開発・構築への貢献

これらの取り組みを通じて，計算機科学・計算科学の両分野に精通した分野融合・横断型の人材育成

Hiroaki Kobayashi, Tohoku University

東北大学サイバーサイエンスセンターの今後のスーパーコンピュータ整備・運用・研究開発計画
～次の10年間の活動計画～

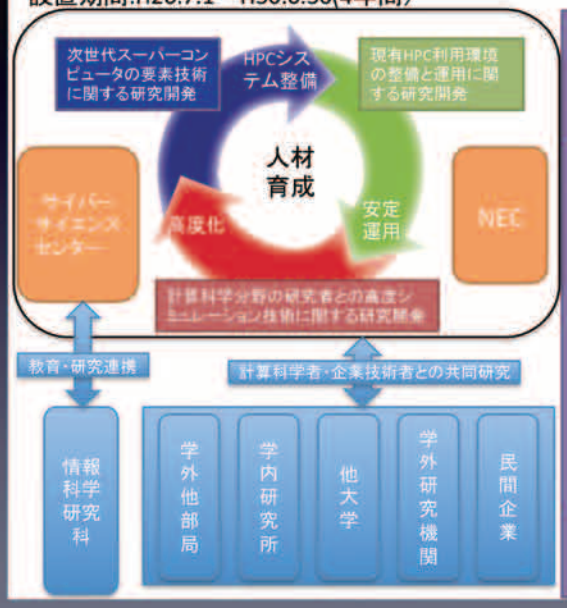
- 「普通の人々のためのスーパーコンピュータセンター（The Supercomputer Center for the Rest of Us!）」を目指して
- ★ 第2階層に位置するNISとして,フラグシップマシンを補完すべく，演算性能とバランスしたメモリ性能を有するマシンの研究・開発に取組み，高い生産性（Short time to Solution)を提供できるシステムの整備・運用を目指す
  - ✓ システム運用と開発を両輪に，ユーザ支援・共同研究で得た知見を，次のシステムの設計・開発に生かす
- ★ 教員・技術系職員・ベンダー技術者が一体となった現ユーザ支援体制のさらなる強化
- ★ 社会貢献として産業利用支援の強化
- ★ 学部・研究科（リーディング大学院等）と連携したHPC教育プログラムへの参画
- ★ 全国共同利用型の学内スパコン連携体制の構築,およびHPCI基盤を活用した学外スパコン機関との連携のさらなる推進



Hiroaki Kobayashi, Tohoku University

HPCに関する研究・人材育成機能の強化をめざして: 高性能計算技術開発(NEC)共同研究部門の新設

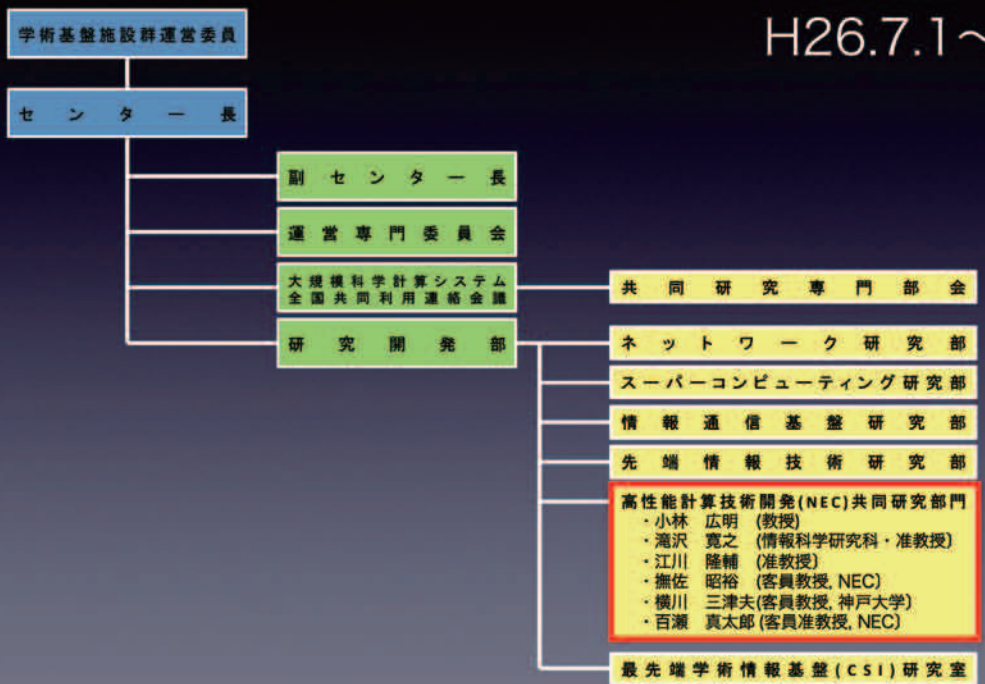
設置期間:H26.7.1～H30.6.30(4年間)



