ISSN 1346-4353

九州大学 情報基盤センター

広報

全国共同利用版
2001年 Vol. 1 No. 2

目次

解　説
VPP5000/64利用法 …… 渡部善隆、田中省作　67
たかが内積、されど内積…… 藤野清次　92
プログラム開発のための補助ツール(1) －rsync－…… 池田大輔　104
九州大学情報基盤センター研究用計算機システム
センターニュース（全国共同利用施設版）の案内 ……　112
海外研究会動向
PAKDD01参加報告…… 廣川左千男　115
ICOIN-15 国際会議に出席して …… 岡村耕二　117
報　告
業務報告 ……　119
講習会報告 ……　122
お知らせ ……　126
人事異動 ……　129
編集後記 ……　130

Computing and Communications Center
Kyushu University

www. cc. kyushu-u. ac. jp

センターアクセス一覧

インターネット

接続システム名	ドメイン名	ＩＰアドレス
ＧＰ７０００Ｆ／９００	kyu-cc.cc.kyushu-u.ac.jp	133.5.9.1
ＶＰＰ５０００／６４	kyu-vpp.cc.kyushu-u.ac.jp	133.5.8.160
端末サーバ	wisdom.cc.kyushu-u.ac.jp	133.5.8.1
ＧＳ３２０	kyu-ss.cc.kyushu-u.ac.jp	133.5.8.158

交換回線使用電話番号

接続システム	通　信　速　度	制御手順	電　話　番　号
端末サーバ	14400～64000 (ISDN回線,一般電話,PIAFS)	PPP（同　期） PPP（非同期）	092-642-7340

VPP5000/64 利用法

渡部善隆*, 田中省作*

本稿では, 2001年1月から運用を開始した新スーパーコンピュータ FUJITSU VPP5000/64(ホスト名: kyu-vpp) の利用方法を, Fortran, C, C++ プログラムの翻訳・実行方法を中心に説明します.

1 VPP5000/64への接続

1.1 ホスト名

ネットワーク経由で接続する場合のホスト名は以下の通りです.

ホスト名	kyu-vpp
ドメイン名	cc.kyushu-u.ac.jp
IPアドレス	133.5.8.160

telnet, rlogin などで kyu-vpp に接続する場合には,「ホスト名+ドメイン名」の“kyu-vpp.cc.kyushu-u.ac.jp”またはIPアドレスの“133.5.8.160”を指定します. IPアドレスは予告なしに変更になる場合がありますので,「ホスト名+ドメイン名」の指定方法をお勧めします.

! VPP700/56のときとはIPアドレスが変更になっています.

詳細は以下のページを参照してください.

接続方法について

- UNIXからの遠隔ログイン
  http://www.cc.kyushu-u.ac.jp/scp/system/manual/etc/RemoteLogin.html
- MS-Windowsでの Telnet と Secure Shell(SSH) の利用方法
  http://www.cc.kyushu-u.ac.jp/scp/system/general/GP7000F/telnet.win/
- Macintoshでの Telnet 利用方法
  http://www.cc.kyushu-u.ac.jp/scp/system/general/GP7000F/telnet.mac/telnet.html

ファイルの転送方法について

- UNIXからのファイル転送
  http://www.cc.kyushu-u.ac.jp/scp/system/manual/etc/FTP.html

*情報基盤センター 研究部 E-mail: {watanabe, sho}@cc.kyushu-u.ac.jp

- WS_FTP の使い方
  `http://www.cc.kyushu-u.ac.jp/scp/network/WSFTP/index-j.shtml`
- MacOS でのファイル転送
  `http://www.cc.kyushu-u.ac.jp/scp/system/general/GP7000F/telnet.mac/ftp.html`

1.2 オペレーティングシステム

kyu-vpp のオペレーティングシステム (OS) は，UNIX OS の一種である **UXP/V V20** です．UNIX を使ったことのある方ならば，違和感なく kyu-vpp を利用することができます．日本語文字コードは EUC，ログインシェルは csh です．UNIX の基本的な利用方法については [4] を参照してください．

1.3 kyu-cc からのファイル参照

汎用 UNIX サーバ kyu-cc からスーパーコンピュータ VPP5000/64 のファイルを 参照することができます．kyu-cc ホームディレクトリ (ログイン直後の利用者ディレクトリ) の "VPP" という名前のディレクトリと kyu-vpp のホームディレクトリとの間にシンボリックリンクが張られています．ディレクトリ VPP に移動することにより，kyu-vpp のファイルを kyu-cc のアプリケーションソフトウェアの入力データに指定したり，kyu-vpp のファイルの編集，バッチジョブの投入などを行うことができます．

1.4 エディタ

kyu-vpp で利用できるエディタは以下の通りです．

エディタ名	コマンド名	備考
emacs	emacs(/usr/local/bin/emacs)	GNU Emacs 20.7.1; 日本語入力は Free Wnn 1.1
vi	vi(/usr/bin/vi)	利用方法は man vi で参照して下さい
je	je(/usr/local/bin/je)	PFD[1]ライクなエディタ

1.5 VPP5000/64 用環境設定ファイル

2000 年末まで稼動していた旧スーパーコンピュータ VPP700/56 から継続して VPP5000/64 を利用されている場合には，.cshrc，.login，.profile ファイルは VPP5000/64 に コピーされています．センターでは VPP5000/64 用のファイルの雛型も以下に用意していますので，これを機会に変更することをお勧めします．

```
.cshrc    :  /usr/local/skel/local.cshrc
.login    :  /usr/local/skel/local.login
.profile  :  /usr/local/skel/local.profile
```

[1] 旧汎用計算機 (OS は MSP) で使用されていたエディタです．

コピー方法は以下の手順を参考にしてください.

```
kyu-vpp% mv .cshrc .cshrc.vpp700                    ←バックアップファイルを作成
kyu-vpp% cp /usr/local/skel/local.cshrc .cshrc      ←VPP5000/64用 .cshrc ファイル
                                                      をコピー
```

以上の操作では，現在使用中の csh に，新しい.cshrc の設定は反映されません．変更を直ちに反映させるには，次のように source コマンドを実行する必要があります.

```
kyu-vpp% source .cshrc
```

2 ソフトウェア一覧

VPP5000/64 で利用できるプログラム言語，数値計算ライブラリ，アプリケーションソフトウェアの一覧を示します.

2.1 プログラム言語

ソフトウェア名	コマンド	機能
Fortran	frt	JIS X 3000-1 完全準拠 Fortran 95 コンパイラ
VPP Fortran	frt -Wx	ベクトル並列化コンパイラ
HPF	frt -Wh	分散メモリ型並列計算機向け言語
C	cc -Kvp	ANSI 規格準拠，K&R 仕様 [2] をサポートした C コンパイラ
C++	CC	ARM 仕様 [1] 準拠の C++コンパイラ
DPCE	dpcc	データ並列処理用 C コンパイラ

2.2 プログラム言語開発支援

ソフトウェア名	機能
Analyzer	チューニングおよびデバッグ支援ツール
Workbench	GUI 環境によるプログラム開発支援ツール
MPTools	メッセージパッシングプログラム解析ツール
TotalView	メッセージパッシングプログラムデバッグ支援ツール

2.3 メッセージパッシングライブラリ

ソフトウェア名	機能
MPI	MPI 2.0 規格準拠のメッセージパッシングライブラリ
PVM	PVM 3.3 準拠のメッセージパッシングライブラリ

2.4　数値計算ライブラリ

ソフトウェア名	機能
SSL II/VP	Fortran 用汎用数値計算ライブラリ
SSL II/VPP	SSL II/VP の並列版
C-SSL II/VP	SSL II/VP サブルーチンとの C 言語，C++言語インターフェース
NUMPAC	Fortran 用数値計算パッケージ
LAPACK/VP, BLAS/VP	線形計算ライブラリ
ScaLAPACK	並列版線形計算ライブラリ
IMSL Fortran 90 MP Library	MPI 環境の並列ライブラリ (Fortran 90 用)
IMSL C Library	C 用関数ライブラリ
SALS	最小二乗法プログラムパッケージ

2.5　計算化学

ソフトウェア名	機能
Gaussian 98	非経験的分子軌道計算プログラム
MOPAC2000	半経験的分子軌道計算プログラム
xmo	Gaussian, MOPAC2000 用可視化プログラム
MASPHYC / MASPHYC-SP	材料の物性・構造予測システム

2.6　アプリケーション・ソフトウェア

ソフトウェア名	機能
α-FLOW	汎用 3 次元流体解析システム
Marc / Mentat	汎用有限要素法解析プログラム
Nastran / Patran	汎用非線形構造解析プログラム
LS-DYNA	非線形動的構造解析プログラム

2.7　プログラムライブラリ開発課題

Fortran サブルーチンライブラリ

プログラム名	機能
CGJQ Gauss-Jacobi	積分公式の分点，重み算定
CGLQ Gauss-Laguerre	積分公式の分点，重み算定
DIFF1S, DIFF1D	解析関数の数値微分 (単・倍精度)
MINMAX	線形方程式のミニマックス解
PRESNL	一般化されたフレネル積分
CA01	計量修正による関数の極小点発見
DA02	BCS 方程式
DB01, DB02, DB03	Clebsch-Gordan, Racah, 9-J 係数
KNL1	直交条件模型の積分核
VAR1	微積分方程式の散乱境界条件解
AACOUST	建築音響解析ライブラリ

詳細は，次のページを参照してください．

http://www.cc.kyushu-u.ac.jp/scp/system/library/PROGRAM_LIBRARIES/Fortran_Subroutines/

3 利用形態と制限値

3.1 利用形態

kyu-vpp でプログラムを翻訳・編集結合・実行する方法は，大きく**対話型処理**と**バッチ処理**に分かれます．対話型処理とは，コマンドを入力することによって翻訳処理や実行などをインタラクティブに行う利用方法のことです．対話型処理は，翻訳やデバッグ，小規模なプログラムの実行に適しています．バッチ処理とは，一連の処理を記述した「バッチリクエスト」と呼ばれるシェルスクリプトによって処理を行う方法です．バッチ処理は，多くの記憶領域を必要とするプログラムや並列度の高いジョブに適しています．

なお，UNIX でよく行う，コマンドラインの最後に“&”をつける処理方法（バックグラウンドジョブ）は**バッチ処理ではなく対話型処理とみなされます**ので注意してください．

3.2 制限値

対話型処理，バッチ処理の制限値は以下の通りです．バッチ処理の詳細は「7. バッチ処理」を参照してください．

処理形態	キュー名	記憶容量		演算時間	備考
		省略値	制限値		
対話型処理	–	1GB		1 時間	非並列
バッチ処理	s	2GB	7GB	1 時間	非並列・短時間向き
	p1	2GB	14.5GB	20 時間	非並列 (省略キュー)
	s8	2GB/PE	7GB/PE	1 時間	8PE まで使用可・短時間向き
	p8	2GB/PE	7GB/PE	20 時間	8PE まで使用可
	p16	2GB/PE	7GB/PE	20 時間	16PE まで使用可
	p32	2GB/PE	7GB/PE	20 時間	32PE まで使用可
	x8	10GB/PE	15GB/PE	20 時間	8PE まで使用可・大規模記憶容量向き
	x16	10GB/PE	15GB/PE	20 時間	16PE まで使用可・大規模記憶容量向き

4 Fortran の対話型処理

より詳しい利用方法は次のオンラインマニュアル[3]を参照してください.

- UXP/V Fortran 使用手引書 V20用
- UXP/V Fortran メッセージ説明書 V20用
- UXP/V Fortran プログラミングハンドブック V20用
- UXP/V Fortran User's Guide V20
- UXP/V Fortran Messages (V20)
- UXP/V Fortran Programming Handbook V20

4.1 コマンドとファイル拡張子名

Fortran の翻訳と結合編集のコマンドは frt です. kyu-vpp にログイン後, man frt と入力すると, 機能の詳細を表示させることができます. Fortran ソースファイルの拡張子はプログラムの形式に応じて以下のようにしてください.

拡張子	プログラムの種類
.f90 または .f95	自由形式
.f	固定形式

ソース形式の解釈は, 翻訳時オプション-Fixed または-Free で変更することもできます.

! 拡張子“.f90”または“.f95”のファイルは通常,「自由形式で記述された Fortran 90 または Fortran 95 プログラム」と解釈されます. これに対し, 拡張子“.f”のファイルは通常「固定形式で記述された Fortran 77 プログラム」と解釈されます. したがって, 拡張子“.f”のファイルに Fortran 90 または Fortrn 95 から新たにサポートされた機能を記述する場合には, 翻訳時オプション-X9 が必要になります.

4.2 基本的な手順

まず, frt コマンドにより翻訳および結合編集を行い, 実行可能ファイルを作成します. 以下は, ファイル名を“example.f95”とした例です.

```
kyu-vpp% frt example.f95    ←翻訳と結合編集により実行可能ファイルを作成
```

処理が致命的なエラーを起こすことなく終了した場合, 実行可能ファイル“a.out”が作成されています. 実行はファイル名をコマンドとして入力します.

```
kyu-vpp% ./a.out    ←実行
```

! a.outに先だって指定する“./”は,「現在作業しているディレクトリ(カレントディレクトリ)にあるファイルを対象にする」という意味です. 旧kyu-vpp(VPP700/56)では,“a.out”と指定するだけでプログラムを実行することができました. しかし,セキュリティ強化のため,新kyu-vppではデフォルトではカレントディレクトリのパスを設定していません. 必ず最初に“./”を指定してください.

4.3 オブジェクトファイルの作成と利用方法

翻訳時オプション-cを指定することにより,結合編集を行わず,オブジェクトファイルの作成までを行うこともできます.

```
kyu-vpp% frt -c example.f95    ←オブジェクトファイルの作成
```

オブジェクトファイルの拡張子は.oです. 例では,“example.o”という名前のファイルが作成されます.

オブジェクトファイルは,既にデバックが終了した副プログラムをメインプログラムと切り離して管理する場合によく用いられます. 例として,メインプログラムを“main.f95”,副プログラム群を記述したファイルを“sub.f”とします. まず,sub.fを-cオプションを付けて翻訳し,オブジェクトファイルsub.oを作成します.

```
kyu-vpp% frt -c sub.f    ←オブジェクトファイルの作成(プログラムは固定形式)
```

次に,メインプログラムを翻訳し,オブジェクトファイルを結合します.

```
kyu-vpp% frt main.f95 sub.o    ←実行可能ファイルの作成
```

このようにすると,副プログラムの翻訳は一度だけで済むため,メインプログラムを何度も修正する場合に全体の翻訳時間が短縮できます.

frtコマンドには上の例のようにオブジェクトファイルも指定することができます. また,複数のソースファイル,オブジェクトファイルも指定できます.

4.4 よく用いる翻訳時オプション

よく用いるFortranの翻訳時オプションを以下に示します. 詳細はオンラインマニュアル[3]を参照してください.

-c	オブジェクトファイルの作成までを行います．結合編集を行わず実行可能ファイルは作成されません．
-o *filename*	実行可能ファイル名またはオブジェクトファイル名を *filename* に変更します．
-Free	自由形式の Fortran プログラムとして翻訳します．
-Fixed	固定形式の Fortran プログラムとして翻訳します．
-Dasux	プログラムのデバッグのため，引数の妥当性の検査，添字式・部分列範囲の検査，未定義データの引用の検査を行います．メッセージは翻訳時，実行時に出力されます．実行時間が増大するため，小規模なプログラムに対しデバッグを行い，デバッグ終了後は必ず，実行可能ファイル，オブジェクトファイルは再作成してください．
-Am	モジュールを翻訳します．
-Ob	最適化を基本的な機能のみに制限します．
-O5	最大限の最適化を試みます．数学的に等しい範囲で括弧を無視した変更等を行うため，場合によっては 計算結果に大きな差が生じることがあります．指定する場合には注意が必要です．
-Kfast	VPP5000/64 に適した最適オプションを自動選択 (コンパイラに“おまかせ”) します．
-Eeipu	最適化に関するメッセージを出力します．
-X9	言語仕様が Fortran 95 であると解釈して翻訳します．拡張子が .f のファイルに Fortran 90 から新たにサポートされた組込み関数などを記述する場合に必要です．
-Pa	ソースプログラムと対比させながら配列演算に対するベクトル化情報を出力します．
-Z *filename*	翻訳に関する情報を *filename* に書き出します．
-KA32	オブジェクトファイルを 32 ビットアドレスモードで作成します．省略値は 64 ビットモードです．プログラムによっては実行性能が向上する場合があります．ただし，64 ビットアドレスモードのオブジェクトファイルとの混在はできない上に，記憶容量の最大値が 2GB に制限されるため，指定する場合には注意が必要です．

以下は，デバッグオプションを指定して翻訳・編集結合し，実行する例です．

```
kyu-vpp% frt -Dasux example.f95          ←デバッグオプションの指定
kyu-vpp% ./a.out                         ←実行を通してデバッグ
 jwe0340i-e line 7 ベクトル化された配列要素または文字部分列 (M) の引用で,
2次元目の添字式または部分列式の値 (1002) は，宣言した範囲 (1:1000) 内で
なければなりません.
      :
```

以下は，“おまかせ”最適化オプションの指定 (-Kfast) とともに，プログラムリストと対比させたベクトル化情報 (-Pa) と最適化情報 (-Eeipu) を出力させ，結果をファイル“clist”に書き出す例です．

```
kyu-vpp% frt -Kfast -Pa -Eeipu -Z clist example.f90     ←オプションの指定例
```

この場合，オプションの順番は任意です．

4.5 よく用いる実行時オプション

実行時オプションは実行可能ファイル名の後に“-Wl”(“l”は英小文字)を指定し，続いてオプションを指定します．複数の実行時オプションはカンマ（,）で区切って続けます．

よく用いる実行時オプションを以下に示します．詳細はオンラインマニュアル[3]を参照してください．

-u	浮動小数点アンダーフローが発生した場合，エラーメッセージを出力します．
-C *number*	書式なし入出力文において，装置番号 *number* からIBM370形式浮動小数点データのファイルを入出力するときに指定します．*number* を省略した場合すべての装置番号に対して有効となります．
-M	-Cの指定により行われるIBM370-IEEE形式浮動小数点データの変換によって仮数部のビットが損失したときに，診断メッセージを出力します．
-Oi	入出力に関する統計情報を出力します．

以下は，実行時オプション-Cを指定して，装置番号10番からIBM370形式浮動小数点データのファイル入出力を行い，あわせて診断メッセージを出力する例です．

```
kyu-vpp% ./a.out -Wl,-C10,-M    ← IBM370形式浮動小数点データのファイル入出力
```

4.6 最適化オプション

Fortranの最適化レベルは標準レベルである“-Oe”(-O3と等価)です．“-Oe”レベルでは，演算評価方法の変更，不変式の先行評価，総和演算のベクトル化などが行われます．これらの最適化の副作用として，計算結果に違いが生じたり，正しいプログラムにもかかわらずごくまれに実行時に異常終了したりすることがあります．最適化機能の使用上の注意についてはオンラインマニュアル[3]を参照してください．

4.6.1 基本最適化レベルでの翻訳

最適化を基本レベルに抑えた翻訳を行う場合には，“-Ob”オプションまたは“-O2”オプションを指定します．作成したプログラムの精度を標準オプションと確認する場合などにも利用できます．

```
kyu-vpp% frt -Ob main.f90     ←最適化オプションを下げる
kyu-vpp% ./a.out              ←実行
```

4.6.2 最適化オプションの指定例

-Kfastオプションを指定して“おまかせ最適化”をおこないます．翻訳時間は一般に増大します．プログラムによってはかなりの高速化が得られる場合があります．

```
kyu-vpp% frt -Kfast main.f90        ←最適化オプションの指定
kyu-vpp% ./a.out                    ←実行
```

4.6.3 最大限の最適化オプション

最大限の最適化オプションは-O5 です．さらに，記憶領域が 2GB 以下であれば，-KA32 オプションを指定することで，さらに高速になる場合があります．ただし，実行結果に副作用が生じる可能性があり，-KA32 オプションを指定して作成したオブジェクトファイルは通常の 64 ビットアドレスモードのオブジェクトファイルと互換性がありません．使用する場合は注意してください．

```
kyu-vpp% frt -O5 -KA32 main.f90          ←最大限の最適化
```

4.7 時間計測コマンド

timex(/usr/bin/timex) コマンドにより，翻訳，実行などの経過時間，CPU 時間を計測することができます．

```
kyu-vpp% timex frt example.f95      ←翻訳と結合編集の時間計測
real            4.01                ←経過時間 (4 秒 1)
user            1.70                ←ユーザ CPU 時間
sys             0.80                ←システム CPU 時間
vu-user         0.00
vu-sys          0.00

kyu-vpp% timex ./a.out              ←実行時間の計測
eal            10.30                ←経過時間 (10 秒 3)
user           10.24                ←ユーザ CPU 時間
sys             0.03                ←システム CPU 時間
vu-user         8.21                ←ユーザ CPU 時間の中でベクトルユニットが
                                      動作した時間
vu-sys          0.00                ←システム CPU 時間の中でベクトルユニット
                                      が動作した時間
```

4.8 数値計算ライブラリの組み込み

数値計算ライブラリ，図形ライブラリを使用する場合には，frt コマンドのオプション-l を用いてライブラリを指定します．-l は結合編集時のオプションのため，通常は次のように，コマンド並びの最後に指定してください．

```
kyu-vpp% frt main.f90 -lssl2vp      ← SSL II/VP の組み込み
```

利用できるライブラリとオプションは以下の通りです．

ライブラリ名	オプション
SSL II/VP	-lssl2vp
SSL II/VPP	-lssl2vpp (実行可能ファイルを作成するまで)
NUMPAC	-lnumpac
BLAS	-lblasvp
LAPACK	-llapackvp -lblasvp
ScaLAPACK	-scalapack
IMSL Fortran 90 MP Library	http://www.cc.kyushu-u.ac.jp/scp/system/library/IMSL/kyu-vpp.html を参照してください
プログラムライブラリ開発課題分	-lsslq

4.9 ファイル入出力

ここでは，OPEN文にファイル名を陽に指定しない場合のファイル入出力について簡単に紹介します．詳細はオンラインマニュアル[3]を参照してください．

4.9.1 標準入出力

装置番号5番(READ(5)に対応)と6番(WRITE(6))は通常端末の入出力になります．UNIXの「リダイレクション機能」を用いることで，これらの入出力をファイルに切替えることができます．ただし，操作に慣れるまでは，**リダイレクションの方向に十分注意する**ようにしてください．以下の例では，実行可能ファイルを“a.out”，入力ファイルを“in.data”，出力ファイルを“out.data”としています．

```
kyu-vpp% ./a.out < in.data                ←装置番号5番から in.data の内容を読み込む
kyu-vpp% ./a.out > out.data               ←装置番号6番への出力結果を out.data に保存
kyu-vpp% ./a.out >> out.data              ←装置番号6番への出力結果を既存の out.data
                                            に追加書き
kyu-vpp% ./a.out >& out.data              ←装置番号6番への出力結果および実行時
                                            のエラーメッセージを out.data に保存
kyu-vpp% ./a.out < in.data > out.data     ← in.data の内容を読み込み out.data に保存
```

4.9.2 環境変数による結合

標準入出力以外の装置番号とファイルとの結合は環境変数“fuXX”で行います．XXに装置番号を指定します．番号が一桁の場合“fu03”などと指定してください．環境変数を設定せずにファイル出力を行った場合は，“fort.XX”というファイルに出力されます．

以下の例は，装置番号10番とファイル“example.data”を結合し，実行する例です．

```
kyu-vpp% setenv fu10 example.data          ←ファイルと装置番号の結合
kyu-vpp% ./a.out                           ←実行
```

4.9.3 書式なし入出力

書式なし入出力を行う場合には，OPEN文にFORM='UNFORMATTED'を指定してください．Fortran 95仕様からは，明示的に指定しないとエラーとなります．

```
REAL(KIND=8),DIMENSION(100) :: X                    ←配列の宣言
  :
OPEN(1,FILE='EXAMPLE.DATA',FORM='UNFORMATTED')  ←書式なしデータを開く
READ(1) X                                           ←書式なし入力
CLOSE(1)                                            ←ファイルを閉じる
```

5 C, C++ の対話型処理

より詳しい利用方法は次のオンラインマニュアル[3]を参照してください．

- UXP/V C言語使用手引書 V20用
- UXP/V C++オンラインマニュアル
- UXP/V C Language User's Guide V20
- UXP/V C++ Online Manual

5.1 コマンドとファイル拡張子名

Cの翻訳と結合編集のコマンドはccです．C++の翻訳と結合編集のコマンドはCCです．ccコマンドに“-Kvp”オプションを指定することにより，ベクトル翻訳を行います．省略した場合にはスカラー翻訳を行います．CCコマンドでは“-Kvp”オプションが省略値です．

! VPP700/56ではCのスカラー翻訳コマンドがcc，ベクトル翻訳コマンドがvccでした．VPP5000/64では，ccコマンドのオプションで使い分けるように変更になっています．

ソースファイルの拡張子は以下の名前にしてください．

プログラムの種類	拡張子
C	.c
C++	.ccまたは.Cまたは.c

5.2 基本的な手順

ccコマンドまたはCCコマンドにより翻訳および結合編集を行い，実行可能ファイルを作成します．以下は，ファイル“example.c”から実行可能ファイルを作成する例です．

```
kyu-vpp% cc example.c    ←翻訳と結合編集により実行可能ファイルを作成
                           (Cの場合・スカラー翻訳)
```

処理が正常に終了した場合，実行可能ファイル“a.out”が作成されています．実行はファイル名をコマンドとして入力します．

```
kyu-vpp% ./a.out    ←実行
```

ベクトル翻訳を行う場合には“-Kvp”オプションを指定します．配列要素の繰返し演算が多く含まれているプログラムの場合，**実行時間が大幅に短縮**することが期待されます．

```
kyu-vpp% cc -Kvp example.c    ←翻訳と結合編集により実行可能ファイルを作成
                                (Cの場合・ベクトル翻訳)
```

5.3 オブジェクトファイルの作成と利用方法

翻訳時オプション-cを指定することにより，結合編集を行わず，オブジェクトファイルの作成までを行うこともできます．

```
kyu-vpp% cc -c example.c    ←オブジェクトファイルの作成
```

オブジェクトファイルの拡張子は.oです．例では，“example.o”という名前のファイルが作成されます．

オブジェクトファイルは，既にデバックが終了した副プログラムをメインプログラムと切り離して管理する場合によく用いられます．例として，メインプログラムを“main.c”，副プログラム群を記述したファイルを“sub.c”とします．まず，sub.cを-cオプションを付けて翻訳し，オブジェクトファイルsub.oを作成します．

```
kyu-vpp% cc -c sub.c    ←オブジェクトファイルの作成
```

次に，メインプログラムを翻訳し，オブジェクトファイルを結合します．

```
kyu-vpp% cc main.c sub.o    ←実行可能ファイルの作成
```

このようにすると，副プログラムの翻訳は一度だけで済み，メインプログラムを何度も修正する場合に全体の翻訳時間が短縮できます．

cc, CCコマンドには上の例のようにオブジェクトファイルも指定することができます．また，複数のソースファイル，オブジェクトファイルも指定できます．

5.4 よく用いる翻訳時オプション

よく用いるC, C++の翻訳時オプションを以下に示します．詳細はオンラインマニュアル[3]を参照してください．

-Kvp	ベクトル翻訳を行うことを指示します.
-c	オブジェクトファイルの作成までを行います. 結合編集を行わず実行可能ファイルは作成されません.
-o *filename*	実行可能ファイル名またはオブジェクトファイル名を *filename* に変更します.
-O -K4	最大限の最適化を指示します. 副作用が生じる可能性がありますので, 指定する場合には注意が必要です.
-Wv,-m3	ベクトル化メッセージの情報を出力します.
-Ksrc	ソースリストに対応した最適化・ベクトル化情報を出力します.
-Ksta	統計情報を出力します.
-Z *filename*	翻訳に関する情報を *filename* に書き出します. 長大なエラーメッセージや-Ksrc オプションの結果をファイルに書き出す場合に利用します.
-KA32	オブジェクトファイルを 32 ビットアドレスモードで作成します. 省略値は 64 ビットモードです. プログラムによっては実行性能が向上する場合があります. ただし, 64 ビットアドレスモードのオブジェクトファイルとの混在はできない上に, 記憶容量の最大値が 2GB に制限されるため, 指定する場合には注意が必要です.

以下は, 最適化オプションを指定してベクトル翻訳・編集結合し, 実行する例です.

```
kyu-vpp% cc -Kvp -O -K4 example.c   ←最適化オプションの指定
kyu-vpp% ./a.out                    ←実行
```

5.5 数値計算ライブラリの組み込み

ライブラリ名	オプション
C-SSL II/VP	-Wg, -S -DMAIN_=main
IMSL C Library	($CFLAGS) と ($LINK_CNL) の設定

数値計算ライブラリの利用方法は以下のページを参考にしてください.

- C-SSL II/VP
  http://www.cc.kyushu-u.ac.jp/scp/system/library/SSL2/C-SSL2.html
- IMSL C Library
  http://www.cc.kyushu-u.ac.jp/scp/system/library/IMSL/IMSL.html

6 並列プログラムの対話型翻訳

VPP5000/64 は分散メモリ型ベクトル並列計算機です. 現在のところ, 自動並列化機能は有していません. 複数 PE を利用した並列計算のためには, 利用者自身で並列化を行う必要があります.

VPP5000/64 で可能な並列化手法は以下の通りです.

ソフトウェア名	並列化の概要
VPP Fortran	Fortranプログラムに指示行を挿入
HPF	Fortranプログラムに指示行を挿入
DPCE	ANSI-C言語のデータパラレル型拡張仕様
MPI	Fortran, Cプログラムからライブラリ関数を呼出す
PVM	Fortran, Cプログラムからライブラリ関数を呼出す

VPP5000/64の並列プログラムは，対話型に翻訳，結合・編集を行い，実行可能ファイルを作成することができます．実行はバッチ処理で行います．以下，各並列化の概要を説明します．

6.1 VPP Fortran

詳細は次のオンラインマニュアル[3]を参照してください．

- UXP/V Fortran/VPP使用手引書 V20用
- UXP/V VPP Fortranプログラミングハンドブック V20用
- UXP/V Fortran/VPP User's Guide (V20)
- UXP/V VPP Fortran Programming Handbook (V20)

VPP Fortranプログラムの翻訳のためには，frtコマンドに-Wxオプションを指定します．このオプションの指定によって，VPP Fortranコンパイラが起動され，ソースプログラムに挿入した「拡張最適化制御行」が有効になります．

```
kyu-vpp% frt -Wx example.f90      ←VPP Fortranコンパイラの起動
```

6.2 HPF(High Performance Fortran)

詳細は次のオンラインマニュアル[3]を参照してください．

- UXP/V HPF使用手引書 V20L20 L00061用
- UXP/V HPF User's Guide (V20L20 L00061)

HPFプログラムの翻訳のためには，frtコマンドに-Whオプションを指定します．このオプションの指定によって，HPFコンパイラが起動され，ソースプログラムに挿入した「HPF指示行」が有効になります．

```
kyu-vpp% frt -Wh example.f90      ←HPFコンパイラの起動
```

6.3 DPCE(Data-Parallel C Extensions)

詳細は次のオンラインマニュアル[3]を参照してください．

- UXP/V DPCE使用手引書 V20L20

- UXP/V DPCE User's Guide V20

DPCEプログラムの翻訳および実行可能ファイルの作成はkyu-vppのdpccコマンドによって行います．DPCEソースプログラムの拡張子は“.dpc”とします．

```
kyu-vpp% dpce example.dpc      ←DPCEプログラムの翻訳，編集結合
```

6.4 MPI(Message Passing Interface)

詳細は次のオンラインマニュアル[3]を参照してください．

- UXP/V MPI使用手引書 V20用
- UXP/V MPI User's Guide V20

MPIプログラムの翻訳および実行可能ファイルの作成はkyu-vppのmpifrtコマンド(Fortranの場合)またはmpiccコマンド(Cの場合)によって行います．

```
kyu-vpp% mpicc example.c      ←MPIプログラムの翻訳，編集結合
```

6.5 PVM(Parallel Virtual Machine)

詳細は次のオンラインマニュアル[3]を参照してください．

- UXP/V PVM使用手引書 V20用
- UXP/V PVM User's Guide V20

PVMプログラムの翻訳および実行可能ファイルの作成はkyu-vppのfrtコマンド(Fortranの場合)またはccコマンド(Cの場合)にPVMライブラリを指定して行います．以下は，指定例です[2]．詳細はマニュアルを参照してください．

```
kyu-vpp% cc  test.c -Wl,-P \
? -L/usr/lang/pvm64/lib/UXPV \
? -J -dy -lpvm -lmp -lgen -lelf -lsocket -lpx \
?  -lc -I/usr/lang/pvm64(pvm32)/include          ←PVMプログラムの翻訳，編集結合
```

“”

7 バッチ処理

対話型の制限値を超えるジョブおよび並列プログラムの実行はバッチ処理で行います．

バッチ処理は対話型処理に比較して負担金が安価に設定されています．バッチ処理は，対話型処理で行う一連の処理の流れを「バッチリクエスト」と呼ばれるシェルスクリプトに記述し，qsubコマンドによって投入します．バッチ処理に用いるコマンドは以下の通りです．

[2]指定するオプションが多い場合など，例のように“\”を入れることで，複数行に渡ってコマンドを入力することができます．

コマンド	機能
qsub *options filename*	*filename*に記述したバッチリクエストを投入
qstat	バッチキューの状態表示
qps	投入したジョブの実行状況を表示
qdel *options request_ID*	バッチリクエスト識別子 *request_ID* をキャンセル

7.1 バッチリクエストの書き方

バッチリクエストはファイルとしてエディタにより作成します．ファイル名は任意です．ここでは“a.sh”という名前とします．以下，具体的な例にそって説明します．

```
#                          ←cshであることを指定
cd EXAMPLE                 ←ディレクトリの移動
frt main.f90               ←プログラムの翻訳
./a.out                    ←実行
```

- 先頭の#はジョブスクリプトをcshで記述することを指定します．この記事ではcshの記述にしたがっています．よくわからない方は，「おまじない」だと思って記述してください．また，mule, emacsで拡張子“.sh”を付けて新規にファイルを作成する場合，自動的に“#!/usr/bin/csh”が先頭に挿入されています．この記述でも問題ありません．
- 次はcdコマンドでディレクトリ“EXAMPLE”に移動しています．バッチリクエストの開始は利用者の**ホームディレクトリ**(ログイン時のディレクトリ)からになります．プログラムファイル，実行可能ファイルなどのあるディレクトリまで必ず移動するようにしてください．
- 以降の記述は対話型処理のコマンドと同様です．ここでは，“main.f90”を翻訳・結合編集し，実行しています．

7.1.1 バッチリクエストの記述例

1. 既に対話型処理などで作成済みの実行可能ファイル“a.out”を実行します．作業用ディレクトリは“mydir”です．

```
#
cd mydir
./a.out
```

2. Fortranプログラムを翻訳・結合編集し，実行します．装置番号5番からの入力ファイルとして“in.data”を指定します．

```
#
cd mydir
frt main.f90
./a.out < in.data
```

3. Fortranプログラムを翻訳・結合編集し，実行します．編集結合時にSSL II/VPを結合しています．装置番号1番，23番からの入出力ファイルに“inout1.data”，“inout2.data”をそれぞれ指定します．環境変数の設定は実行の前に行ってください．

```
#
cd mydir
frt main.f90 -lssl2vp
setenv fu01 inout1.data
setenv fu23 inout2.data
./a.out
```

4. 最適化オプション-O5を指定してFortranプログラムを翻訳・結合編集し，実行します．またその際timexコマンドによって経過時間，CPU時間を計測します．実行速度には影響ありません．

```
#
cd mydir
timex frt -O5 main.f90
timex ./a.out
```

5. VPP Fortranプログラムを翻訳・結合編集し，実行します．翻訳時オプション-Wxが必要です．

```
#
cd mydir
frt -Wx main.f90
./a.out
```

6. Cプログラム“main.c”をベクトル翻訳・結合編集し，実行します．標準入力ファイルとして“in.data”，標準出力ファイルとして“out.data”を指定します．

```
#
cd mydir
cc -Kvp main.c
./a.out < in.data > out.data
```

7. C++プログラム“main.C”をベクトル翻訳・結合編集し，実行します．

```
#
cd mydir
CC -Kvp main.C
./a.out
```

8. “#@$”に続けて，qsubコマンドのオプションを指定することができます．この例では，p8キューに投入することを指示する-q p8オプションを記述しています．また，VPP Fortranプログラムを翻訳するオプション“-Wx”を指定し，実行可能ファイル名を“b.out”に変更しています．

```
#
#@$-q p8
cd mydir
frt -Wx -o b.out main.f90
./b.out
```

9. あらかじめ作業を行うディレクトリに“a.out”というファイルがある場合，そのファイルを消去してから翻訳・編集結合し，実行します．この処理によって，翻訳が正常に終了しなかった場合に既存の実行可能ファイルが指定されてしまうという事態を回避します．

```
#
cd mydir
if ( -f a.out ) then
  rm -f a.out
endif
frt main.f95
./a.out
```

7.2 バッチリクエストの投入

バッチリクエストの投入はqsub(/usr/bin/qsub)コマンドによって行います．投入するジョブの規模に応じて以下のキュー名を選択します．

キュー名	記憶容量		演算時間	備考
	省略値	制限値		
s	2GB	7GB	1時間	非並列・短時間向き
p1	2GB	14.5GB	20時間	非並列(省略キュー)
s8	2GB/PE	7GB/PE	1時間	8PEまで使用可・短時間向き
p8	2GB/PE	7GB/PE	20時間	8PEまで使用可
p16	2GB/PE	7GB/PE	20時間	16PEまで使用可
p32	2GB/PE	7GB/PE	20時間	32PEまで使用可
x8	10GB/PE	15GB/PE	20時間	8PEまで使用可・大規模記憶容量向き
x16	10GB/PE	15GB/PE	20時間	16PEまで使用可・大規模記憶容量向き

7.2.1 p1キューに投入する例

ファイル“a.sh”に記述したバッチリクエストをqsubコマンドにより投入します．

```
kyu-vpp% qsub a.sh
Request 211.kyu-vpp submitted to queue: p1.
```

この例の“211”がリクエスト番号，“kyu-vpp”がホスト名です．これらをあわせた“211.kyu-vpp”を「バッチリクエスト識別子」と呼びます．バッチリクエスト識別子は，特にリクエストをキャンセルする時に必要となります．

7.2.2 p8キューに投入する例

-qに続けてキュー名を指定します．このオプションを省略した場合，p1キューに投入されます．

```
kyu-vpp% qsub -q p8 a.sh
```

7.2.3　p16キューに投入する例

以下の例ではエラーメッセージもファイルに出力させるため-eoオプションも併せて指定しています．qsubコマンドのオプションはバッチリクエストにも記述できます．

```
kyu-vpp% qsub -q p16 -eo a.sh
```