- 我が国のHPC基盤の強化，およびリーディングマシンの研究開発への貢献
  - HPCIシステムの高度化に関する研究開発の実施
  - ポスト京及びそれを補完するシステムが位置付けられるリーディングマシンの研究開発への貢献
- 学内外の計算科学者との共同研究の受け皿として，シミュレーション科学の高度化に貢献
  - G空間を活用したリアルタイム津波シミュレーション(災害科学国際研究所との共同研究)
  - 防災・減災に資するスーパーコンピュータシステムの研究開発(JAMSTEC・東北大組織的連携協定，電気通信研究機構との共同研究，その他)
- 部門教員が担当する情報科学研究科の基幹・協力講座において，計算科学・計算機科学の両分野に精通した学際的，かつ実践的人材育成
  - NECおよびその他外部協力機関から客員教員の招聘し，研究・教育に従事
  - H26年度情報科学研究科プロジェクト「ビッグデータ解析利活用ソリューションラボの構築」への参画
- HPCを通じて産学連携研究拠点の構築
  - NECとの次期スパコンに関する共同研究
  - これまでに，MRJに対する航空機開発支援を含め7社へスパコン提供・利用支援を実施

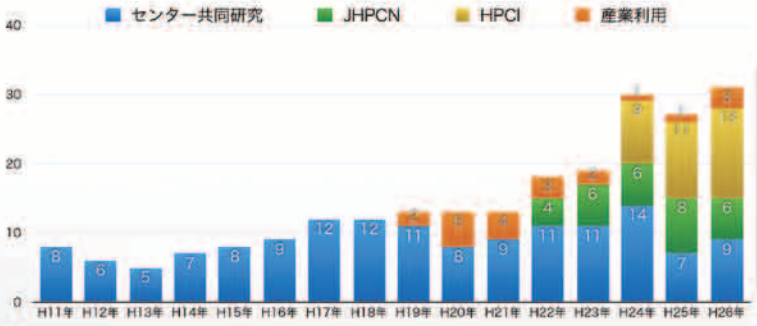
Hiroaki Kobayashi, Tohoku University

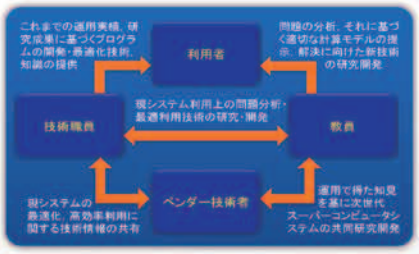
サイバーサイエンスセンター新組織図 H26.7.1～



利用者支援機能の強化：プログラムの高速化に関する取り組み

□ 平成9年に整備したセンター職員・教員・ベンダエンジニアによる高速化支援体制を共同研究部門へ引き継ぎ，ユーザ支援をより一層強化





これまで約200件の共同研究を推進，高速化支援対象となるコード数も増加

年	H11	H12	H13	H14	H15	H16	H17	H18	H19	H20	H21	H22	H23	H24	H25
件数	8	9	10	7	18	20	8	16	10	15	8	8	13	6	11
単体性能向上比	4.5	2.5	1.6	2.2	6.7	2.9	1.5	2.9	33	9.3	381	47	16.2	19.7	16.7
並列性能向上比	31.7	8.6	4.9	2.8	18.6	4.5	4.1	8.1	1.9	5.1	3.6	48	17.2	15.3	12.9

Hiroaki Kobayashi, Tohoku University

社会に役立つスーパーコンピュータ技術の創出：リアルタイム津波浸水・被害予測システムの研究開発

地震発生後20分以内に
★地震断層モデルに基づく津波発生予測（10分）
★SX-ACEによる10mメッシュの津波浸水予測および被害推計（10分）

世界初

●10メートルメッシュの精度で、6時間分の浸水域・建物被害予測情報を自治体に提供
●将来はスマホなどへlocation-awareな災害情報配信技術の確立を目指す

Hiroaki Kobayashi, Tohoku University

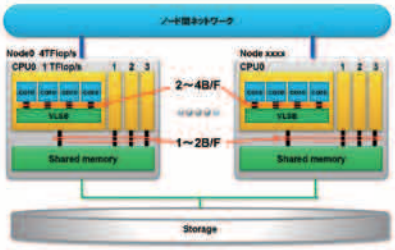
次世代HPCの要素技術の研究開発：高メモリバンド幅アプリケーションのための将来のHPCIシステムのあり方調査研究の継承・発展