7.2.4　よく用いるqsubのオプション

詳細はman qsubで参照してください．

-eo	標準エラー出力を標準出力ファイルへ出力します．通常は別々のファイルに出力されます．
-q *que_name*	*que_name*キューにジョブを投入します．省略した場合p1キューに投入されます．
-lM *size*	*size*まで記憶容量を使用することを指定します．投入するバッチキューの制限値以下の値だけが有効です．たとえば10GBに設定する場合は“-lM 10gb”とします．
-lt *time*	CPU時間のリミットを*time*に設定します．バッチキューの制限値以下の値だけが有効です．たとえば1時間45分に設定する場合は“-lt 1:45:00”とします．
-me	実行終了をメールで通知します．
-mi	統計情報をメールで通知します．
-o *filename*	標準出力を*filename*というファイルに出力します．

7.2.5　バッチリクエストの返却

処理が終了すると，バッチリクエストファイル名とリクエスト番号に対応したファイルが返却されます．リクエスト番号の前に“o”のつく方に標準出力(たとえばFortranの装置番号6番)，“e”のつく方に標準エラー出力(システムの発行するメッセージなど)が書き出されています．例えば，ファイル名“a.sh”をqsubコマンドによって投入し，リクエスト番号が“211”であった場合，標準出力ファイルは“a.sh.o211”標準エラー出力ファイルは“a.sh.e211”という名前になります．**バッチリクエストファイル名が8文字以上の場合は，7文字で打ち切られて拡張子が付加されます．**

! 標準出力ファイルの先頭に

```
Warning: no access to tty (Bad file number).
Thus no job control in this shell.
```

というメッセージが付加される場合があります．これはエラーメッセージではありませんので無視して結構です．

7.3 バッチリクエストの状態表示

7.3.1 qstatコマンド

バッチキューの状態を表示するコマンドにqstat(/usr/bin/qstat)があります．バッチリクエスト識別子を確認する場合によく用います．

```
kyu-vpp% qstat          ←バッチリクエストの状態表示
s@kyu-vpp;  type=BATCH;  [ENABLED, INACTIVE];  pri=31
  0 exit;   1 run;   0 queued;   0 wait;   0 hold;   0 arrive;
p1@kyu-vpp;  type=BATCH;  [ENABLED, RUNNING];  pri=31
  0 exit;   2 run;   2 queued;   0 wait;   0 hold;   0 arrive;
        REQUEST NAME       REQUEST ID          USER PRI    STATE  JOB-ID  PHASE
<1 request RUNNING>
    2:           a.sh    213.kyu-vpp         a79999a  31  RUNNING     189  RUN
s8@kyu-vpp;  type=BATCH;  [ENABLED, INACTIVE];  pri=31
  2 exit;   1 run;   2 queued;   0 wait;   0 hold;   0 arrive;
    2:           c.sh    244.kyu-vpp         a79999a  31  RUNNING     189  RUN
                         :
```

7.3.2 qpsコマンド

投入したジョブの実行状況を調べるコマンドにqps(/usr/local/bin/qps)があります．
qpsと入力すると，実行中のリクエストの状態を表示します．

```
kyu-vpp% qps                ←リクエストの状態表示
que  user     request_ID       cpu        vu    vu/cpu cpu-limit  elapse  v-mem(MB)
p1 a79999a   211.kyu-vpp   0:00:02   0:00:00    0%  20:00:00   0:00:04  384/2048
p8 a79999a   218.kyu-vpp   1:32:11   1:23:23   90%  20:00:00   1:43:04 1984/2048
```

7.4 バッチリクエストのキャンセル

何らかの理由で投入したバッチリクエストをキャンセルするには，qdel(/usr/bin/qdel)コマンドを用います．キャンセルのためには，qstatコマンドによってバッチリクエスト識別子を確認しておく必要があります．

7.4.1 実行待ちのバッチリクエストのキャンセル

qdelに続けてバッチリクエスト識別子を指定します．

```
kyu-vpp% qdel 338.kyu-vpp       ←実行待ちのリクエストをキャンセル
Request 338.kyu-vpp has been deleted.
```

7.4.2 実行中のバッチリクエストのキャンセル

-kオプションを指定します．

```
kyu-vpp% qdel -k 338.kyu-vpp        ←実行中のリクエストをキャンセル
Request 338.kyu-vpp is running, and has been signalled.
```

7.5 kyu-ccからのバッチリクエスト

kyu-ccからスーパーコンピュータVPP5000/64にジョブを投入することができます．ただし，kyu-ccで作成したソースプログラム，データファイル，バッチリクエストファイルなどは必ずホームディレクトリにある“VPP”ディレクトリにコピーまたは移動してください．VPPディレクトリがkyu-vppのホームディレクトリとなります．

kyu-ccとkyu-vppではコマンド，オプションが異なる場合があります．また，kyu-ccで作成した実行可能ファイルをkyu-vppで実行することはできません．

7.5.1 バッチリクエストの投入

kyu-ccからkyu-vppのバッチリクエストを投入する場合には，-qに続けて，キュー名を指定します．p1キューに投入する場合にも省略することはできません．

```
kyu-cc% qsub -q p1 a.sh     ←p1キューに投入
Request 338.kyu-cc submitted to queue: p1.
```

7.5.2 バッチリクエストの状態表示

kyu-vppバッチキューの状態を表示するためには，@kyu-vppオプションを指定します．

```
kyu-cc% qstat @kyu-vpp   ←kyu-vppのバッチリクエストの状態表示
```

kyu-vppに投入したリクエストを調べるためにはqpsコマンドに続けて@kyu-vppを追加します．

```
kyu-cc% qps @kyu-vpp                ←kyu-vppのリクエストの状態表示
```

7.5.3 実行待ちのバッチリクエストのキャンセル

-r kyu-vppオプションを指定します．

```
kyu-cc% qdel -r kyu-vpp 3519.kyu-cc        ←実行待ちのリクエストをキャンセル
Request 3519.kyu-cc has been deleted.
```

7.5.4 実行中のバッチリクエストのキャンセル

-r kyu-vpp -kオプションを指定します．

```
kyu-cc% qdel -r kyu-vpp -k 3519.kyu-cc        ←実行中のリクエストをキャンセル
Request 3519.kyu-cc is running, and has been signalled.
```

8 VPP700/56からの移行

VPP5000/56は平成12年度末まで運用していたスーパーコンピュータVPP700/56の上位互換機であるため，VPP700/56の資産は基本的にそのまま利用できます．以下に移行に関しての注意点を列挙します．

8.1 実行可能ファイルの指定方法の変更

VPP700/56では，実行可能ファイル名を，例えば以下のように“a.out”と指定するだけでプログラムを実行することができました．

```
kyu-vpp% a.out              ←実行 (VPP700/56の場合)
```

しかし，セキュリティ強化のため，新kyu-vppの推奨する.cshrcではカレントディレクトリ(現在作業中のディレクトリ)へのパスを設定していません．実行する場合には必ず最初に“./”を指定してください．

```
kyu-vpp% ./a.out            ←実行 (VPP5000/64の場合)
```

a.outに先だって指定する“./”は，「現在作業しているディレクトリ(カレントディレクトリ)にあるファイルを対象にする」という意味です．また，バッチリクエストファイルも同様に修正が必要になります．

```
#                           ←バッチリクエストの記述例
cd mydir                    ←ディレクトリの変更
frt example.f90             ←プログラムの翻訳
./a.out                     ←実行 (./の指定)
```

8.2 VPP700のオブジェクトファイルの利用方法

VPP5000/64でFortran, C, C++プログラムを翻訳する場合には，省略値として64ビットアドレスモードにより処理が行われます．64ビットアドレスモードによって作成されたオブジェクトファイルと，32ビットモードであるVPP700/56で作成されたオブジェクトファイルとは**互換性がありません**．このため，次のようなエラーメッセージが出されることがあります．

```
kyu-vpp% frt main.f90 sub1.o    ← VPP700/56で作成したオブジェクトファイルとの結合編集
ld: sub1.o: fatal error: wrong machine class
```

オブジェクトファイルのアドレスモードは，file(/usr/bin/file)コマンドによって調べることができます．

```
kyu-vpp% file sub1.o        ←ファイル型の判定
sub1.o:       ELF 32 ビット MSB 再配置可能 VPP Version 1
```

VPP700/56で作成したオブジェクトファイルと結合するためには，frt, cc, CCコマンドのオプションとして-KA32を指定し，32ビットアドレスモードでの翻訳に切替えます．

```
kyu-vpp% frt -KA32 main.f90 sub1.o        ←32ビットアドレスモードでの翻訳・結合編集
```

ただし，32ビットアドレスモードで作成した実行可能ファイルは記憶容量の上限が2GBに制限されます．2GB以上の記憶容量を使用する場合には，VPP700/56で作成したソースファイルを省略値である64ビットアドレスモードで再コンパイルする必要があります．

8.3 Cコンパイラ起動コマンドの変更

VPP700/56ではCのスカラー翻訳コマンドがcc，ベクトル翻訳コマンドがvccでした．VPP5000/64ではccコマンドに統一され，“-Kvp”オプションを指定することによりベクトル翻訳が行われます．

8.4 Fortranの書式なし入出力文

Fortran 95仕様からは書式なし入出力を行う場合にOPEN文にFORM='UNFORMATTED'を指定していないとエラーとなります．

```
REAL(KIND=8),DIMENSION(100) :: X                    ! 配列の宣言
   :
OPEN(1,FILE='EXAMPLE.DATA',FORM='UNFORMATTED')      ! 書式なしデータを開く
READ(1) X                                           ! 書式なし入力
CLOSE(1)                                            ! ファイルを閉じる
```

8.5 Gaussian 98の環境設定

VPP700/56でGaussian 98を利用されていた方は，kyu-vppに接続する際に以下のメッセージが出力されることがあります．

```
/usr/local/gaussian98/g98/bsd/g98.login: Not a directory.
```

VPP5000/64では，Gaussian 98のレベルアップにともない環境設定の方法が変更になりました．お手数ですが，

http://www.cc.kyushu-u.ac.jp/scp/system/library/Gaussian/Gaussian98.html

を参考に利用者自身で設定変更をお願いします．

8.6 バッチジョブ終了後の使用額通知メールの廃止

これまでkyu-vppに投入したバッチジョブが終了すると，kyu-cc宛にジョブの使用額が送信されていました．VPP5000/64では，現在のところ，使用額通知機能が未サポートです．

現在の使用額はutlist(/usr/local/bin/utlistf)コマンドによって調べることができます．ただし集計が完了していない課金対象資源もありますのでutlistコマンドで表示される額は目安とお考えください．

8.7 ベクトル演算の丸めモード

VPP700/56のベクトル演算の丸めモードは「0方向丸めモード」でした．VPP5000/64のベクトル演算の丸めモードはIEEE754規格に準拠した「最近値丸めモード」です．このため，丸め誤差の影響を受けやすいプログラムでは実行結果に違いが生じる可能性があります．なお，スカラー演算ではVPP700/56, VPP5000/64ともに「最近値丸めモード」です．

8.8 実行結果に精度差が生じる可能性

上記浮動小数点演算の丸めモードの違いの他に，ハードウェアの違いによって，総和演算の演算順序，複合演算，組込み関数の性能がVPP700/56とVPP5000/64とでは異なります．そのため，実行結果に精度差の生じる可能性があります．

参考文献

[1] Ellis, M. A. and Stroustrup, B.: The Annotated C++ Reference Manual, Addison Wesley (1990). (足立高徳, 小川祐司訳: 注解 C++ リファレンスマニュアル, トッパン)

[2] Kernighan, B. W. and Ritchie, D. W.: The C Programming Language, Prentice Hall (1998). (石田晴久訳: プログラミング言語 C, 共立出版)

[3] 九州大学情報基盤センター スーパーコンピュータ オンラインマニュアル [‡], http://www.cc.kyushu-u.ac.jp/scp/system/manual/UXPV_MANUAL/

[4] 南里豪志: 情報基盤センター・研究者用システムでのUNIX利用法, UNIX初級講習会資料 (2001).

- PDFファイル (449,738 bytes) " unixguide.pdf "
  http://spring.cc.kyushu-u.ac.jp/scp/system/library/UNIX/unixguide.pdf
- PostScriptファイル (600dpi; 1,030,609 bytes) " unixguide.ps "
  http://www.cc.kyushu-u.ac.jp/scp/system/library/UNIX/guide1.ps
- PostScriptファイル・gzip圧縮 (600dpi; 196,824 bytes) " unixguide.ps.gz "
  http://www.cc.kyushu-u.ac.jp/scp/system/library/UNIX/guide1.ps.gz

[5] 南里豪志: 新スーパーコンピュータシステムの紹介, 九州大学情報基盤センター広報, Vol. 1, No. 1, pp.20–30 (2001).

[‡]本センターの利用資格を有している方しか参照できません．登録番号とパスワードを尋ねてきますので，登録番号のみを入力してください．パスワードは空白で構いません．

たかが内積、されど内積

藤野清次*

1　はじめに

1.1　積和 (内積) 計算

科学技術計算の分野では次のような計算がよく出てきます。ここで、a_i, b_i はデータで N はその個数です。

$$S = \sum_{i=1}^{N} a_i b_i \tag{1.1}$$

一般に、上式のような計算は積和計算と呼ばれています。つまり、$S = a_1b_1 + a_2b_2 + \cdots + a_Nb_N$ というように2つのデータの積の合計 (和) の値を S に入れることを意味します。また、これはベクトル $\boldsymbol{a}$, $\boldsymbol{b}$ に対する内積 $(\boldsymbol{a}, \boldsymbol{b})$ の定義と同じです。したがって、内積あるいは内積演算とも呼ばれます。$a_1, a_2, \cdots, a_N$ と $b_1, b_2, \cdots, b_N$ は各々ベクトル $\boldsymbol{a}$ と $\boldsymbol{b}$ の要素になります。積和を求めるプログラムは簡単で、例えば以下のようになります。左がFortran90風に書いたもの、右がC言語風に書いたものです。"たかが積和" です。"たかが内積" です。

```
S=0.0
DO K=1,N
S=S+A(I)*B(I)
END DO
```

```
s=0.0;
for (i=0; i<N; i++){
 s+=a[i]*b[i];
}
```

1.2　積和 (内積) 計算の事例

では、上のような積和あるいは内積計算は科学技術計算のどんなところで出てくるのでしょう。ここでは代表的な2つの事例を紹介しましょう。

1.2.1　事例その1(行列積の計算)

$N \times N$ の正方行列 $A = (a_{ij})$ と $B = (b_{ij})$ を考えます。そして、行列 $A = (a_{ij})$ と行列 $B = (b_{ij})$ の積を求めることにします。ここでは、その結果を正方行列 $C = (c_{ij})$ に代入します。このような計算は、一般に行列と行列の積の計算あるいは略して行列積の計算と呼ばれています。すなわち、行列積の計算は以下のように表せます。

$$c_{ij} = \sum_{k=1}^{N} a_{ik} b_{kj} \quad (i, j = 1, 2, \cdots, N) \tag{1.2}$$

*九州大学情報基盤センター研究部スーパーコンピュティング研究部門
E-mail:fujino@cc.kyushu-u.ac.jp

また、行列積の計算のプログラムは、積和のときと同様に、以下のように書けます。

```
DO J=1,N                          for(i=0; i<N; i++)
DO I=1,N                          for(j=0; j<N; j++){
S=0.0                                 s=0.0;
DO K=1,N                              for (k=0; k<N; k++)
S=S+A(I,K)*B(K,J)                         s+=a[i][k]*b[k][j];
END DO                                c[i][j]=s;
C(I,J)=S                          }
END DO
```

このような行列積の計算は、連立1次方程式のブロック三角分解をするときに現れます。また、固有値解析のハウスホルダー変換などでも現れます。あるいは、いろいろな計算機の性能評価を行うときにもこれは利用されます。演算のパターンが単純で個数が一定であるため計算機が持つ最高演算性能が発揮され易いためと思われます。すなわち、行列積の計算は線形計算あるいは性能評価の基本計算といって過言ではありません。でも、まだ“たかが行列積の計算”でしょう。恐るに足りずです。次にもう一つ事例を紹介しましょう。

1.2.2 事例その2(共役勾配法と内積計算)

次の連立1次方程式を反復解法で解くことを考えます。

$$A\boldsymbol{x} = \boldsymbol{b}. \tag{1.3}$$

ここで、係数行列 A は非対称疎行列で大きさは $n \times n$ とします。$\boldsymbol{b}$ と $\boldsymbol{x}$ は n 次元の右辺ベクトルと解ベクトルとします。疎行列とは、行列要素が零である割合が大きい行列で、有限要素法や有限差分法で離散化した方程式は多くの場合このような疎行列になります。また、このような連立1次方程式は、その規模が大きくなればなるほど計算時間の面あるいは必要なメモリー容量の面から一般に反復解法で解かれます。特に、行列 A が対称正定値行列の場合は、適当な前処理を係数行列に施した後、共役勾配 (Conjugate Gradient, CG と略す) 法を適用するのが最も有効です。共役勾配法は“きょうやくこうばいほう”と呼びます。この前処理付き共役勾配法は連立1次方程式の解法として非常に強力で、いろいろな応用分野の計算ですでに定評があります。以下に共役勾配法のアルゴリズムを示します。ただし、ε は収束判定のための微小な値です。

アルゴリズム 1　CG法

$\boldsymbol{x}_0$ *is an initial guess,* $\boldsymbol{r}_0 = \boldsymbol{b} - A\boldsymbol{x}_0$, *set* $\beta_{-1} = 0$,
for $n = 0, 1, \cdots$ *until* $\| \boldsymbol{r}_n \| \leq \varepsilon \| \boldsymbol{b} \|$ do :
begin
$\boldsymbol{p}_n = \boldsymbol{r}_n + \beta_{n-1}\boldsymbol{p}_{n-1}$,

$$\alpha_n = \frac{(\boldsymbol{r}_n, \boldsymbol{r}_n)}{(\boldsymbol{p}_n, A\boldsymbol{p}_n)},$$
$$\boldsymbol{x}_{n+1} = \boldsymbol{x}_n + \alpha_n \boldsymbol{p}_n,$$
$$\boldsymbol{r}_{n+1} = \boldsymbol{r}_n - \alpha_n A\boldsymbol{p}_n,$$
$$\beta_n = \frac{(\boldsymbol{r}_{n+1}, \boldsymbol{r}_{n+1})}{(\boldsymbol{r}_n, \boldsymbol{r}_n)},$$
end.

ここで、α_n, β_n はパラメータ (=定数) で、反復回数 n ごとに設定されます。これらのパラメータは、CG 法のアルゴリズムの中で、近似解 x_n を真の解 x に近づけさせるための重要な働きをしています。そして、アルゴリズムからわかるように、α_n, β_n の計算はまさに内積計算そのものなのです。しかも、内積の要素数は解くべき連立一次方程式の次元数になるので、一般に非常に大きな数です。その数は100万どころか最近では1億を超える計算事例も報告されるようになってきました。

CG 法の詳細はここでは省略しますが (例えば、文献 [12] 参照)、来年2002年は CG 法が Hestenes と Stiefel によって1952年 [4] に発表されてちょうど50年目にあたります。そこで、それを記念する会議† が2月に計画されています。このように CG 法は発表から半世紀を経ても研究者の探求心を離さないとても魅力的な解法の一つです。

1.2.3　試しに簡単な総和計算をやってみましょう

積和計算の前にもっと簡単な総和計算: $S=\sum_{i=1}^{N} a_i$ をやってみましょう。次の計算を手元の PC (FMV-Biblo NE5, 600MHz) でやらせてみました。項は $a_i=1/(i(i+1))$ です。

$$S = \sum_{i=1}^{N} \frac{1}{i(i+1)} \quad = \quad \frac{N}{N+1} \tag{1.4}$$

正解は右側に記した $N/(N+1)$ です。計算値、正解、誤差を表1に示します。

もちろん、計算は全て倍精度演算で行いました。表から、項数を増やしていくとあるところから誤差が急に大きくなる (積み残しが始まった!) こと、そして100万項では誤差が 10^{-6} にも達していることがわかります。よく見ると、項数が50万を超えたところから (表中の罫線参照) 計算値が変わらなくなっています。大規模な計算の場合この結果では少し心配ですね。"たかが内積" だったはずですが、雲行きが怪しくなってきました。

1.2.4　情報落ちと Kahan の補償アルゴリズム

この現象を少しまとめておきましょう。一般に、N 個のデータ a_i の総和計算: $S = \sum_{i=1}^{N} a_i$ をするとき、項数が多いと総和 S の値と各項 a_i の値のオーダーが大きく異なってきて、その結果、総和 S に a_i の有用な情報が反映せず、いわゆる "情報落ち"、あるいは "情報の積み残し" という現象が現れる場合があります。今回の現象もこれに当たります。このとき、表1に示したように求めた総和 S の精度は著しく悪くなります。

†A CONFERENCE COMMEMORATING: "50 YEARS OF CONJUGATE GRADIENTS", ETH Zurich, Switzerland, 18-20 February 2002. http://www.sam.math.ethz.ch/~ mhg/CG50/

表 1: 簡単な総和計算の数値実験結果

項数 N	計算値	正解	誤差
10 万	0.9999900001000118	0.9999900000999991	0.128E-13
20 万	0.9999950000250054	0.9999950000249999	0.555E-14
30 万	0.9999966666778231	0.9999966666777778	0.453E-13
40 万	0.9999975000062854	0.9999975000062500	0.354E-13
50 万	0.9999980000040377	0.9999980000040000	0.377E-13
60 万	0.9999980000040377	0.9999983333361111	0.333E-06
70 万	0.9999980000040377	0.9999985714306122	0.571E-06
80 万	0.9999980000040377	0.9999987500015625	0.750E-06
90 万	0.9999980000040377	0.9999988888901234	0.889E-06
100 万	0.9999980000040377	0.9999990000009999	0.100E-05

では、このような現象を回避することはできないのでしょうか? この例では、項 a_i が単調に小さくなる性質を積極的に生して、後向きに計算する、すなわち第 N 項、$(N-1)$ 項、$(N-2)$ 項 $\cdots$ と第 1 項までを計算すれば誤差の発生を抑えることができます (詳しくは文献 [3] を参照)。しかし、あらかじめそのような項の大きさに関する情報などがないとき、例えばCG 法のようにアルゴリズム中に現れる内積計算のような場合はもうお手上げです。

単に加算するだけではダメで、欠落した情報をもっと陽的に補償する新しいアルゴリズムが必要です。それを解決するアルゴリズムの一つに **Kahan** が提唱した総和に対する補償アルゴリズムがあります [2, 3, 5, 7]。ここで、文献 [7] は原著論文、文献 [2] には誤差評価がされています。文献 [5] には最近の研究がきれいに整理されています。文献 [3] にはわかりやすく解説がなされています。

また、このような補償付きアルゴリズムは、この外にも常微分方程式の数値解法における Möller 法 [11] や Runge-Kutta-Gill の方法 [1] が知られています。ご存じの方もおられると思います [6]。次に、Kahan[‡] の補償アルゴリズムの考え方を積和計算にも生かせないかを考えてみましょう。ちょっと、その前に Tea Time をとりましょう。

Tea Time

ここで、数学用語の内積: Inner product の語源について簡単に紹介しておきましょう。これは、Hermann Grassmann (1809.4.15 Prussia - 1877.9.26 Stettin) による著書 "Die lineale Ausdehnungslehre" (1844 年刊) の中に出てきたドイツ語の inneres produkt の訳語から生まれました。関係する部分の英訳を以下に示します (Oxford 大辞典より)。

"the sum of the products of corresponding components of two real vectors $(a_1, a_2, \cdots, a_n)$ and $(b_1, b_2, \cdots, b_n)$, i.e. the number $a_1b_1 + a_2b_2 + \cdots + a_nb_n$; in a complex vector space, "

[‡]William Kahan: California 大学 Berkeley 校教授、1989 年 IEEE 浮動小数演算標準 754 と 854 に対する功績で ACM Turing 賞受賞 [8]。この他にも ACM, IEEE, SIAM などの多数の賞を受賞。

2　補償付き積和計算

積和計算 $S = \sum_{i=1}^{N} a_i b_i$ に対する通常の計算法 (以下、この方法を “定義通りの積和計算” と呼ぶ) によるプログラムの一例を図 1(a) に、同じく補償付きの積和計算 (以下、“補償付きの積和計算” と呼ぶ) のプログラムの一例を図 1(b) に示します。

```
S=0.0
DO I=1,N
S=S+A(I)*B(I)
END DO
```

(a) 定義通りの積和計算

```
      S=A(1)*B(1)
      C=0.0
      DO I=2,N
[1L]  Y=A(I)*B(I)-C
[2L]  T=S+Y
[3L]  C=(T-S)-Y
[4L]  S=T
      END DO
```

(b) 補償付きの積和計算

図 1: 2 つの積和計算法のプログラムの一例

補償付き積和計算における補償過程の仕組みと図 1(b) 中の [2L] 行と [3L] 行との対応関係を図 2 に示します。ここで、Y の上位桁を Y_{high}、下位の桁を Y_{low} で表しています。いま $|S| \geq |Y|$ と仮定します。すると、S+Y の加算において浮動小数演算の結果失われた Y の下位の桁の情報 $-Y_{low}$ が一時的変数 C に入り、その符号を変えたものが次のステップの計算で S に加えられ積和計算の精度が補償される、という仕組みを図 2 は示しています。元になったのは総和に対する Kahan の補償付きアルゴリズムです。

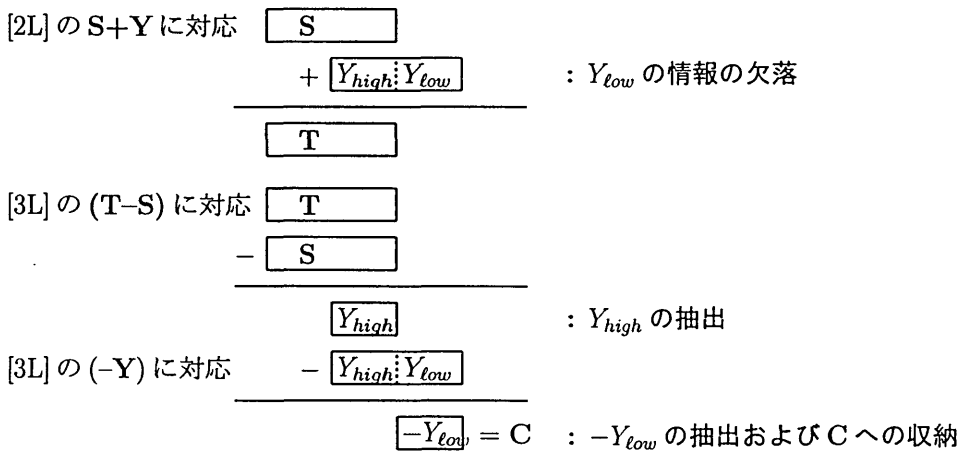


図 2: 補償付き内積演算における補償過程の模式図と図 1(b) 中の処理行との対応。Y_{high} は Y の上位の桁, Y_{low} は Y の下位の桁を表す。

2.1 テスト問題

ここでは、以下に示す2つの問題 [10] に対して通常の積和計算および補償付き積和計算の誤差を以下の3つのケースについて調べました。

1. 問題 (2.1) において項数 N が50万項の場合。この問題は、(1.4) 式で項 $a_i = 1/(i(i+1))$ を求める問題と同じですが、今度は項 $a_i = 1/i$ と項 $b_i = 1/(i+1)$ との積の合計を求める問題とみなしています。Kahan の補償アルゴリズムの応用を考えた結果です。
2. 問題 (2.2) において項数 N が 50 万項の場合。ただし、$x = 0.5$ とする。
3. さらに、問題 (2.2) では、x の値として区間 $(0, \pi/2)$ において等間隔に 25 分割した値をとり、項数 N については 5 万個から 50 万個まで 5 万個ずつ増やして全部で 10 通り、これらの合計 250 通りの場合について 2 つの積和計算法の誤差の平均値を求めました。また、各々250 通り、合計 500 通りの場合の分布の様子や傾向を見るために、2 次元の配列に誤差をセットし、数学ソフトウエア *Mathematica* を使って誤差の 3 次元の可視化を試みました。

$$\sum_{i=1}^{N} \frac{1}{i} \times \frac{1}{i+1} = \frac{N}{N+1} \tag{2.1}$$

$$\sum_{i=1}^{N} \sin ix \times \sin(i+2)x = \frac{N}{2}\cos 2x - \frac{\cos(N+3)x \sin Nx}{2\sin x} \tag{2.2}$$

2.2 補償付き積和計算に対する数値実験

数値実験は九州大学情報基盤センターおよび表 3 に示す各センターに設置された合計 7 種類の計算機上で行いました。計算機の主な仕様を表 2 と表 3 に示します。ただし、計算には 1PE(Processing Element) のみを使用。使用言語は Fortran90 を使用しました。コンパイラーコマンドは表 4 に示すものを使用しました。また、最適化オプションは計算順序の変更による副作用 (誤差の増減) の影響を完全に取り除くために今回は極力使用しませんでした。計算はすべて倍精度演算で行いました。なお、倍精度演算での計算機イプシロンは約 2.22×10^{-16} です。測定は、2001 年 5 月 23–29 日の間に実施しました。

表 5 に問題 1 に対する結果を、同じく表 6 に問題 2 に対する結果を計算機毎に示します。

これらの 2 つの表から、補償付き積和計算の方が定義通りに積和計算を行うよりも誤差がはるかに少ないことがわかります。特に表 5 の場合には補償付き積和計算での誤差はまったく発生していません。

さらに、表 7 に、問題 2 の 250 ケースに対する補償付き (図 4 以降では Compensated と表示) 積和計算法および定義通り (図 4 以降では Definition と表示) の積和計算の各誤差の平均値を示します。表に示す結果から、番号に*印を付けたもの (No.1, 2, 3, 5) とそうでないもの (No.4, 6, 7, 誤差は太字で表示) の 2 つの場合に大きく分けられることがわかります。前者では、補償付き積和計算の誤差が少ないのに比べて定義通りの積和計算の結果は誤差が非常に大きいことが見られます。一方、後者では、補償付き積和計算の誤差も定義通りの積和計算のそれも同程度に小さいことがわかります。また、少しですが補償付き積和計算の誤差の方が小さいことも見てとれます。

表 2: 数値実験で使用した計算機の主な仕様 (その 1)

	FMV715GTX6	GS320	GP7000F
Vendor	Fujitsu	Compaq	Fujitsu
CPU	Pentium4	Alpha21264	SPARC64
Clock	1.5 GHz	0.731 GHz	0.3 GHz
Memory	0.64GB	2GB/PE	1GB/PE
OS	Linux2.2.18 (VineLinux 2.1.5)	Tru64UNIX	Solaris7

表 3: 数値実験で使用した計算機の主な仕様 (その 2)

	VPP700E/160	SR8000/MPP	VPP5000/64	S-X5/128M8
Vendor	Fujitsu	Hitachi	Fujitsu	NEC
設置場所	理化学研究所 情報環境室	東京大学 情報基盤センタ	九州大学 情報基盤センタ	大阪大学サイバーメディアセンタ
最大性能 (Gflops)	2.4/PE	1.8/PE	9.6/PE	10.0/PE
PE 数	160	1152	64	128

表 4: Fortran90 で書かれたプログラムのコンパイラーコマンドについて

計算機	コマンド
FMV715GTX6	f90
GS320	f90
GP7000F	frt
VPP700E/160	frt -Wv,-md
SR8000/MPP	f90
VPP5000/64	frt
SX-5/128M8	sxf90

表 5: 問題 1 に対する補償付き積和計算および定義通りの積和計算の誤差 (N=50 万個のとき、真の値=0.9999980000040000)

No.	計算機	補償付き	定義通り
1	FMV715GTX6	0.0	0.390 E-13
2	GP7000F	0.0	0.390 E-13
3	GS320	0.0	0.390 E-13
4	VPP700E/160	0.0	0.999 E-15
5	SR8000/MPP	0.0	0.390 E-13
6	VPP5000/64	0.0	0.333 E-15
7	SX-5/128M8	0.0	0.222 E-15

表 6: 問題 2 に対する補償付き積和計算および定義通りの積和計算の誤差 (N=50 万個、$x = 0.5$ のとき、真の値=.1350766019190951d+06)

No.	計算機	補償付き	定義通り
1	FMV715GTX6	0.0	0.192 E-08
2	GP7000F	0.0	0.195 E-08
3	GS320	0.0	0.195 E-08
4	VPP700E/160	0.291 E-10	0.203 E-09
5	SR8000/MPP	0.0	0.195 E-08
6	VPP5000/64	0.0	0.0
7	SX-5/128M8	0.291 E-10	0.291 E-10

表 7: 問題 2 の 250 ケースに対する補償付き積和計算および定義通りの積和計算の各誤差の平均値

No.	計算機	補償付き	定義通り
1*	FMV715GTX6	7.478 E-11	1.290 E-08
2*	GP7000F	7.395 E-11	1.289 E-08
3*	GS320	7.424 E-11	1.291 E-08
4	VPP700E/160	**1.119 E-10**	**2.276 E-10**
5*	SR8000/MPP	6.962 E-11	1.290 E-08
6	VPP5000/64	**1.057 E-10**	**1.273 E-10**
7	SX-5/128M8	**1.181 E-10**	**1.733 E-10**

2.3　2つの積和計算法の誤差の可視化

図3に可視化データの配置を示します。水平方向はデータ数(単位：万個)を表します。図3にあるように、下半分には補償付きの積和計算のときの誤差を、上半分には定義通りの積和計算のときの誤差を収めます。また、図4にデータのカラー指標を示します。高さ方向は計算誤差の大きさを表します。データの値は10^{-12}から10^{-6}までの範囲にありましたので、目盛は常用対数表示で-12から-6までです。また、この3次元表示の図では、手前半分が補償付きの計算誤差のデータにあたり、奥半分が定義通りの計算誤差のデータにあたります。図中のx_{min}とx_{max}は、式(2.2)の中のxの値が$x = \pi/50$と$x = \pi/2 - x_{min}$を各々表します。また、3次元の立体図を視る方向を矢印で表します。

図 3: 可視化データの配置について

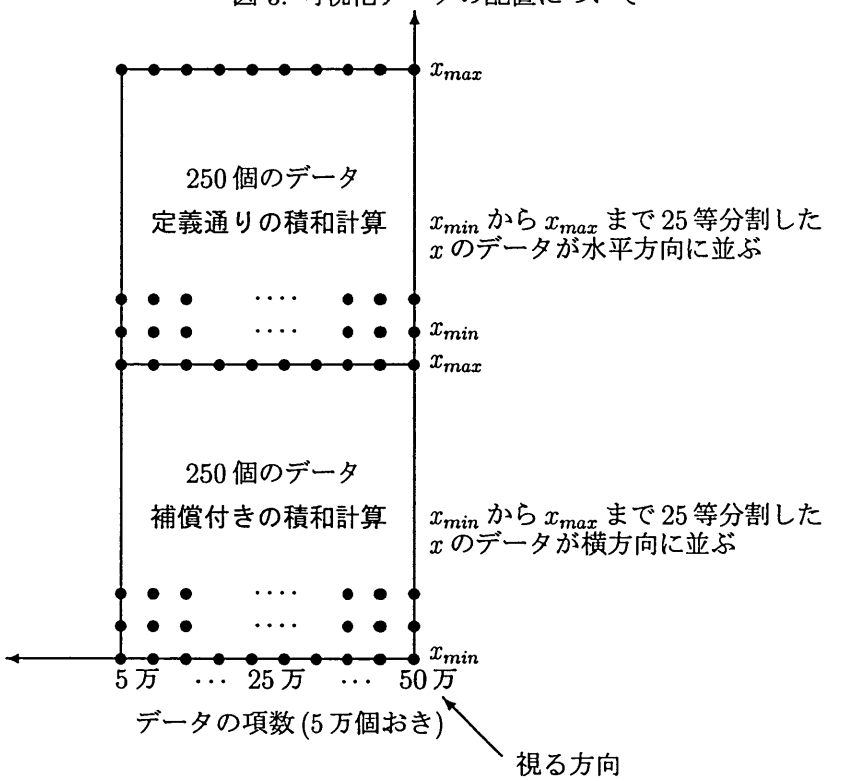


図5にFMV715GTX6での二つの積和計算法の誤差分布を、図6にGP7000Fでの同結果を、図7にGS320での同結果を、図8にVPP700E/160での同結果を、図9にSR8000/MPPでの同結果を、図10にVPP5000/64での同結果を、そして最後に図11にSX-5/128M8での同結果を各々示します。表7の結果のところで、誤差の傾向が二つに分かれることを言いましたが、可視化するとこの傾向が一層鮮明におわかりになると思います。黄緑色や空色のデータは誤差が相対的に小さいことを表し、反対にピンク色や赤紫色のデータは誤差が大きいことが表しています。

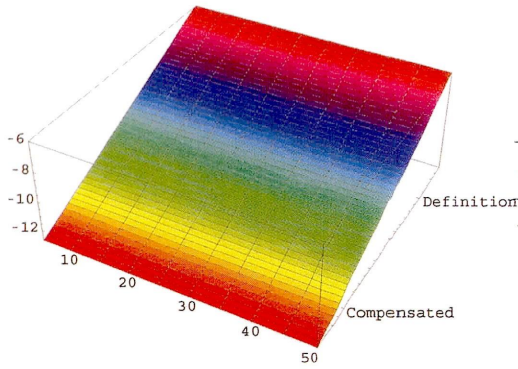


図 4: カラー指標 (水平方向：データ数 (単位：万個)、手前のデータは補償付きの計算誤差、後方は定義通りの計算誤差を表す)