■ 文科省「将来のHPCIシステムのあり方の調査研究事業」(H24～25年度)の補助を受けて実施(東北大，東大，筑波の3機関からの3提案が採択)
- ◆2020年頃をターゲットに，解決が求められる社会的・科学的課題を精査し，そのためのアプリケーションの検討・評価を行い，併せて，ターゲットとする重要アプリ,特に，防災・減災,ものづくりアプリケーションのためのスーパーコンピュータシステムに求められる機能・性能的要件を明確にし，システム実現のための要素技術の調査研究を行っている．

■ 参加研究者
- ◆東北大：小林広明(研究代表者)，今村文彦，越村俊一，小柳光正他21名
- ◆JAMSTEC:渡邉國彦，金田義行他18名
- ◆その他，協力者としてNEC，東大，港湾研，JAXA,理研等44名

■ 成果発表イベント企画
- ◆国際シンポジウムWSSP(workshop on Sustained Simulation Performance)等を通じて成果を国内外に発信

■ 研究打ち合わせ全体会議を6回実施し，80ページを超える成果報告書を文科省に提出．
- ◆同時期に事業を実施していた東大・筑波チームが達成できない性能・機能を有するとの評価結果



Hiroaki Kobayashi , Tohoku Univ.

さいごに

サイバーサイエンスセンターは、これからも多様な高性能計算にニーズに応え，シミュレーションの生産性を高めるために

★メモリ性能と演算性能がバランスした最先端のスーパーコンピュータシステムの導入・提供

★使い易い利用環境の整備・提供、利用者支援

★次世代スーパーコンピュータ技術に関する研究開発

に取り組み

★普通の人々に役立つスーパーコンピュータセンター
Supercomputer Center for the Rest of US!

をめざします

引き続きご支援ご指導ご協力をお願いいたします!

［報 告］

第 21 回高性能シミュレーションに関するワークショップ(WSSP)を開催しました

東北大学サイバーサイエンスセンター　小林広明

東北大学サイバーサイエンスセンターは、東北大学、海洋研究開発機構（JAMSTEC)、ドイツのシュトゥットガルト大学高性能計算センター(HLRS)、NEC の共同主催で、2015 年 2 月 18 日（水）～2 月 19 日(木)に第 21 回 Workshop on Sustained Simulation Performance (WSSP)を開催しました。本ワークショップは、サイバーサイエンスセンターと HLRS との間の高性能計算に関する組織的連携協定に基づき、両センターのスーパーコンピュータシステムの利用者、並びに国際的に活躍する計算科学者・計算機科学者を招いて、毎年、春と秋にシュトゥットガルト大学と東北大学で交互に開催しております。今回のワークショップは 2 日間で延べ 159 名の参加者が集い、活発な議論がなされました。

ワークショップは、本学進藤秀夫理事（産学連携担当）の挨拶で始まり、引き続き、文部科学省研究振興局計算科学技術推進室川口悦生室長から我が国の HPCI 政策に関するご講演がありました。ご講演では、京の次のスーパーコンピュータ開発プロジェクト「FLAGSHIP2020」についての現状と今後の計画が示されました。

今回は、2 件の基調講演をスタンフォード大学マイケルフリン教授と神戸大学小柳義夫教授にしていただきました。フリン教授は、30 年以上継続されている東北大学とスタンフォード大学の研究者による高性能マイクロプロセッサに関する国際共同研究に触れられたあと、今後ますます性能が引き出すことが難しくなる高性能計算分野において、アプリケーションに応じて動的にハードウェア構成を変えることができる FPGA の可能性について将来展望を述べられました。また、小柳教授は、過去 50 年のスーパーコンピュータの開発史を紹介されたあと、世界各国が目指すエクサコンピューティングに向けたスーパーコンピュータの開発にはアプリケーションとシステムのコデザインが重要であることを述べられました。

本ワークショップでは、東北大学と JAMSTEC の間で行われている共同研究課題「地球科学, 防災・減災のための HPC システムの開発研究」の成果発表も行われました。東北大学からは、この度サイバーサイエンスセンターで新規導入したスーパーコンピュータ SX-ACE を活用したリアルタイム津波震災予測システムの研究開発に関する成果発表を行いました。また、JAMSTEC からは 3 月に導入が計画されている ES3（地球シミュレーター3）の構成とその地球科学分野での応用の発表がありました。

このほか、一般講演では、国内外の研究者から、スーパーコンピュータアーキテクチャから応用まで、計算機科学分野と計算科学分野を幅広くカバーする興味深い講演がありました。ご講演についてご興味がございましたら、講演予稿集に残部がございますので、サイバーサイエンスセンターまでお問い合わせください。