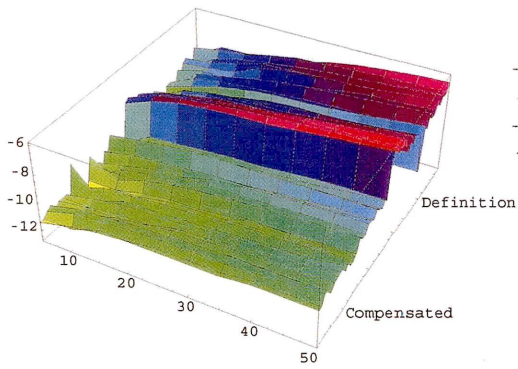


図 5: FMV715GTX6

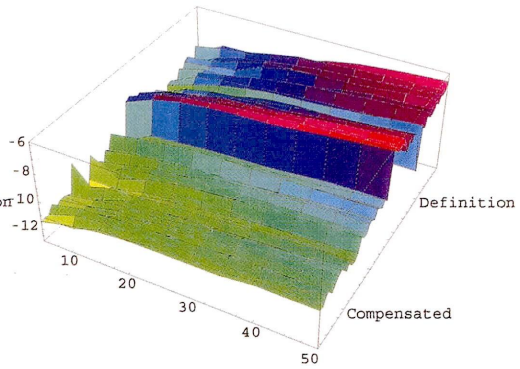


図 6: GP7000F

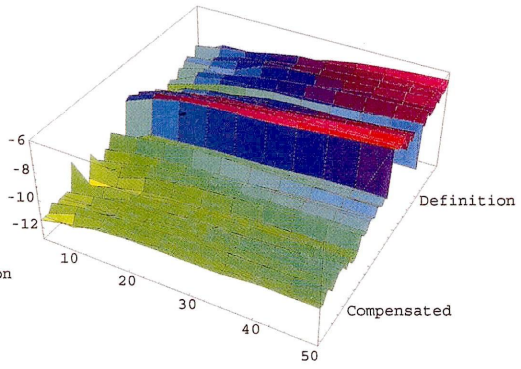


図 7: GS320

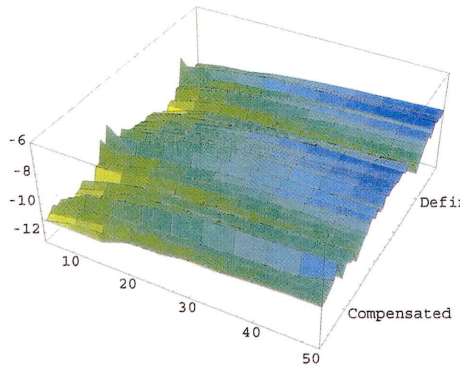


図 8: VPP700E/160

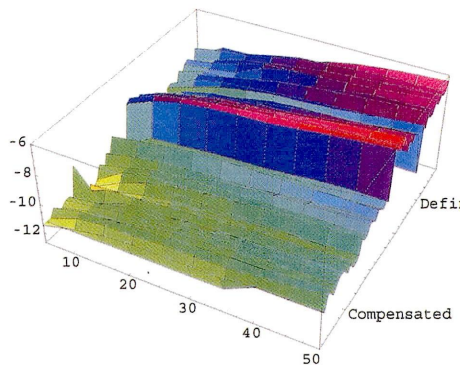


図 9: SR8000/MPP

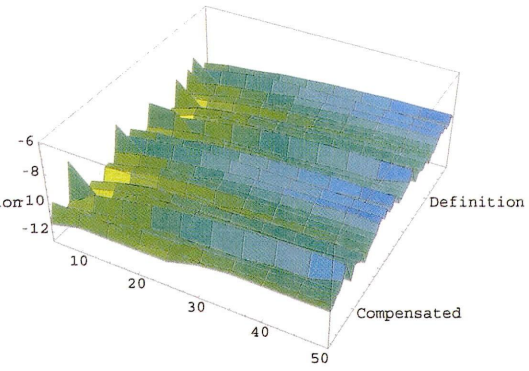


図 10: VPP5000/64

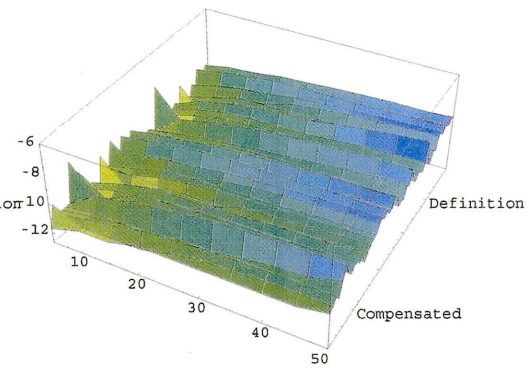


図 11: SX-5/128M8

定義通りの積和の計算で、計算機によってこのように誤差の大きさに違いが出る訳は、積和の計算のときにその結果をそのまま演算器に保持するのか、あるいはレジスターなどに一旦移すのかの違いによって生まれるものと思われます。一方、補償付きの積和の計算では、誤差が非常に少なくなることは今まで述べてきた通りです。そのために余分にかかる計算時間は、手元にあるデータではおよそ 20% ～ 30% 程度です。

3 おわりに

表題につけた "たかが内積、されど内積" の気持がわかっていただけましたでしょうか。補償付き積和計算法、特に CG 法などに現れる内積計算へのそれの適用については大変興味深い上、いろいろな応用分野の計算への波及効果も大きいと思われますが、それは後日報告する機会に譲りたいと思います。

謝辞

本記事を執筆するに当たり、数値実験などにご協力をいただきさらに有用なご助言を賜わった理化学研究所 阿部邦美博士、本学情報基盤センター三浦謙一客員教授、渡部善隆助教授、南里豪志助教授に心より感謝の意を表します。

参考文献

[1] Gill, S., A process for the step-by-step integration of differential equations in an automatic digital computing machine, *Proc. Cambridge Phil. Soc.*, 47(1951), 96–108.

[2] Goldberg, D., What every computer scientist should know about floating-point arithmetic, *ACM Computing Surveys*, 23(1991), 5–48.

[3] 長谷川武光, 数値計算のつぼ (1)　$0.1 \times 10 = 1$?, 名古屋大学大型計算機センター ニュース, 31(2000), 271–280.

[4] Hestenes, M. R., Stiefel, E., Methods of Conjugate Gradients for Solving Linear Systems, J. Res. Nat. Bur. Standards, 49(1952), 409–435.

[5] Higham, N., Accuracy and Stability of Numerical Algorithms , *SIAM*, Philadelphia, 1996.

[6] 伊理正夫, 松谷泰行, Runge-Kutta-Gill 法について, 情報処理, 8(1967), 103–107.

[7] Kahan, W., Further remarks on reducing truncation errors, *Comm. ACM*, 8(1965), p.40.

[8] Kahan, W., IEEE 754 – Interview, IEEE Computer, 31(1998), 114–115.

[9] Knuth, D., The Art of Computer Programming, Volume 2, Seminumerical Algorithms. 3rd edition, Addison-Wesley, Reading, 1998.

[10] 森口繁一, 宇田川欽久, 一松信, 岩波 数学公式 II 級数, 岩波書店, 東京, 1988.

[11] Müller, O., Note on quasi double-precision, *BIT*, 5(1965), 251-255.

[12] 名取亮, すうがくぶっくす 線形計算, 朝倉書店, 東京, 1993.

プログラム開発のための補助ツール(1)

—rsync—

池田大輔*

1　はじめに

情報基盤センター(以下，センターと呼ぶ．)の利用者は，通常は所属する研究室のパソコンやワークステーションを利用し，大規模な科学技術計算を行なう必要があるときに，センターのシステムを利用するという方が多いと思います．研究室の計算機環境とセンターの計算機環境とは違う場合がほとんどで，利用方法に戸惑うことも多いと思います．特にセンターのシステムでは，利用に対し課金が発生するので，効率のよい利用が望まれます．

本連載では，センターのシステムのみの利用方法を説明するのではなく，通常は研究室のパソコンやワークステーションを利用する利用者が，効率よくセンターのシステムを使うための方法について解説していく予定です．

研究室の計算機環境はUNIX系のOSが動いているものと仮定します．また，研究室の計算機環境にもFortranやCのコンパイラがあり，ソースプログラムを共有すると仮定します．つまり，利用者は，大規模な計算機実験やアプリケーションソフトウェアを用いた実験等はセンターのシステムで行なうけれども，予備的な実験や最終的な結果の生成は研究室の計算機で行なうものとします．

第1回は，異なる計算機間でファイルのコピーをしたり，ディレクトリの中身を同一に保つ(同期をとる)ためのコマンドrsyncの使い方を紹介します．

2　rsyncとは

rsyncは，異なる計算機間でファイル複製やディレクトリの同期をとるためのコマンドです．大規模科学技術計算はセンターのシステムで行なうとはいえ，小規模な予備計算等はパソコンでも充分できるかもしれません．その場合，ソースコードや実験用入力データ等は研究室の計算機とセンターの計算機で同一で構いません．また，ソースコードに大きく変更を加える場合は使い慣れた自分の計算機で編集作業を行ない，それからセンターの計算機へftpなどで転送することもあるでしょう．このような場合rsyncは，安全かつ高速なディレクトリの同期を実現してくれます．

rsyncの公式Webページはhttp://rsync.samba.org/です．このページでは，マニュアル，使い方の紹介やFAQのドキュメント類が英語で提供されています．ソースコードのダウンロー

*情報基盤センター研究部　mailto:daisuke@cc.kyushu-u.ac.jp

ドページもありますが，ネットワーク的な近さを考えるとRing Server[1]の/pub/net/rsyncから取得するとよいでしょう．

2.1 準備

基本的にrsyncは異なる計算機間で使用するものです．以下，利用者の手元にある計算機を“local”とし，遠隔の計算機を“remote”と表記します．センターの利用者であれば，研究室の計算機が“local”で，センターの計算機が“remote”になります．

rsyncを“local”と“remote”間で利用するには，どちらの計算機にも自分のアカウントが用意されている，かつ，どちらの計算機にもrsyncがインストールされていることが必要です．インストールにはroot権限は不要で，パスの通った場所であればどこにインストールしても構いません．

さらに，安全なデータの転送を行なうためには“remote”上でsshサーバが動いていて，“local”上にsshクライアントがインストールされている必要があります．sshのインストール等については文献[1, 2]を参照してください．

現在，センターの計算機としては“kyu-cc”，“kyu-ss”，“wisdom”等にrsyncがインストールされています．また，サービスを提供している全ての計算機上でsshサーバが稼働しています．

sshを利用する場合も，そうでない場合も，初めてrsyncをremoteとの間で利用する場合には準備が必要です．これについては5節で説明します．

3 基本的な使い方

rsyncはcpと同様に，“どのファイル”を“どこへ”コピーするかを指定します．

```
local% rsync file1 file2↵
local% rsync file1 file2 dir↵
```

どちらの形式も同じ計算機間でのコピーですので，通常のcpでも同様に実行できます．最初の形式ではfile1の内容をfile2という名前のファイルにコピーします．次の形式ではfile1, file2という二つのファイルをdirというディレクトリの下にコピーします．

異なる計算機間の場合はファイル名の前に“ホスト名”と“:”をつけます．この場合は“remote”側で多少の準備が必要です(5節を参照)．準備ができたら，以下のように実行できます．

```
local% rsync remote:file1 file2↵
local% rsync file1 remote:file2↵
local% rsync remote:file1 remote:file2 dir↵
local% rsync file1 file2 remote:dir↵
```

[1] http://www.ring.gr.jp/

これらはrcpやscpでも同様に実行することができます.

"remote" 側のファイル指定は絶対パスかホームディレクトリからの相対パスで記述してください. 例えば

```
local% rsync remote:Mail/inbox/1 .↵
```

とすると$HOME/Mail/inboxにある1というファイルを, "local" 上のカレントディレクトリにコピーします.

"local" と "remote" で, ユーザ名が異なる場合は, remoteのユーザ名を "@" の前に指定して

```
local% rsync user@remote:file1 file2↵
```

と実行します.

ここまでは, コピー元が個々のファイルでしたが, 以下のようにディレクトリを指定することもできます.

```
local% rsync -a remote:dir1 dir2↵
local% rsync -a dir1 remote:dir2↵
```

この-aオプションによりディレクトリの中身が再帰的に転送されます. この場合rsyncはファイルの所有者, 変更日時等の情報をできる限り保持します.

コピー元にディレクトリを指定する場合, 重要な注意があります. コピー元であるdir1の最後に "/" があるかないかで, rsyncは挙動を変えます. "/" が最後にない場合はdir1というディレクトリ自体がコピー先ディレクトリにコピーされます. 例えばremote:srcにA, B, Cというファイルがあるとします. このとき, コピー元の最後に "/" をつけずに実行すると

```
local% rsync -a remote:src .↵
local% ls↵
src
local% ls src↵
A B C
```

となり "local" のカレントディレクトリにsrcというサブディレクトリが作られ, ファイルA, B, Cはこのサブディレクトリの下にコピーされます.

一方 "/" が最後にある場合はsrc自身は転送されず, src以下のファイルのみを転送します.

```
local% rsync -a remote:src/ .↵
local% ls↵
A B C
```

この場合srcというサブディレクトリは作られません.

4 ファイル同期

ここまでの方法は，コピー元として与えた全てのファイル(ディレクトリの場合は再帰的に)をコピーします．したがってrsyncでなくともtarやrcpで充分です．また，セキュリティの面を考えてもscpやsftpで充分です．

しかし，多くのファイルからなるディレクトリであっても，転送すべきファイルは少数かもしれません．更新されたものだけ転送するためのオプションが-uオプションです．さらに--deleteオプションによりコピー元のディレクトリにないファイルを削除しますので，同期をとることができます．

同期をとる場合は前述したコピー元の最後の"/"と，コピー先のディレクトリの指定に気をつける必要があります．例えば"local"上の$HOME/workディレクトリと"remote"上の$HOME/Program/workディレクトリの内容を同一にすることを考えてみましょう．"remote"側が最新の情報を保持しているとします．この場合，

```
local% cd↵                                          ←workがある場所へ移動
local% rsync -au --delete remote:Program/work .↵
```

とします．コピー元の最後に"/"をつけていませんので，ディレクトリworkそのものが転送されます．したがって"local"側はworkがあるホームディレクトリに移動しておく必要があります．

誤って

```
local% cd work↵                                     ←間違い
local% rsync -au --delete remote:Program/work .↵
```

とした場合"local"上に$HOME/work/workというディレクトリができます(図1参照)．work

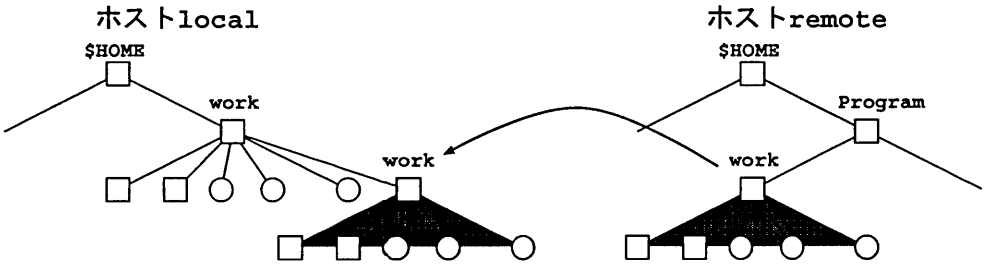


図1: 二重にコピーした例

へ移動した場合は

```
local% cd work↵
local% rsync -au --delete remote:Program/work/ .↵      ←最後に‘‘/’’
```

として，最後に “/” を付ける必要があります．

最後にもう一つの間違った使用例を紹介します．さきほどの誤りでは，ファイルは二重にコピーされますので，無駄な転送が発生しますが，データの消失はおこりません．しかし，先に work があるホームディレクトリへ移動して最後に “/” をつけると，“local” 上のファイルが消えてしまいます (図 2 参照)．

```
local% cd ↵                                          ←work がある場所へ移動
local% rsync -au --delete remote:Program/work/ .↵          ←最後に ‘‘/’’
```

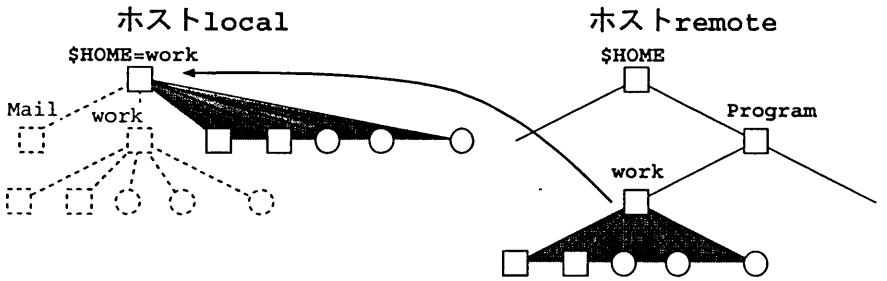


図 2: データ消失が伴う誤った例で，点線は消去されるファイルやディレクトリ

この場合，ホームディレクトリの内容を “remote” 上の$HOME/Program/work 以下の内容と同一に保とうとします．したがって，例えばメールフォルダ Mail など本来 work にはないファイル (図 2 の点線部分) は全て消されてしまいます．

このような結果を避けるために-n オプションを付けて

```
local% rsync -nau --delete remote:Program/work/ .↵
```

を一度実行しましょう．これは，消去やデータ転送は行なわず，どのファイルが消されてどのファイルが転送されるかのみを表示します．あるいは-b オプションを付けてバックアップファイルを作成してもよいでしょう．

次に “local” 側で work 以下のファイルに変更を加えます．これらの変更を “remote” 側へ反映させるには，

```
local% cd ↵                                          ←work がある場所へ移動
local% rsync -au --delete work remote:Program/↵
```

を実行します．work ディレクトリそのものが転送されますので，コピー先は Program を指定します．

5 認証

rsyncでは，デフォルトでrshを用いて認証を行ないます．通常のrshの利用では，毎回パスワードを尋ねられます．rsyncでrshを利用する場合には，パスワードを尋ねられないように接続先(この場合は"remote")のホームディレクトリに.rhostsというファイルを作っておく必要があります．ファイルにはホスト名とユーザ名を

```
local.xxx.kyushu-u.ac.jp username
```

のように書きます．"local"と"remote"でユーザ名が同じ場合はusernameの指定は不要です．

rshはセキュリティの面では脆弱です．rsyncでは，認証にsshが用いることもできます．この場合は，環境変数RSYNC_RSHを設定するか，オプションでsshを使うことを指示します．csh系のシェルを利用している場合，環境変数の設定は，

```
local% setenv RSYNC_RSH ssh
```

とします．オプションで指示する場合は-e sshを用います．

sshの認証には，鍵を用いないパスワード認証と鍵を用いるRSA認証があります．パスワード認証で入力するパスワードは，接続先計算機のパスワードです．このとき入力するパスワードは暗号化されて"remote"側に渡されますのでrshと比較すると，安全性は高いといえます．しかし，暗号化されるとはいえ，パスワードがネットワーク上を流れますし，なにより毎回ディレクトリの同期をとるたびにパスワードを入力するのは面倒です．より安全で，しかもパスワードなしでrsyncを利用するためには，ssh-keygenにより暗号鍵を生成して，RSA認証を利用します．さらにssh-agentによる鍵の管理をして，パスワードの入力なしで接続できるようにします．

まず"local"上でssh-keygenを実行します[2]．

```
local% ssh-keygen↵
Generating RSA keys:  Key generation complete.
Enter file in which to save the key (/xxxx/.ssh/identity):↵←鍵ファイル置場
Enter passphrase:  **************************↵          ←パスフレーズの入力
Enter the same passphrase again:  **************************↵     ←再入力
```

これで暗号鍵が生成されます．鍵ファイル置場として指定したファイルが秘密鍵で，これに.pubをつけたファイルが公開鍵です．公開鍵は，その名の通り公開してよい性質のものですが，秘密鍵の安全には充分気をつけてください．この公開鍵を"remote"の$HOME/.ssh/authorized_keysに追加し，鍵に関する設定は終了です．このファイルがない場合は，新規に作成してください．

"local"にssh-keygenがない場合は，ssh-keygenがある計算機に遠隔ログインして作成した秘密鍵と公開鍵を"local"の$HOME/.sshディレクトリの下に置きます．このとき，ssh-keygenがあるホストへは通常のtelnet等でログインせずに，sshなど，通信路が暗号化されるコマ

[2]表示されるメッセージはsshのバージョンによって異なるかもしれません．

ンドでログインしてください．通常の telnet 等は，通信路が暗号化されないので，鍵の生成時に入力するパスフレーズが平文のままネットワーク上を流れてしまいます．

以上で，認証時に暗号化されたパスワードがネットワークを流れることがなくなります．しかし，このままでは認証のたびに ssh-keygen のときに入力したパスフレーズの入力が必要になります．そこで，一度パスフレーズを入力すれば，以後はパスフレーズ入力なしで ssh が利用できる ssh-agent を利用します．

ssh-agent は，メモリ上に保持してあるユーザの秘密鍵を，秘密鍵が必要な ssh や scp などに渡してくれます．秘密鍵を ssh-agent の管理下に登録するために ssh-add コマンドを使います．ユーザが秘密鍵の正当な保有者かどうかは，ssh-add による登録時に，パスフレーズを入力することで確認されます．

ssh-agent は X Window を起動するときか，ログイン直後に起動します．X Window を利用している場合は

```
local% ssh-agent startx↵
```

とします．startx は X Window を起動する標準のスクリプトであり，自作のスクリプトを指定しても構いません．X Window を利用しない場合は

```
local% ssh-agent tcsh↵
```

などとして，新たにシェルを起動します．

ssh-agent を起動した後 X Window の場合は xterm 等の端末エミュレータ上で，そうでない場合は起動したシェル上で

```
local% ssh-add↵
Need passphrase for /home/daisuke/.ssh/identity
Enter passphrase for username@local:  **********************↵←パスフレーズ
```

とします．これにより，“remote” だけでなく，接続先の authorized_keys に “local” で作成した公開鍵が登録されている計算機との間の ssh 接続で，パスワードもパスフレーズも不要になります．

6　様々なオプション

rsync のオプション一覧は

```
local% rsync --help↵
```

で表示させることができます．これらのオプションから，いくつか有用と思われるものを紹介します．

-z オプションはデータを圧縮して転送します.
-v オプションは冗長なメッセージ出力を行ないます.
--progress オプションは, 各ファイルごとにどの程度転送されたかを示す割合を出力します.
--backup-dir オプションは-b オプションで作成されるバックアップファイルの出力先を指定します.
--exclude=PATTERN オプションは PATTERN で指定されたファイルを無視します. 例えば

```
local% rsync -au --delete --exclude='*.o' work remote:Program/↵
```

のように指定します. PATTERN で指定したいパタンの数が多い場合はパタンをファイルに書いておき

```
local% rsync -au --delete --exclude-from=FILE work remote:Program/↵
```

とします.
--rsync-path=PATH は, "remote" 上の rsync がインストールされたディレクトリを指定します.

7 おわりに

rsync には, 他にも様々な使い方が考えられます. 例えば, 特定のファイル (群) を多数の配布先にコピーする場合は, 配布元の rsync に--daemon オプションをつけて起動しておき, サーバとして動作させると便利でしょう [3]. 同一計算機上の異なるディスクへのバックアップにも使えるでしょう. 紹介したオプション以外にも様々なオプションがありますので, オンラインマニュアル等に一度目を通しておくとよいでしょう.

参考文献

[1] 伊東 栄典, SSH: Secure Shell ～おでかけ前に鍵かけて～, 九州大学大型計算機センター広報 Vol. 32, No. 2, pp.76–89, 1999. (http://www.cc.kyushu-u.ac.jp/RD/itou/koho/1999.vol.32.no.2/ssh1.html/ssh1.html)

[2] 伊東 栄典, 池田 大輔, SSH: Secure Shell (2) ～小荷物を秘かに港で横流し～, 九州大学大型計算機センター Vol. 32, No. 3, pp.127–138, 1999. (http://www.cc.kyushu-u.ac.jp/RD/itou/koho/1999.vol.32.no.3/ssh2.html/ssh2.html)

[3] 島 慶一, UNIX 知恵袋–rsync– UNIX MAGAZINE 2000 年 7 月

九州大学情報基盤センター研究用計算機システム センターニュース(全国共同利用施設版)の案内

情報基盤センター広報室*

1　はじめに

九州大学情報基盤センターでは，旧大型計算機センターが提供していた計算およびデータベースサービスを主に行うシステムを「研究用計算機システム」と呼び，このシステムに関するお知らせを「センターニュース全国共同利用施設版」(以下，センターニュース)として利用者の皆様に提供しています。

九州大学情報基盤センター(以下，センター)の発行するセンターニュースは，電子メールとWWWページ[1]の形で発行しています。このセンターニュースには，講習会の案内や計算機の停止，計算機構成の変更など，利用者の皆様に重要なお知らせが含まれていますので，九州大学情報基盤センター研究用計算機システム利用者の皆様は，センターニュースの電子メールを読んで頂くようお願いします。本稿では，センターからの電子メールによるセンターニュースを読む方法について説明します。

2　配送先

センターニュースは，下記のどちらかに配送されています。

- 利用申請時に記述したメールアドレス
- 端末サーバ wisdom のアカウント

実際には図1に示すように，センターニュースはまず“wisdom.cc.kyushu-u.ac.jp”の宛先に配送されます。つまり，ユーザIDが“i70034a”の人には，まず

“i70034a@wisdom.cc.kyushu-u.ac.jp”

へ配送されています。利用申請時に，メールアドレスを記入されている場合，その宛先に転送するように wisdom 上で設定されています。

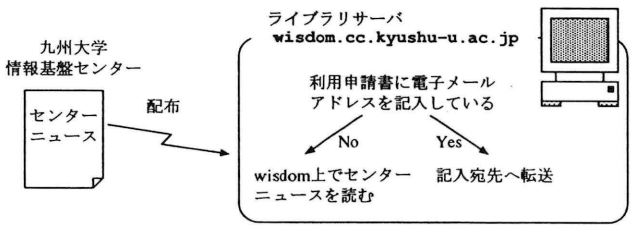


図 1: センターニュースの配送

*E-mail : info-room@cc.kyushu-u.ac.jp

3 転送先の指定

wisdom.cc.kyushu-u.ac.jpへ配送されている電子メールを別の所へ転送したい場合の設定について説明します。

wisdomでは".forward"という名前のファイルをホームディレクトリに置き，そのファイル内に電子メールアドレスを記述しておくと，その宛先にメールを転送するようになっています。センターニュースも電子メールで配信されているため，メールの転送先を指定すれば，その転送先にセンターニュースが届く事になります。

電子メールの転送先を直接指定するには，wisdomにログインし，ホームディレクトリに新たに".forward"ファイルを作成するか，あるいは再編集して，ファイル内に転送先を記述して下さい。下記は"itou@cc.kyushu-u.ac.jp"へ転送するように".forward"を記述している例です。

```
itou@cc.kyushu-u.ac.jp
```

4 wisdom 上での電子メール利用

メールを転送せずに，wisdom上で電子メールを利用する方法について説明します。wisdomに届いている電子メールを読むには，POPを利用する方法と，ログインして電子メールを読み書きする方法の二つの方法があります。

4.1 POPによる方法

電子メールを読み書きするためのソフトウェアであるメーラーには，マイクロソフト社Outlook Express，Netscape社のNetscape Messenger，ジャストシステムShurikenなど様々なメーラーが存在します。これらのソフトウェアではPOPによる電子メールの利用が可能です。POPとはPost Office Protocolの略で，電子メールサーバからメーラーへ電子メールのデータを受け渡す通信手順の一つです。

wisdomにはPOPサーバ (YAT/POP3 server version 4.10p.2) が動作しています。手元にあるメーラーの設定を下記の様に設定すれば，POP方式でメールを読み書きする事が可能です。

メールの授受方式	POP (あるいはPOP3)
POPサーバ名	wisdom.cc.kyushu-u.ac.jp
アカウント名	情報基盤センター研究用システムのユーザID

4.2 ログインしてのメールの読み書き

wisdomに直接ログインし，ターミナル画面でメールを利用する方法について説明します。wisdomでは，mnewsとMew(Emacsで利用)を利用可能です。

■ mnewsの利用

mnewsは電子ニュースと電子メールを読み書きするためのソフトウェアです。mnewsを起動するには，コマンドラインから"mnews"と入力して下さい。普通に起動するとmnewsは起

動時に電子ニュースサーバと接続しニュース記事の読み込みを行います。そのため，起動に時間が掛かります。電子メールだけを利用するためには下記のように "-m" オプションを付けて起動して下さい。

```
% mnews -m
```

mnewsの詳細な利用方法に付きましては，下記WWWページを参照してください。

- 電子メールによるセンターニュースの配布
  http://www.cc.kyushu-u.ac.jp/scp/users/cn-mail.html

■ Mewの利用

Mewは，Emacs系エディタで使用できるメール管理ソフトの1つです。Mewは "Message interface to Emacs Window" の略で，「みゅう」と読みます。Mewは，通常のメールの読み書きだけでなく，MIMEと呼ばれる形式のメールの読み書きや，メールの分類等の便利な機能を持っています。詳細な利用法については，下記の解説や，Mewの公式ホームページ[2]を参照してください。

- メール管理ソフト Mew の使い方
  http://www.cc.kyushu-u.ac.jp/scp/system/library/softs/mew.html
- 電子メール用ソフトウェア Mew の使い方 [3]
  http://www.cc.kyushu-u.ac.jp/RD/itou/1998/mew.pdf

5　おわりに

センターニュースの利用方法について説明しました。冒頭にも書きましたように，講習会やサービスの停止・変更といったセンターからの重要なお知らせを，センターニュースで広報しています。九州大学情報基盤センター研究用計算機システム利用者の皆様は，転送先の指定や，wisdomでのメール利用などをして頂き，センターニュースを読んで頂くようお願いします。

参考文献

[1] 九州大学情報基盤センター研究用計算機システム「センターニュース」
http://www.cc.kyushu-u.ac.jp/scp/users/news.html

[2] MEWホームページ："MEW, Messaging in the Emacs World,"
http://www.mew.org/ .

[3] 伊東栄典, 笠原義晃："電子メール用ソフトウェアMewの使い方",
九州大学大型計算機センター 広報 Vol.31, No.3, pp.143–155, 1998.
http://www.cc.kyushu-u.ac.jp/RD/itou/1998/mew.pdf

PAKDD01参加報告

廣川佐千男*

PAKDD(Pacific-Asia Conference on Knowledge Discovery and Data Mining) に参加した. この国際会議はデータマイニングの理論・技術・応用・発展についての太平洋アジア地域における国際会議で, 毎年アジアの各地で開催され, 昨年は京都府のけいはんなプラザで開催された. 第5回目の今年は香港の Kowloon Shangri-la ホテルで 4/16(月) から 18(水) まで開催された. 米国を中心とする KDD, ヨーロッパを中心とする PKDD とともに, 3つの KDD 国際会議の一つである. 投稿数 152 件, 採択はフルペーパー 38 件, ショートペーパー 22 件. 論文集は Springer LNAI 2035 として出版されている. 初日の 16 日にワークショップ 4 件, チュートリアルが 3 パラレルで 10 件, 17 日, 18 日は朝 9 時から 6 時まで 3 パラレルセッションでびっしり詰まった 3 日間だった.

これらの会議は以前から名前だけは知っていたが, 参加するのは今回が初めてであった. 今回, 参加するしようと思った一番の動機は, Simon Fraser 大学の Jiawei Han のチュートリアル "Sequential Pattern Mining: From Shopping History Analysis to Weblog Mining and DSN Mining" と, 台湾中央大学の Chia-Hui Chan 等の論文 "Applying Pattern Mining to Web Information Extraction" であった. いずれの発表も, HTML, XML などの一般的に半構造化データと呼ばるテキストから, まとまりのあるデータを抽出するものである. このようなプログラムはラッパーと呼ばれ, 1997 年に怒涛のように多くの論文が発表され, それ以降も研究が続いている. この分野は Web Mining として今や最も活発な分野となっている. 例えば 雑誌 Artificial Intelligence[1]や, 国内でも人工知能学会誌[2]でも特集が組まれている.

猫も赤子も WWW の時代となり, 私の研究室でもテーマの一つとして 3 年程前から検索サイトの統合というテーマで取り組んでいる. Yahoo![3]や goo[4]などの複数の検索サイトに一括して検索を行うメタ・サーチのシステムがいくつか知られている. 一般的な検索エンジンでなく, 背後にデータベースがあるようなページが 10 万サイト以上あるという報告[5] もある. 人手での統合は現実的ではなく, それぞれの検索結果の HTML ファイルから, 検索結果の内容だけを自動的に抽出する必要がでてくる. 我々は HTML 文書中のタグの繰り返しパターン着目し, 検索サイトの統合システムを開発している. このようなときに, 近くの香港でその手の最新の研究が多くある PAKDD が開催されると知り, 絶好の機会と思った訳である.

チュートリアルの Han は, 昨年 5 月 IBM Almaden 研究所で, Kleinberg, Brin, Ulmann らが開いた DIMACS/IBM Workshop on Data Mining in the Internet Age [6]や SIGMOD2000 (ACM

*情報基盤センター研究部学術情報メディア研究部門 hirokawa@cc.kyushu-u.ac.jp

[1]Special Issue on Intelligent Internet Systems,Vol. 118, No.1-2, 2000

[2]特集:テキストマイニング 2001 年第 16 巻第 2 号

[3]http://www.yahoo.com/

[4]http://www.goo.ne.jp/

[5]http://www.completeplanet.com/Tutorials/DeepWeb/summary03.asp

[6]http://dimacs.rutgers.edu/Workshops/DataMining/

SIGMOD-SIGACT-SIGART Symp. on Principles of Database Systems)[7]において, "Mining Frequent Patterns without Candidate Generation" という話をしている. 頻出パターンの抽出という意味では, 我々のアプローチは非常に近いのだが, HTML などのラッパーという観点からは捉えられていないので見逃していた. 一方 Chan の話は, HTML などの半構造化データに現れる繰り返しパターンの抽出に, パトリシア・ツリーというデータ構造を用いるものである. 彼女等は, 一般の検索サイトを対象としたメタ・サーチしか考えておらず, 休み時間に, 10万以上の専門検索サイトの話をすると非常に興味を示した. 他に, 広島市立大の宮原等の "Discovery of Frequent Tree Structured Patterns in Semistructured Web Documents" も, HTML ファイルに含まれる繰り返しパターンを木として抽出する話で, 我々が目指しているものに近い研究であった. この3件は, プログラムを見て, いろいろ事前に調べ予想していた内容であった.

その他に, 実際に発表を聞いて, 興味深かったのが次の二人の話であった. 一人はシンガポールの Kent Ridge Digital Labs の Ah-Hwee Tan である. Ah-Hwee Tan は2件発表していて, 1件は, "Predictive Self-Organizing Networks for Text Categorization" であった. これは ARAM というテキスト分類の新しい方法で, Reuters-21578 について, SVM, KNN, LLSF, Gradient descnet NNet, Naive Bayes などテキスト分類の他の手法と比べても性能がよいことを示し, TREC に出す予定といっていた. 一方, そんな効率の競争にどれだけ意義があるかという疑問の声も聞かれた. 同じく Tan のもう1件の発表 "Topic Detection, Tracking, and Trend Analysis Using Self-Organizing Neural Networks" の方が, 実験としては興味深かった. 自己組織化ニューラルネットのクラスタリングを用いて, ニュースのトピックやトレンドを発見するという発表であった. CNET[8]や ZDNet[9]のニュースについて, 昨年の9月, 10月の8週間分の 1468 件のニュースについて, トピックの発見とトレンド解析を行い, 結果として, Microsoft 裁判, Open source conference 関連で Linux, Windows ME などの抽出に成功している. もう一人は, 2次元データのクラスタリングを扱ったオーストラリア Newcastle 大学 Ickjai Lee の "Criteria on Proximity Graphs for Boundary Extraction and Spatial Clustering" である. 16日に開催されたワークショップで他に "MESCAT:Multi-purpose Exploratory Spatial Clustering Analysis Toolbox" という関連研究を発表していた. Short-Long Criterion という概念を導入し, 密なクラスタだけでなく疎なクラスタも同時に抽出する手法を示し, 具体的な図形について分かりやすい発表だった.

じっくりゆったり研究に浸ることができた3日間だった. 返還前, ヨーロッパ行きの乗り継ぎのために, 香港を通過したことはあったが, 目的地として香港に来たのは初めてだった. 当時はビルの合間を縫って狭い滑走路に降り立った. 新空港は街から離れていて, 中心街までの高速道路も整備され, 島に架る吊り橋の見栄えも良かった. 数年内に, ディズニーランドが空港の近くにできるらしい. 日本国内では最近感じない「発展するアジア」の活気にあふれていた. そういえば, 先ほどの Ickjai Lee は, 大学は韓国で, 地理情報システムをテーマにオーストラリアの Peirce で修士を修了した後, オーストラリアでもより活気のある東側の Newcastel 大学へ Ph. D の学生として移ったと話していた. 印象として 2/3 以上の参加者がアジア系だが, 所属は世界中に散らばっていた. 最近は, Web で論文やプレゼンテーション資料が手に入るので, 事前に時間を掛けて読んでおけば何も参加して直接話しを聞く必要もない, という声を聞くことがある. しかし, 実際にその場で見て聞いて感じる活気はやはり直接会わなければ分からない.