フリン教授（スタンフォード大学）による基調講演

ワークショップの様子

［報 告］

第13回 情報シナジー研究会報告

東北大学サイバーサイエンスセンターでは、先端的情報技術の研究発表と情報交換を目的として情報シナジー研究会を企画・開催しております。第13回研究会が以下のとおり開催されました。

日時： 平成27年3月2日（月） 10:00ー12:30
会場： サイバーサイエンスセンター・本館5階・講義室

プログラム：

開会

発表

10:00ー10:30
スーパーコンピュータSX-ACEと活用事例の紹介
山下 毅 (1)，大泉 健治 (1)，齋藤 敦子 (1)，小野 敏 (1)，撫佐 昭裕 (2)(3)，江川 隆輔 (2)，小林 広明 (2)
(1) 東北大学情報部情報基盤課，(2) 東北大学サイバーサイエンスセンター，(3) NEC

10:30ー11:00
Card-Based Cryptographic Protocols for Two-Bit Output Functions
シャリファ アルジュニド (1)，西田 拓也 (2)，林 優一 (2)，水木 敬明 (3)，曽根 秀昭 (3)
(1) 東北大学工学部，(2) 東北大学大学院情報科学研究科，(3) 東北大学サイバーサイエンスセンター

11:00ー11:30
時系列の画像・深度情報を用いた複数の物体の領域抽出
菅原 勝也 (1)，阿部 亨 (2)，菅沼 拓夫 (2)
(1) 東北大学大学院情報科学研究科，(2) 東北大学サイバーサイエンスセンター

11:30ー12:00
心電図計測不要の補助人工心臓の心拍動同期制御
廣橋 義寛 (1)，田中 明 (2)，吉澤 誠 (3)，杉田 典大 (1)，阿部 誠 (1)，白石 泰之 (4)，三浦 英和 (4)，山家 智之 (4)
(1) 東北大学大学院工学研究科，(2) 福島大学共生システム理工学類，(3) 東北大学サイバーサイエンスセンター，(4) 東北大学加齢医学研究所

12:00ー12:30
Refactoring of HPC Applications with User Knowledge
Chunyan Wang (1)(2)，Shoichi Hirasawa (1)(2)，Hiroyuki Takizawa (1)(2)，Hiroaki Kobayashi (3)
(1) 東北大学情報科学研究科，(2) JST CREST，(3) 東北大学サイバーサイエンスセンター

閉会

［報 告］

海洋研究開発機構地震津波・防災研究プロジェクトの有吉技術研究員が平成 26 年度地震学会論文賞を受賞しました

受賞論文名：Migration process of very low-frequency events based on a chain-reaction model and its application to the detection of preseismic slip for megathrust earthquakes

受賞者：有吉 慶介*1・堀 高峰*1・金田 義行*1・中田 令子*1・Jean-Paul Ampuero*2・
松澤 暢*3・日野 亮太*3・長谷川 昭*3
*1 独立行政法人海洋研究開発機構 地震津波・防災研究プロジェクト（現：地震津波海域観測研究開発センター）
*2 カリフォルニア工科大学
*3 東北大学大学院理学研究科附属地震・噴火予知研究観測センター

本研究では，研究対象を沈み込みプレート境界の深部で発生する超低周波地震に限定することによって実際の現象に即したスケールでのシミュレーションを可能とし，巨大地震発生サイクル中での発生様式の変化という観測可能な指標を提示することに成功しています．こうしたアプローチは地震学におけるシミュレーション研究の今後の方向性の一つを示したものとして高く評価され、平成 26 年度の「日本地震学会論文賞」を受賞いたしました。

また、シミュレーションに用いたプログラムコードは当センターとの共同研究によって開発され、スーパーコンピュータ SX-9 向けに最適化されたものとなっています。

参考 1)日本地震学会 web ページ
http://www.zisin.jp/modules/pico/index.php?content_id=3093
(参考 2)関連する研究成果(当センター広報誌 SENAC 掲載研究成果)
http://www.ss.cc.tohoku.ac.jp/refer/pdf_data/v44-4/v44-4_p35-48.pdf

[Web 版大規模科学計算システムニュースより]

大規模科学計算システムニュースに掲載された記事の一部を転載しています。　http://www.ss.cc.tohoku.ac.jp/tayori/

平成 27 年度講習会計画について (No.192)

平成 27 年度サイバーサイエンスセンター講習会をご案内いたします。多くのみなさまの参加をお待ちしております。

No.	名　称	開催日程	募集人数	講　師	内容概略
1	UNIX 入門	4月 20 日(月) 13:00-16:00 <終　了>	20	共同利用支援係 共同研究支援係	・UNIX システムの基本的な使い方 ・エディタの使い方 ・プログラムの実行方法
2	新スーパーコンピュータ SX-ACE の紹介と利用法	4月 21 日(火) 13:00-15:30 <終　了>	20	共同利用支援係 共同研究支援係	・大規模科学計算システムの紹介と利用法
3	並列プログラミングの概要と Open MP プログラミング入門	4月 22 日(水) 13:00-17:00 <終　了>	20	スーパーコンピューティング研究部	・Open MP による並列プログラミングの基礎 ・利用法
4	MPI プログラミング入門	4月 24 日(金) 10:00-17:00 <終　了>	20	スーパーコンピューティング研究部	・MPI による並列プログラミングの基礎 ・利用法
5	新スーパーコンピュータ SX-ACE における高速化技法の基礎	4月 27 日(月) 13:00-16:00 <終　了>	20	スーパーコンピューティング研究部	・実習によるプログラムの高速化を目的としたスーパーコンピュータの最適化および並列化の基礎
6	高速化ハンズオンセミナー	5月下旬 13:00-16:00	10	スーパーコンピューティング研究部	・利用者持ち込みプログラムのハンズオンセミナー(第5回講習会受講者を対象)
7	可視化システムの利用法	6月 15 日(月) 10:00-17:15	10	メーカ担当者	・可視化システムの基本的な使い方
8	MATLAB 入門	6月 5 日(金) 13:00-17:00	20	陳(秋田県立大)	・MATLAB の基本的な使い方
9	ネットワークとセキュリティ入門	8月 4 日(火) 13:30-16:00	30	水木(ネットワーク研究部)	・ネットワークの基本的な仕組み ・ネットワークの危険性と安全対策
10	Gaussian 入門	8月 28 日(金) 13:00-17:00	20	岸本(理)	・Gaussian の基本的な使い方
11	Mathematica 入門	9月 2 日(水) 13:00-17:00	20	横井(情科)	・Mathematica の基本的な使い方
12	Marc 入門	9月 3 日(木) 13:00-17:00	20	内藤(工)	・Marc の基本的な使い方
13	UNIX 入門	9月 7 日(月) 13:00-16:00	20	共同利用支援係 共同研究支援係	・UNIX システムの基本的な使い方 ・エディタの使い方 ・プログラムの実行方法
14	新スーパーコンピュータ SX-ACE の紹介と利用法	9月 8 日(火) 13:00-15:30	20	共同利用支援係 共同研究支援係	・大規模科学計算システムの紹介と利用法
15	並列プログラミングの概要と Open MP プログラミング入門	9月 9 日(水) 13:00-17:00	20	スーパーコンピューティング研究部	・Open MP による並列プログラミングの基礎 ・利用法
16	MPI プログラミング入門	9月 10 日(木) 10:00-17:00	20	スーパーコンピューティング研究部	・MPI による並列プログラミングの基礎 ・利用法
17	新スーパーコンピュータ SX-ACE における高速化技法の基礎＋ハンズオンセミナー	9月 11 日(金) 10:00-14:00 14:00-17:00	20 (10)	スーパーコンピューティング研究部	・実習によるプログラムの高速化を目的としたスーパーコンピュータの最適化および並列化の基礎 ・ハンズオンセミナー

備考：申し込み、各講習会詳細については、ウェブページ
http://www.ss.cc.tohoku.ac.jp/guide/kosyu.cgi をご覧ください。

平成 27 年度の共同研究について (No.193)

本センターでは、大規模科学計算システムの利用者と共同でプログラムやアルゴリズムを開発する共同研究を行っています。今年度の募集に応募されたものについて共同研究専門部会で審査の結果、以下の９件が採択されましたのでお知らせします。

No.	申請者	所属	研究課題
1	有馬 卓司	東京農工大学大学院 先端電気電子部門	アンテナ放射効率低下メカニズムの解明と放射効率改善手法に関する研究
2	有吉 慶介	海洋研究開発機構 地震津波海域観測研究開発センター	数値シミュレーションに基く余効すべり伝播速度と摩擦特性との関係の解明
3	河野 裕彦	東北大学大学院 理学研究科	密度汎関数緊密結合法を用いたナノスケール分子のナノ秒化学反応シミュレーション
4	茂田 正哉	大阪大学 接合科学研究所	プラズマプロセスにおけるナノ粒子群の集団形成および輸送過程の大規模数値シミュレーション
5	陳 強	東北大学大学院 工学研究科	大規模問題のための超高速モーメント法に関する研究
6	松岡 浩	東北大学 電気通信研究所	連続感度解析の実現を目指した整数型格子ボルツマン法流体解析手法の開発
7	三坂 孝志	東北大学 学際科学フロンティア研究所	Building-Cube 法による大規模高レイノルズ数流れ解析に関する研究
8	森野 裕行	三菱航空機株式会社	民間航空機開発における大規模 CFD 解析の適用
9	横堀 壽光 大見 敏仁	東北大学大学院 工学研究科	血管疾患を有する血管壁の拍動下での時系列変形挙動特性再現シミュレータの開発