[7] http://www.seas.smu.edu/sigmod2000/
[8] http://www.cnet.com/frontdoor/0-1.html
[9] http://www.zdnet.com/

ICOIN-15 国際会議に出席して

岡村耕二†

2001年1月31日から2月2日まで大分県別府市にある別府コンベンションプラザで開催された国際会議 ICOIN-15 (The 15th International Conference on Information Networking) に出席した。ICOINは、日本、韓国、台湾を中心にした分散処理やマルチメディア通信の最新の研究を扱った国際会議であり、この3カ国の間で持ち回りで開催されている。最近では、日本、韓国、台湾の3カ国に限らず、オーストラリア、ヨーロッパ、またアメリカ本土からの論文発表もエントリされている。

ICOINは当初、3カ国による *Joint Workshop on Computer Communication* (JWCC) というコンピュータ通信に関するワークショップとして1986年から発足したが、現在では、IEEE Computer Society がスポンサーとなるような国際会議に発展した。また、今回は、総務省 通信総合研究所が共催という形で開催された。さて、ICOIN-15 のメインテーマは、*Navigating New Directions for Information Networking in the 21st Century* ということで、21世紀に向けた新しい情報ネットワークに関する120件以上の論文発表が行なわれた。

論文発表は、各セッション4つの会場で並行して行なわれた。また、3日間で合計9つのセッションが開催された。論文発表のトピックは以下の通りであり、ICOIN-15 のテーマである次世紀の情報ネットワークに関する内容はもちろん、基盤となるネットワーク技術に関する内容もふんだんに盛り込まれたプログラムであった。

- Mobile Communications
- Traffic Management
- Security
- Distributed Object
- Internet
- Protocol
- Multicast
- High-Speed Network
- Security and Fault Tolerance
- Multimedia Systems
- Distributed Systems
- Fault-tolerant Systems
- Multimedia Communications
- Mobile Systems
- Routing
- QoS (Quality of Service)
- Algorithms
- Performance Evaluation
- Network Application
- Network Architecture
- Agent

また、論文発表とは別に3つのキーノートスピーチがあった。講演者と講演のタイトルは以下の通りである。3つのキーノートスピーチは、いずれも ICOIN-15 のテーマである次世代情報ネットワークが意識されたものであった。最初の講演が行政からの視点、次の野口先生の講演が大学からの視点、最後の講演がビジネスから見た視点と、産官学それぞれの視点で次世代に向けた情報ネットワークに関する講演が行なわれたのは興味深かった。

- **Dr. Bao-Shuh Paul Lin,** *"IA Industry Development Trend and Technology R&D Direction"*, Managing Director, Philips Research East Asia, and Senior Vice President, Philips Research, Taiwan, ROC
- **Dr. Shoichi Noguchi,** *"Future Information Technology and New Social Model"*, President, University of Aizu, Japan
- **Dr. Alexander D. Gelman,** *"Consumer Communications and Mass Market Information Networking at the Edge*

†九州大学情報基盤センター
Email: oka@cc.kyushu-u.ac.jp

of the New Millennium", Vice President, Society Relations, IEEE Communications Society, Chief Scientist, Panasonic Information and Networking Technologies , Laboratory, USA

それから、今回の ICOIN-15 では、*Live Session* という特別のセッションが設けられた。このセッションは、会場の別府、東京、ロンドン、マドリードが総務省の支援による高速インターネットで接続され、IPv6 に関するセッションが遠隔に行なわれた。ネットワークプロトコルとしてはこの国際会議のテーマを反映して次世代 IP プロトコル である IPv6 が採用された。この遠隔セッションでは、メディアとして DV (デジタルビデオ) 形式が用いられ、デジタルビデオデータが IPv6 上を通信され、非常に高品質な画像、音声が提供された。このセッションではまず、マドリードで開催されていた Global IPv6 Summit の様子が中継され、その後、東京とロンドンの間で IPv6 に関する議論が行なわれた。最後に別府、東京、イギリスの間で IPv6 や次世代ネットワークに関する質疑応答が行なわれた。

別府から東京までのネットワークは、総務省の研究開発用ギガビットネットワークである JGN (Japan Giga bit Network) が用いられた。別府に一番近い JGN のノードは北九州であったため、北九州から別府まで専用の ATM 回線が臨時にひかれていた。また、日本からロンドン、マドリードへのヨーロッパへの回線もこのセッションのために臨時の ATM 回線が用意された。デジタルビデオデータを通信するためには 30Mbps 程度の通信帯域を必要とするが、セッション中かなりのパケットロスが発生していた。これは日本とヨーロッパの間の ATM 回線が予約型ではなく、ベストエフォート型であったため、途中の他のトラフィックの影響を受けてしまったためであろう。

この遠隔セッションは同時に次世代ネットワークのデモ的な位置付けもあったが、その効果を強烈に参加者に提示することができたと思う。しかし、同時にパケットロスの多発を目の辺りにして次世代ネットワークを実現するためには解決すべき多くの問題が山積みされていることが認識されたのではないだろうか。

さて、筆者は、ICOIN-15 の豊富なトピックのうち、Mobile Communications 、Internet、Protocol、Multicast 、Multimedia Systems 、Multimedia Communications 、QoS のセッションに積極的に参加して聴講し、また時には議論に参加した。筆者の主観ではネットワーク技術に関するディスカッションは I-ETF (Internet Engineering Task Force) のような即物的なものではなく、基本的なネットワーク技術に基づいた奥の深い内容のものが多かったように思える。一方で、ATM 技術など、学術的には研究する内容が残っているけれども、実用的な面を考えるとその効果に疑問を感じる論文発表が気になった。また、現在の日本でもそうであるように他国でもモーバイルコンピューティング技術に関する研究が盛んであるように感じられた。また、日本に比べると実ネットワークを用いた実証実験的なものが多いように感じられた。それから、筆者の指導する学生による、インターネット電話、マルチキャスト経路制御、アクティブネットワークを用いた高信頼マルチキャストに関するそれぞれの論文発表に対して各国の研究者から色々な議論、またコメントを頂く機会を得ることができたのは非常にありがたいことだと感じた。

次回の ICOIN は、ICOIN-16 として韓国の済州島で 2002 年の 1 月 30 日から 2 月 1 日までの日程で開催される予定である。いま、インターネットはブロードバンドの時代に突入しており、日本でも遅まきながら JGN といった研究ネットワークだけではなく ASDL や、光ファイバを用いた中高速ネットワークが家庭にも浸透してきた。またいわゆる自治体ネットワークも整備が進み、ブロードバンド時代の幕開けということができるであろう。このような日本のネットワーク環境における研究開発と、韓国、台湾といったアジアの他の国でのネットワークの研究開発事情を比較検討するためにもまたこの国際会議に出席し、有益な情報を収集するのは意味があると考えている。

業　務　報　告

ジョブ処理状況

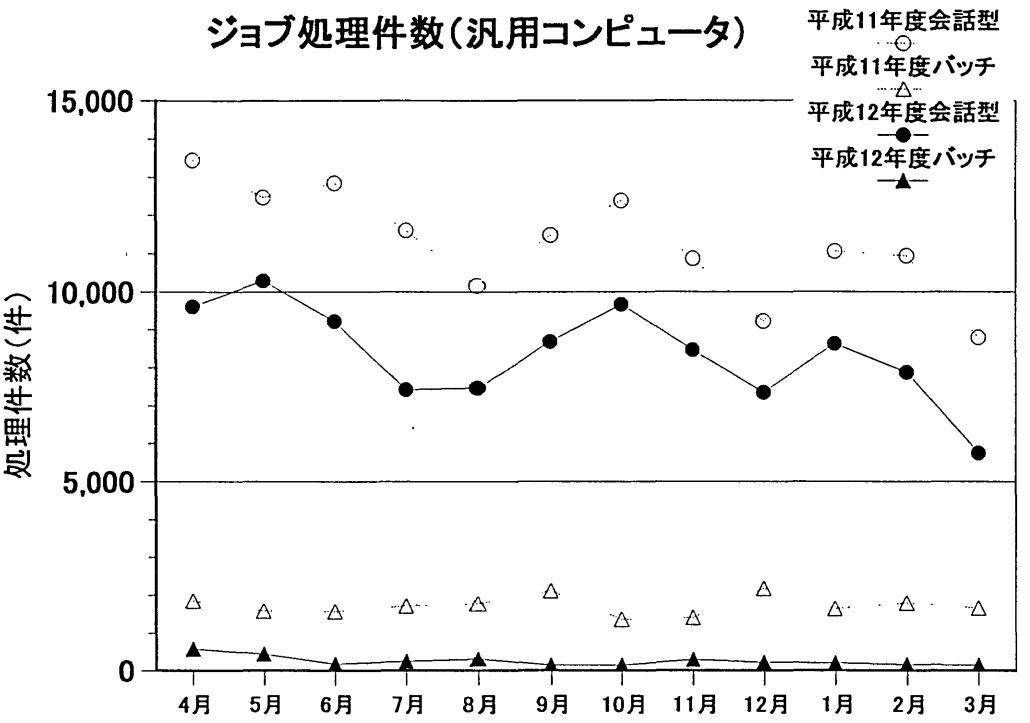
汎用UNIXサーバ Fujitsu GP7000Fモデル900

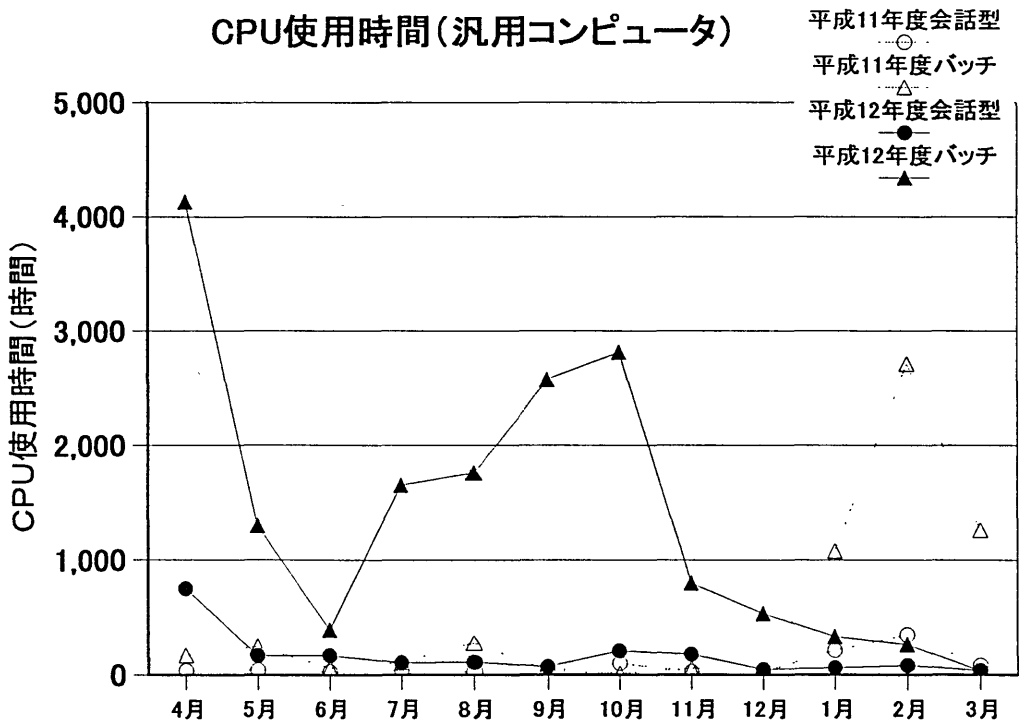
年月	会話型処理		バッチ処理		合計	
	件数（件）	CPU時間（時間）	件数（件）	CPU時間（時間）	件数（件）	CPU時間（時間）
平成12年 4月	9,618	751.35	578	4,130.57	10,196	4,881.92
5月	10,288	173.74	445	1,298.55	10,733	1,472.29
6月	9,229	171.92	181	396.02	9,410	567.94
7月	7,424	109.21	257	1,651.64	7,681	1,760.85
8月	7,477	116.92	323	1,761.46	7,800	1,878.38
9月	8,697	76.76	150	2,574.77	8,847	2,651.53
10月	9,669	213.44	154	2,812.32	9,823	3,025.76
11月	8,469	183.99	310	798.41	8,779	982.40
12月	7,350	53.72	235	536.69	7,585	590.41
平成13年 1月	8,636	65.47	215	335.87	8,851	401.34
2月	7,879	85.31	177	266.26	8,056	351.57
3月	5,761	40.20	160	38.88	5,921	79.08
合計	100,497	2,042.03	3,185	16,601.44	103,682	18,643.47

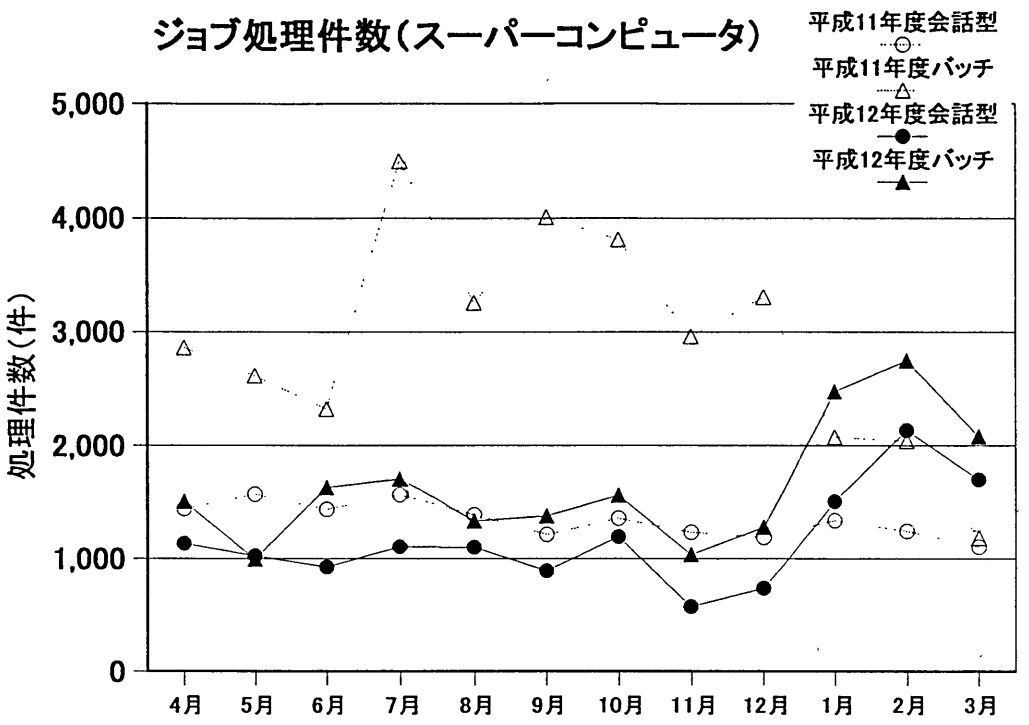
スーパーコンピュータ Fujitsu VPP700モデル56

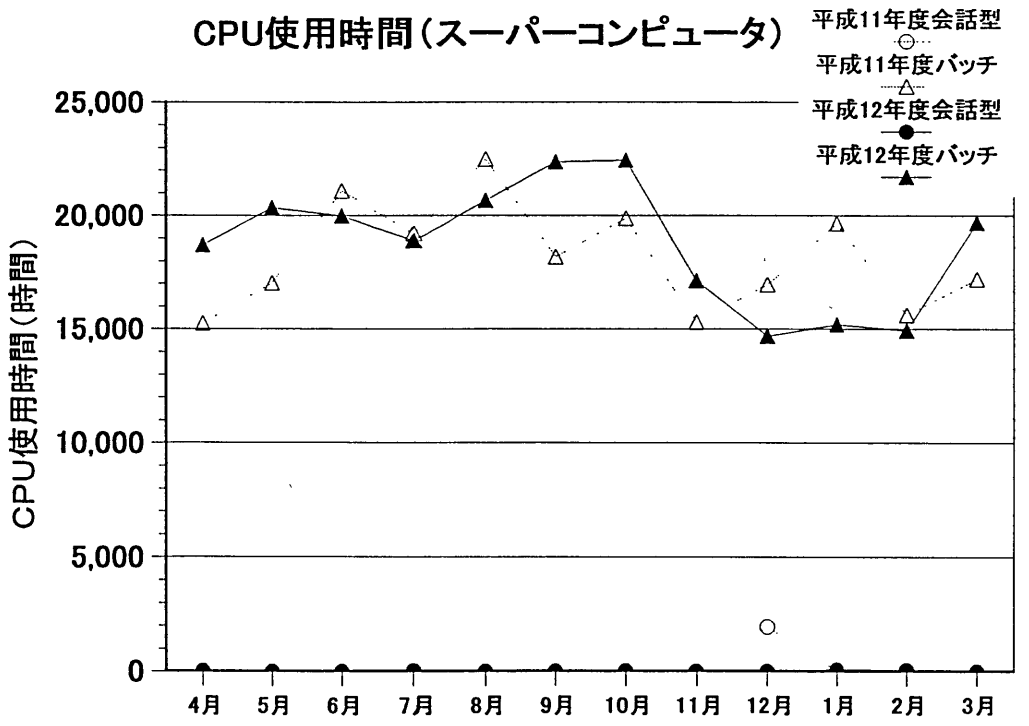
年月	会話型処理		バッチ処理		合計	
	件数（件）	CPU時間（時間）	件数（件）	CPU時間（時間）	件数（件）	CPU時間（時間）
平成12年 4月	1,133	12.39	1,505	18,710.18	2,638	18,722.57
5月	1,024	17.96	996	20,364.89	2,020	20,382.85
6月	926	14.78	1,632	19,983.01	2,558	19,997.79
7月	1,104	29.64	1,702	18,915.92	2,806	18,945.56
8月	1,100	9.25	1,334	20,680.80	2,434	20,690.05
9月	895	9.51	1,377	22,367.31	2,272	22,376.82
10月	1,196	36.97	1,559	22,455.42	2,755	22,492.39
11月	574	10.87	1,037	17,136.84	1,611	17,147.71
12月	741	10.27	1,280	14,682.48	2,021	14,692.75
平成13年 1月	1,506	86.74	2,473	15,203.88	3,979	15,290.62
2月	2,133	67.91	2,749	14,939.17	4,882	15,007.08
3月	1,698	9.05	2,080	19,682.18	3,778	19,691.23
合計	14,030	315.34	19,724	225,122.08	33,754	225,437.42

※並列ジョブのCPU時間は，使用PE（プロセッサ）のCPU時間の総和で表す
※平成13年1月から Fujitsu VPP5000モデル64


ジョブ処理件数(汎用コンピュータ)
平成11年度会話型
平成11年度バッチ
平成12年度会話型
平成12年度バッチ
処理件数(件)
15,000
10,000
5,000
0
4月
5月
6月
7月
8月
9月
10月
11月
12月
1月
2月
3月


CPU使用時間(汎用コンピュータ)
平成11年度会話型
平成11年度バッチ
平成12年度会話型
平成12年度バッチ
CPU使用時間(時間)
5,000
4,000
3,000
2,000
1,000
0
4月
5月
6月
7月
8月
9月
10月
11月
12月
1月
2月
3月


ジョブ処理件数(スーパーコンピュータ)
平成11年度会話型
平成11年度バッチ
平成12年度会話型
平成12年度バッチ
処理件数(件)
5,000
4,000
3,000
2,000
1,000
0
4月
5月
6月
7月
8月
9月
10月
11月
12月
1月
2月
3月


CPU使用時間(スーパーコンピュータ)
平成11年度会話型
平成11年度バッチ
平成12年度会話型
平成12年度バッチ
CPU使用時間(時間)
25,000
20,000
15,000
10,000
5,000
0
4月
5月
6月
7月
8月
9月
10月
11月
12月
1月
2月
3月

平成13年度　講習会開催一覧

講　習　会　名	開催日（曜日）	募集人員	申込者数	受講者数	備考
センター入門	4月25日（水）	40　名	1　名	0　名	実施済
UNIX 初級	5月14日（月） ～15日（火）	40　名	28　名	18　名	実施済
ネットワーク利用法（初級）	5月16日（水）	40　名	8　名	2　名	実施済
プログラミング言語利用	5月25日（金）	40　名	22　名	11　名	実施済
ネットワーク利用法（中級）	5月30日（水）	40　名	22　名	13　名	実施済
UNIX 中級	6月1日（金）	40　名	25　名	18　名	実施済
スーパーコンピュータ VPP 並列プログラミング	6月5日（火） ～6日（水）	40　名	21　名	19　名	実施済
汎用 UNIX サーバ GP7000F 並列プログラミング	6月13日（水） ～14日（木）	40　名	9　名	6　名	実施済
MOPAC2000	6月15日（金）	40　名	5　名	3　名	実施済
MASPHYC	6月29日（金）	40　名	名	名	
α-FLOW	7月　3日（火）	40　名	名	名	
LS-DYNA	7月13日（金）	40　名	名	名	

センター入門

受講者内訳　　4月25日（水）実施

職名 / 学校名	教授	助教授	講師	助手	博士	修士	学部・研究生	その他（技官・事務官）	合計
九州大学	0	0	0	0	0	0	0	0	0
合計	0	0	0	0	0	0	0	0	0

ＵＮＩＸ初級

受講者内訳　　5月14日（月）、15日（火）実施

職名 / 学校名	教授	助教授	講師	助手	博士	修士	学部・研究生	その他（技官・事務官）	合計
九州大学	0	0	0	2	0	3	1	7	13
合計	0	0	0	2	0	3	1	7	13

ネットワーク利用法（初級））

受講者内訳　　5月16日（水）実施

職名 / 学校名	教授	助教授	講師	助手	博士	修士	学部・研究生	その他（技官・事務官）	合計
九州大学	0	0	0	0	0	0	1	1	2
合計	0	0	0	0	0	0	1	1	2

プログラミング言語利用

受講者内訳　　5月25日（金）実施

職名 / 学校名	教授	助教授	講師	助手	博士	修士	学部・研究生	その他（技官・事務官）	合計
九州大学	0	0	0	1	1	5	3	1	11
合計	0	0	0	1	1	5	3	1	11

ネットワーク利用法（中級）

受講者内訳　　5月30日（水）実施

職名 / 学校名	教授	助教授	講師	助手	博士	修士	学部・研究生	その他（技官・事務官）	合計
九州大学	0	1	0	1	1	3	3	4	13
合計	0	1	0	1	1	3	3	4	13