（スーパーコンピューティング研究部，共同研究支援係）

計算科学・計算機科学人材育成のための スーパーコンピュータ無償提供制度について (No.193)

東北大学サイバーサイエンスセンターでは、計算科学・計算機科学分野での教育貢献・人材育成を目的として、大学院・学部での講義実習等の教育目的での利用について、無料（ただし、利用状況によっては上限を設定する場合があります）で大規模科学計算システムをご利用いただける制度を用意しております。

利用を希望される場合は、以下の情報を添えて、edu-prog@cc.tohoku.ac.jp までお申し込みください。

・講義担当者氏名
・同所属
・同連絡先（住所，電話，電子メール）
・講義名
・講義実施日時（１セメスターの中で実習を予定している回数）
・センターでの実習利用希望の有無（必要であれば予定日）
・講師派遣の有無
・講義シラバス
・講義ウェブ（もし用意されていれば）
・受講者数（予定）
・必要とする理由（利用目的：例えば、高速数値実験の研修を行うなど）
・期待できる教育効果
・その他（センターへの要望等）

なお、講義終了後、報告書（広報誌 SENAC へ掲載）の提出をお願いいたします。たくさんのお申し込みをお待ちしております。不明な点は、edu-prog@cc.tohoku.ac.jp までお問い合わせください。

（スーパーコンピューティング研究部，共同利用支援係）

民間企業利用サービスについて（No.193）

東北大学サイバーサイエンスセンターでは、社会貢献の一環として大学で開発された応用ソフトウェアとスーパーコンピュータを、民間企業の方が無償または有償にてご利用頂ける制度を用意しております。本サービスにおける利用課題区分は以下の２つとなります。

・大規模計算利用(有償利用)
・トライアルユース(無償利用)

詳細については以下を参照し、利用を希望される場合は共同利用支援係までお申し込みください。

http://www.ss.cc.tohoku.ac.jp/utilize/business.html

問い合わせ先
東北大学サイバーサイエンスセンター
情報部情報基盤課 共同利用支援係
電話: 022(795)6251
E-mail : uketuke@cc.tohoku.ac.jp

（共同利用支援係）

平成 27 年度利用負担金について (No.194)

平成 27 年度の利用負担金は、表 1(大学・学術利用)、表 2(民間機関利用)のとおりとなります。
なお、今後電気料金が高騰した場合には、年度途中において負担経費を値上げする場合があります。あらかじめご了承ください。

表１　基本利用負担金【大学･学術利用】

区　分	項　目	利用形態	負　担　額
演　算 負担経費	スーパー コンピュータ	共有	利用ノード数 1(実行数、実行時間の制限有)　無料(備考２)
			利用ノード数 1～32 まで　経過時間 1 秒につき　0.06 円
			利用ノード数 33～256 まで　経過時間 1 秒につき (利用ノード数-32)×0.002 円+0.06 円
			利用ノード数 257 以上　経過時間 1 秒につき (利用ノード数-256)×0.0016 円+0.508 円
		占有	利用ノード数 32　利用期間 3 ヶ月につき　400,000 円 利用期間 6 ヶ月につき　720,000 円
			利用ノード数 64　利用期間 3 ヶ月につき　720,000 円 利用期間 6 ヶ月につき　1,300,000 円
			利用ノード数 128　利用期間 3 ヶ月につき　1,300,000 円 利用期間 6 ヶ月につき　2,340,000 円
	並列 コンピュータ	共有	利用ノード数 1～6 まで　経過時間 1 秒につき　0.04 円
			利用ノード数 7～12 まで　経過時間 1 秒につき　0.07 円
			利用ノード数 13～18 まで　経過時間 1 秒につき　0.1 円
			利用ノード数 19～24 まで　経過時間 1 秒につき　0.13 円
		占有	利用ノード数 1　利用期間 3 ヶ月につき　160,000 円 (可視化システムの 20 時間無料利用を含む) 利用期間 6 ヶ月につき　320,000 円 (可視化システムの 40 時間無料利用を含む)
ファイル 負担経費	1TB まで無料、追加容量 1TB につき年額		3,000 円
出力 負担経費	大判プリンタによるカラープリント		フォト光沢用紙 1 枚につき　600 円 クロス 1 枚につき　1,200 円
可視化 機器室利用 負担経費	1 時間の利用につき		2,500 円

備考

1　負担額算定の基礎となる測定数量に端数が出た場合は、切り上げる。
2　負担額が無料となるのは専用のジョブクラスで実行されたものとし、制限時間を超えた場合には強制終了する。
3　占有利用期間は年度を超えないものとし、期間中に障害、メンテナンス作業が発生した場合においても、原則利用期間の延長はしない。また、占有利用期間中のファイル負担経費は 10TB まで無料とする。
4　ファイル負担経費については申請日から当該年度末までの料金とする。

表２　基本利用負担金【民間機関利用】

区　分	項　目	利用形態	負　担　額
演　算 負担経費	スーパー コンピュータ	共有	利用ノード数 1(実行数、実行時間の制限有)　無料(備考２)
			利用ノード数 1～32 まで　　経過時間 1 秒につき　0.18 円
			利用ノード数 33～256 まで　経過時間 1 秒につき (利用ノード数-32)×0.006 円+0.18 円
			利用ノード数 257 以上　　経過時間 1 秒につき (利用ノード数-256)×0.0048 円+1.524 円
		占有	利用ノード数 32　利用期間 3 ヶ月につき　1,200,000 円 利用期間 6 ヶ月につき　2,160,000 円
			利用ノード数 64　利用期間 3 ヶ月につき　2,160,000 円 利用期間 6 ヶ月につき　3,900,000 円
			利用ノード数 128　利用期間 3 ヶ月につき　3,900,000 円 利用期間 6 ヶ月につき　7,020,000 円
	並列 コンピュータ	共有	利用ノード数 1～6 まで　　経過時間 1 秒につき　0.12 円
			利用ノード数 7～12 まで　経過時間 1 秒につき　0.21 円
			利用ノード数 13～18 まで　経過時間 1 秒につき　0.3 円
			利用ノード数 19～24 まで　経過時間 1 秒につき　0.39 円
		占有	利用ノード数 1　利用期間 3 ヶ月につき　480,000 円 (可視化システムの 20 時間無料利用を含む) 利用期間 6 ヶ月につき　960,000 円 (可視化システムの 40 時間無料利用を含む)
ファイル 負担経費	1TB まで無料、追加容量 1TB につき年額		9,000 円
出力 負担経費	大判プリンタによるカラープリント　フォト光沢用紙 1 枚につき		1,800 円
			クロス 1 枚につき　3,600 円
可視化 機器室利用 負担経費	1 時間の利用につき		7,500 円

備考

1　負担額算定の基礎となる測定数量に端数が出た場合は、切り上げる。

2　負担額が無料となるのは専用のジョブクラスで実行されたものとし、制限時間を超えた場合には強制終了する。

3　占有利用期間は年度を超えないものとし、期間中に障害、メンテナンス作業が発生した場合においても、原則利用期間の延長はしない。また、占有利用期間中のファイル負担経費は 10TB まで無料とする。

4　ファイル負担経費については申請日から当該年度末までの料金とする。

— SENAC 執筆要項 —

１．お寄せいただきたい投稿内容

次のような内容の投稿のうち、当センターで適当と判定したものを掲載します。その際に原稿の修正をお願いすることもありますのであらかじめご了承ください。

・一般利用者の方々が関心をもたれる事項に関する論説
・センターの計算機を利用して行った研究論文の概要
・プログラミングの実例と解説
・センターに対する意見、要望
・利用者相互の情報交換

２．執筆にあたってご注意いただく事項

(1)原稿は横書きです。
(2)術語以外は、「常用漢字」を用い、かなは「現代かなづかい」を用いるものとします。
(3)学術あるいは技術に関する原稿の場合､200 字～400 字程度のアブストラクトをつけてください。
(4)参考文献は通し番号を付し末尾に一括記載し、本文中の該当箇所に引用番号を記入ください。
　・雑誌：著者, タイトル, 雑誌名, 巻, 号, ページ, 発行年
　・書籍：著者, 書名, ページ, 発行所, 発行年

３．原稿の提出方法

原稿のファイル形式は Word を標準としますが、PDF での提出も可能です。サイズ*は以下を参照してください。ファイルは電子メールで提出してください。

－Word の場合－
　・用紙サイズ：A4
　・余白：上＝30㎜　下＝25㎜　左右＝25㎜　綴じ代＝0
　・標準の文字数（45 文字 47 行）

＜文字サイズ等の目安＞
　・表題＝ゴシック体 14pt 中央　・副題＝明朝体 12pt 中央
　・氏名＝明朝体 10.5pt 中央
　・所属＝明朝体 10.5pt 中央
　・本文＝明朝体 10.5pt
　・章・見出し番号＝ゴシック体 11pt～12pt
　　*余白サイズ、文字数、文字サイズは目安とお考えください。