ＵＮＩＸ中級

受講者内訳　　6月1日（金）実施

職名 / 学校名	教授	助教授	講師	助手	博士	修士	学部・研究生	その他（技官・事務官）	合計
九州大学	0	0	0	0	2	4	6	6	18
合計	0	0	0	0	2	4	6	6	18

スーパーコンピュータ VPP 並列プログラミング

受講者内訳　　6月5日（火）、6日（水）実施

職名 / 学校名	教授	助教授	講師	助手	博士	修士	学部・研究生	その他（技官・事務官）	合計
九州大学	0	0	0	1	3	11	3	0	18
宇宙開発事業団	0	0	0	0	0	0	0	1	1
合計	0	0	0	1	3	11	3	1	19

汎用 UNIX サーバ GP7000F　並列プログラミング

受講者内訳　　6月13日（水）、14日（木）実施

職名 / 学校名	教授	助教授	講師	助手	博士	修士	学部・研究生	その他（技官・事務官）	合計
九州大学	0	1	0	0	0	1	4	0	6
合計	0	1	0	0	0	1	4	0	6

MOPAC2000

受講者内訳　　6月15日（金）実施

職名 / 学校名	教授	助教授	講師	助手	博士	修士	学部・研究生	その他（技官・事務官）	合計
九州大学	0	0	0	1	0	1	1	0	3
合計	0	0	0	1	0	1	1	0	3

お　知　ら　せ

平成13年度利用者旅費について …… 126
平成13年度CPU定額利用制度の試行運用のお知らせ（再掲） …… 127
平成13年度年間定額利用制度の試行運用のお知らせ（再掲） …… 128

平成13年度利用者旅費について

遠隔地の利用者が，本センターへ出向き計算機を利用する場合，利用者からの申請に基づいて利用者旅費が支給される制度があります。この制度は本センターを利用するための環境が，不十分な遠隔地の利用者の便宜を図るためのものです。本制度を利用して計算機の利用を希望される方は，下記により手続きをしてください。

記

1. 利用期間 平成13年4月1日 ～ 平成14年3月31日

2. 利用者の出張期間及び旅費支給基準

地区	出発地	出張期間	日当	宿泊料	備考
1～6	北海道，東北，東京，名古屋，京都，大阪，四国	4泊5日以内	円 1,700	円 8,700	センター内利用期間：3日以内
7	鹿児島，宮崎，沖縄	3泊4日以内	1,700	8,700	センター内利用期間：3日以内
	広島，山口，大分，長崎，熊本	2泊3日以内	1,700	8,700	センター内利用期間：3日以内
	上記以外の地域（佐賀，北九州等）	日帰り	1,700		センター内利用期間：3日以内

a. 旅費は，国家公務員等の旅費に関する法律及び文部科学省所管旅費規則の定めるところにより支給します。ただし，鉄道賃については片道50Km以上の場合は急行料金，片道100Km以上の場合は特急料金を支給します。また，日当・宿泊料については，上表のとおり，行政職俸給表（一）1級相当額を支給します。
なお，航空賃については「領収書等及び搭乗券の半券」または「搭乗券の全券（写）」の提出が必要です。

b. 旅費の支給は，精算払いです。

c. 利用者は，所属の機関に設置された連絡所に備付けの「利用者旅費支給申請書」に必要事項を記入して当該連絡所に提出するものとし，提出を受けた連絡所は当該申請書の記入事項を確認の上，連絡所責任者印を押印し，申請者の所属長（実際には人事担当部署）に提出して，本センターに出張利用予定日の10日前までに必ず到着するように手続きをとってください。

d. 文部科学省科学研究費補助金（科研費）による利用者には旅費の支給はできません。

e. 出張利用は申請者本人に限ります。代理者による出張利用は認められません。

f. 土曜日及び日曜祝日にまたがる出張利用は認められません。

g. 宿泊所のあっせん等は行いませんので，利用者各自で手配してください。

3. その他

出張利用申請書を連絡所に提出しただけでは，旅費の支給が承認されたことにはなりません。本センターで申請の内容を審査し，該当する場合は，後日出張依頼書を連絡所宛に送付します。
なお，旅費については，予算の都合上出張期間の短縮を含め，調整させていただく場合がありますので，予め御了承ください。

（共同利用掛 ダイヤルイン 092-642-2305）

平成13年度CPU定額利用制度の試行運用のお知らせ（再掲）

平成12年度実施いたしましたCPU定額利用制度の試行運用を平成13年度も引き続き4月2日(月)より実施します。
これは，計算機システムに余裕のある前期に，出力負担金，ファイル使用負担 金，データベース負担金等の利用負担金のうち演算負担金(CPU利用負担金)に限り，申請額(10万円)の負 担により，その5倍(50万円)まで利用できる制度です。

申請及び利用要領は下記のとおりです。

記

(1)申請について

・申請区分　　10万円コース

・申請受付期間　平成13年2月1日(木)～平成13年8月31日(金)必着
ただし，この申請受付期間内に利用限度額(50万円)を終了した場合に限り，受付期間内であれば何回でも追加申請(更新)を受け付けます。その際，支払費目は1回目と異なってもかまいません。なお，そのときは対象となる利用者番号が変更になります。
(例) 1回目 国立学校等校費(a79999a)　2回目 科学研究費(a79999k)

・利用期間　平成13年4月2日(月)～平成13年10月31日(水)

・申請方法
「CPU定額利用申請書」(コピーでも可)に必要事項を記入の上共同利用掛に提出
[「CPU定額利用申請書」の記入に際しては，(3)「CPU定額利用申請書」記入上の注意をご参照ください。]

・申請の承認「平成13年度CPU定額利用申請承認書」の送付

・申請条件
支払費目はすべての予算区分で利用できます。
1利用者につき1課題のみとします。
承認後の取消しはできません。

※注意事項

◎実際の負担金の請求額は，演算負担金（ＣＰＵ利用負担金）のほかファイル負担金，出力負担金等が加算されますので10万円を超える予算額（最低105,000円）が必要になります。
◎支払費目の予算区分を科学研究費及び産学連携等研究費で申請する場合，通常は利用額が利用見込額を超えると「利用の打切り」で計算機の利用ができなくなりますが，この制度を利用される場合，10月31日（水）までは「警告」となり引き続き利用できます。
したがって，利用額が利用見込額を超えても利用が可能で，「WARNINGメッセージ」が表示されるだけです。
ファイル使用負担金等は継続して課金されますので予算管理には十分ご注意ください。

(2)利用要領について

1. CPU定額利用制度の利用期間は，平成13年4月2日(月)より平成13年10月31日(水)までであり，承認された日からこの制度の対象となります。
2. CPU定額利用制度の利用範 囲は，演算負担金(CPU利用負担金)の合計額が50万円までです。
なお，承認された日からの演算負担金の合計額が，50万円までは請求しませ んが，50万円を超えて利用された部分は，規程に基づいて負担金を請求します。
3. 演算負担金用のコマンド(teigaku) で利用額が確認できます。
4. 10万円の負担金は申請が承認された月の負担金になります。
5. 出力 負担金，ファイル使用負担金，データベース負担金等は本制度の対象外ですので規程 に基づいて負担金を請求します。
6. ジョブの負担金は，ジョブを投入した時点 ではなく，出力を含めて終了した時点で確定するものとします。

(3)「定額利用申請書」記入上の注意について

1. 「＊登録番号」欄 平成13年度新規に計算機利用の申請をされる方は記 入不要です。
2. 「＊支払責任者番号」欄 平成13年度新規に計算機利用の申請をさ れる方は記入不要です。

※ 不明な点は，共同利用掛(ダイヤルイン 092-642-2305)までお 問い合わせください。

(システム運用掛 ダイヤルイン 092-642-2307)

平成13年度年間定額利用制度の試行運用のお知らせ（再掲）

平成13年度より年間定額利用制度を試行いたします。
これは，当該年度の利用負担金として200万円を支払うことにより，演算負担金やファイル負担金等すべての区分での利用合計額が，1,000万円になるまで利用できる制度です。

申請及び利用要領は下記のとおりです。

記

(1)申請について

・申請受付期間　平成13年2月1日(木)～平成13年7月31日(火)必着

・利用期間　平成13年度内

・申請方法
「年間定額利用申請書」(コピーでも可)に必要事項を記入の上共同利用掛に提出
〔「年間定額利用申請書」の記入に際しては，(3)「年間定額利用申請 書」記入上の注意をご参照ください。〕

・申請の承認「平成13年度年度定額利用申請承認書」の送付

・申請条件
支払費目はすべての予算区分で利用できます(組み合わせ可)。
定額負担金200万円(共通負担金含む)は，承認された月の利用負担金として請求します。
承認後の取消しはできません。

(2)利用要領について

1. 利用が承認された場合，センターより利用者番号(定額利用者番号)を 1つ付与します。
2. 年間定額利用制度の対象となるのは，定額利用者番号による利用のみです。
その他の利用者番号による利用は，規程に基づき負担金を請求します。
3. 定額利用者番号による利用が1,000万円を超えた場合は，利用の打ち切りとします。
4. 年度内の利用額が1,000万円に満たない場合の翌年度への繰り越しはできません。
5. 定額利用者番号の第二センター登録はできません。

(3)「年間定額利用申請書」記入上の注意について

1.「＊登録番号」欄　平成13年度新規に計算機利用の申請をされる方は記入不要です。
2.「＊支払責任者番号」欄　平成13年度新規に計算機利用の申請をされる方は記入不要です。

※ 不明な点は，共同利用掛(ダイヤルイン 092-642-2305)までお問い合わせください。

(システム運用掛 ダイヤルイン 092-642-2307)

人　事　異　動

◎転入等

発令年月日	異動後の官職等	氏　名	異動区分	異動前の官職等
13. 4. 1	事務長	冬野　徹	昇　任	施設部企画課課長補佐
〃	共同利用掛長	服部　武雄	配置換	農学部附属農場・演習林専門職員
〃	会計掛主任	森田　裕子	〃	経理部情報処理課企画掛主任
〃	庶務掛	金島　晴子	〃	工学部等総務課
〃	庶務掛・事務補佐員	北野　秀美	採　用	
〃	マルチメディア機器管理掛 技術補佐員	笠野　智子	採　用	
〃	システム運用掛 事務補佐員（パート）	因　八千代	採　用	研究部・技術補佐員 （研究支援推進員）
13. 6.11	会計掛	宮野　敬一	配置換	共同利用掛
〃	共同利用掛	中村　晃子	〃	工学部等経理課工営掛

◎転出・退職者

発令年月日	異動後の官職等	氏　名	異動区分	異動前の官職等
13. 3.30		堺　美和	退　職	庶務掛・事務補佐員
〃		松本　恵里	退　職	マルチメディア機器管理掛 技術補佐員
13. 3.31		栗山　成昭	定年退職	共同利用掛長
〃		平田　直子	辞　職	システム運用掛
〃		花隈　悦子	退　職	研究部・技術補佐員 （研究支援推進員）
13. 4. 1	理学部等事務長	浦辺　洋太郎	配置換	事務長
〃	法学部庶務掛主任	八尋　眞知子	〃	庶務掛主任
〃	経理部契約課第一用度掛	吉田　礼	〃	会計掛
13. 6.11	歯学部総務課経理掛	吉田　研二	〃	会計掛

編集後記

先日、電子辞書を買い換えました。英和・和英辞典と国語辞典、簡易百科事典が搭載されています。シャツのポケットにも入る軽くて薄いもので、いつも鞄に入れて持ち歩いています。

ふつうの検索の他に、うろ覚えの言葉を補完することもできて、たとえば「ほ＊しゅう」と入力すると「法相宗」「法華宗」「本州」などの対応する言葉一覧がずらっと表示されます。

著名な辞書や百科事典、世界地図の多くはパーソナルコンピュータ版を販売するようになりました。これらの多くはハードディスクに内蔵することができるため、ノート型のパーソナルコンピュータにインストールして持ち歩けば、簡単な事柄ならば書斎や図書館にこもらなくてもすぐに調べることができます。

ネットワークに接続されたコンピュータからは、さらにたくさんの(「くず」データを含む)情報を引き出すことができます。最近では、わからないことは、とりあえずWebの検索エンジンにキーワードを投げて調べてみるという人も多いのではと思います。

このように、百科事典をポケットに入れて持ち歩ける時代になると、ただ知識を蓄えているだけの「蘊蓄」ある人だけでは尊敬されません。飲み屋で「百科事典のような知識」をひけらかしても、それだけだと、ポケットから百科事典を取り出されて間違いを突っ込まれたり、「ふうん。で、だからどうした」と言われて黙り込んでしまうことになります。

これからは、尊敬されたいがためであれ、もてたいがためであれ、笑わせたいがためであれ、手軽に入るこれらの知識を、いかに上手に使いこなすかが大切になってきます。

とはいっても、そもそも自分が知ってることを人に披露したとして、それがいったい何になるのかという疑問は残りますけど。

(飲み会での仕事の話は大嫌いなW)

福岡は現在(2001年7月)博多山笠の時期です。また、今年は世界水泳もあるため、街は祭りの雰囲気で盛り上がっている様に感じます。世界水泳では無線による高速インターネット接続実験なども行なわれており、インターネットもすっかり普通の通信基盤になっているというふうに感じます。

さて、遅くなりましたが、2001年度の広報第一号をお送り致します。今年度からは年に3回広報を発行する予定です。年に4回発行していた旧大型計算機センター時代に比べると回数が減っておりますが、その分利用者の要望に沿った内容を記述していきたいと考えております。こういった記事を載せて欲しいといった御希望や、広報の構成についての改善要求などかざいましたら、info-room@cc.kyushu-u.ac.jpまで御連絡下さい。前向きに善処していきたいと考えております。

(E.I.)

投稿のしおり

九州大学情報基盤センターでは、利用者の方々との深い交流と有益な情報交換のため，以下の種類の原稿を募集しています。

- ●随想
- ●計算機を利用した研究・開発の紹介
- ●アプリケーションの実用例や解説
- ●プログラミングの実例や解説
- ●センターに対する質問・要望
- ●利用者の声
- ●計算機やネットワークに関すること

執筆の際は，後述の《執筆上の注意》を必ずご参照ください。投稿された原稿の掲載については，広報委員会で検討させていただきます。

別刷りは，原稿提出時に希望があれば，50部まで無料で差し上げます。50部以上希望される場合は，原稿提出時にご相談ください。

なお，投稿に関するお問合せは共同利用掛（ﾀﾞｲﾔﾙｲﾝ 092-642-2305）までお願いします。

《原稿送付先》

原稿の送付先は，以下のとおりです。

〒８１２－８５８１
福岡市東区箱崎６丁目１０番１号
九州大学　情報基盤センター
共同利用掛　行

《〆切》

原稿の〆切は，5・9・1月の15日です。発行は，その2ヶ月後になります。〆切を過ぎた原稿は，その次の号への投稿扱いとなることがあります。

《執筆上の注意》

1. 原稿は原則としてワードプロセッサ等の出力結果とします。用紙サイズはB5のみとし，書式は図1をご覧ください。手書きでも結構ですが，その場合センター規定の原稿用紙を使用してください。用紙は，共同利用掛にあります。フロッピーディスクや電子メールでの投稿は別途，共同利用掛までご相談ください。
2. ワードプロセッサによる出力は，写真製版とさせていただきますので，投稿原稿の品質にはご注意ください。
3. 手書きの場合は，黒鉛筆，黒ボールペンまたは黒インクのいずれかで書いてください。また，数字，英文字，大(小)文字，上(下)付き文字等混合しやすい文字ははっきりわかりやすく指定してください。
4. 原則として，常用漢字，現代かな使いで統一してください。
5. 第一ページには，必ず題名，著者名，所属および電子メールアドレスを記入してください。記入位置については，図1をご覧ください。

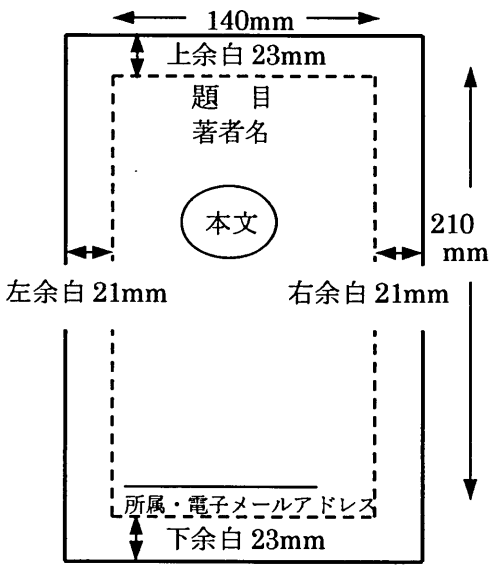


図1：書式設定

業　務　案　内

平成１３年１月１日より

システム利用	サービス時間	各種保守作業による停止時間
スーパーコンピュータ 汎用ＵＮＩＸサーバ	原則通年２４時間 ただし，右欄保守作業の 停止時間を除く	◇定期保守 毎週水曜日　５時～１２時３０分
スカラー並列サーバ その他のコンピュータ	原則通年２４時間	――――

センター内利用	曜　日	開始時刻	終　了　時　刻
オープン利用機器	月～火	９：００	２１：００
	水	１２：３０	
	木～金	９：００	

備考　１．毎週水曜日は，計算機システムの保守のため１２時３０分から計算サービスを開始します。
２．１６時４５分以降（月～金）と土曜日及び日曜祝日は，無人運転を行っており，システム障害が発生した場合は，約２０分後に自動的に計算サービスを再開します。
ただし，３０分経過後に再開できない場合は，計算サービスを打ち切らせていただきます。
３．臨時の停止，運用を行う場合は，その都度センターニュース等でお知らせします。

センター利用案内

092(642)内線番号
※は 092(641)3131(トーン信号)内線番号

担当掛	階	業務案内	内線
研究部	6階	１．計算機システムに関すること ２．ソフトウェアに関すること ３．データベースに関すること ４．開発課題に関すること	２２９６
庶務掛	5階	１．各種委員会に関すること	２３０３
		２．センター案内，見学に関すること	８２２６※
会計掛		１．会計一般に関すること	２３０４
共同利用掛（第七地区協）		１．利用申請，登録に関すること ２．利用の手引き，広報などの配付に関すること ３．講習会に関すること ４．プログラム相談に関すること ５．利用負担金に関すること ６．出張利用者旅費に関すること ７．他センター利用の手続に関すること ８．各大型センター及び連絡所の調整に関すること ９．連絡所の登録，変更及び廃止に関すること 10．センターへの要望，問い合わせに関すること	２３０５ ８２２９※
図書室	4階	１．図書・マニュアル及びセンター関係資料の保管，閲覧及び貸出について	８２３７※
システム運用掛		１．計算サービス全般に関すること	２３０７
		２．計算機システムの運用に関すること	８２３１※
システム管理掛		１．オペレーティングシステムに関すること	２３０８
		２．計算機システムの管理に関すること	８２３２※
ネットワーク運用掛		１．ネットワークの運用に関すること	４０３２
ネットワーク管理掛		１．ネットワークの管理に関すること	２３０９
業務受付	2階	１．オープン室の利用に関すること	２３１１
		２．センターの利用に関すること	８２３４※
プログラム相談室		１．プログラム相談に関する相談・指導	２３１２

◇センター利用に関する質問・要望等の e-mail は下記のアドレスで受け付けます。ご利用ください。

e-mail アドレス　：　request@cc.kyushu-u.ac.jp

九州大学情報基盤センター広報
Vol. 1, No. 2
平成13年7月　発行
編集　九州大学情報基盤センター
広報委員会
印刷　松隈印刷株式会社