４．その他

(1)執筆者には、希望により本誌*（10 部以内の希望部数）と本誌 PDF 版を進呈します。
　*2014 年末で、別刷の進呈は終了しました。
(2)投稿予定の原稿が 15 ページを超す場合は共同利用支援係まで前もってご連絡ください。
(3)初回の校正は、執筆者が行って、誤植の防止をはかるものとします。
(4)原稿の提出先は次のとおりです。

東北大学サイバーサイエンスセンター内　情報部情報基盤課共同利用支援係
e-mail　uketuke@cc.tohoku.ac.jp
TEL　　022-795-3406

スタッフ便り

昨年 7 月、センター内に「高性能計算技術開発（NEC）共同研究部門」が開設されました。この研究部門は東北大学と日本電気株式会社との産学連携拠点として開設されたもので、次世代スーパーコンピュータに向けた要素技術の研究やアプリケーションプログラムの高速化や並列化に関する研究を行っています。

私も 7 月からこの研究部門の一員として研究活動を始めました。昨年度の成果の一つは、2 月に稼働したスーパーコンピュータシステム SX-ACE を使ったリアルタイム津波浸水シミュレーションシステムの開発です。3 月に開催された国連防災世界会議のパブリックフォーラムでは、SX-ACE の実機を使い、南海トラフ地震を想定したデモンストレーションを行いました。このデモンストレーションで、地震発生から約 17 分で高知市内の津波浸水の予測を高知県庁へ送付できることを実証しました。

今年度は、このリアルタイム津波浸水シミュレーションシステムのさらなる強化を行っていきたいと思っています。スーパーコンピュータが社会インフラの一部になり、安全・安心な社会を実現できればと思っています。（A. M）

新スーパーコンピュータ SX-ACE の運用がいよいよ始まり、忙しくも充実した毎日を送っています。昨年 4 月からセンターの技術職員として働きはじめ、早いものでもう 1 年です。社会人になってから、職場も仕事もガラッと変わったのは今回が初めて。この歳で新しいことを一から習得することの大変さを痛感したり、職場が変わるとこんなに文化が違うのかとカルチャーショックを受けたりしつつも、多少はセンター職員として様になってきたかな、と思う今日この頃です。まだまだ勉強勉強の日々ですが、有効にシステムをお使いいただけるよう微力ながらもサポートしていければと思っています。

新年度もどうぞよろしくお願いいたします。（A. S）

整備中の青葉山新キャンパス

平成 27 年 4 月発行
編集・発行　東北大学
サイバーサイエンスセンター
仙台市青葉区荒巻字青葉 6-3
郵便番号　980-8578
印　刷　東北大学生活協同組合
プリントコープ

システム一覧

計算機システム	機　種
スーパーコンピュータ	SX-ACE
並列コンピュータ	LX 406Re-2

サーバ名

フロントエンドサーバ	front.cc.tohoku.ac.jp
SSH アクセス認証鍵生成サーバ	key.cc.tohoku.ac.jp

サービス時間

利用システム名等	利用時間帯
スーパーコンピュータ	連　続　運　転
並列コンピュータ	連　続　運　転
可視化機器室	平日 9:00～21:00
館 内 利 用	平日 8:30～21:00

スーパーコンピュータ（SX-ACE）の利用形態と制限値

利用形態	利用ノード数 ※1	実行時間制限 (経過時間)	メモリサイズ制限	-q オプション	-b オプション
通常	1～256	規定値：2 週間 最大値：1 ヶ月	60GB×ノード数	sx	利用ノード数
	257～1,024	規定値：1 ヶ月 最大値：1 ヶ月			
無料	1	1 時間	60GB		f
デバッグ	1～16	2 時間	60GB×ノード数	debug	利用ノード数
	17～32	24 時間			

並列コンピュータ（LX 406Re-2）の利用形態と制限値

利用形態	利用ノード数 ※1	実行時間制限 (経過時間)	メモリサイズ制限	-q オプション	-b オプション
通常	1～24	規定値：1 ヶ月 最大値：1 ヶ月	128GB×ノード数	lx	利用ノード数
アプリケーション	1	なし	128GB		a
会話型	1 (6 コアまで)	1 時間 (CPU 時間合計)	8GB	-	-

※ 1. 2ノード以上を利用した並列実行にはMPIの利用が必用

目次

東北大学サイバーサイエンスセンター
大規模科学計算システム広報　Vol.48 No.2 2015–4