JAERI-Data/Code
99-027

原子力コードのVPP500における ベクトル化、並列化及び移植（移植編）

— 平成９年度作業報告書 —

1999年5月

石附　茂*・田邊豪信*・根本俊行*・川崎信夫
川井　渉*・足立将晶・小笠原忍・渡辺秀雄*・久米悦雄

日本原子力研究所
Japan Atomic Energy Research Institute

本レポートは，日本原子力研究所が不定期に公刊している研究報告書です。
入手の問合わせは，日本原子力研究所研究情報部研究情報課（〒319-1195　茨城県那珂郡東海村）あて，お申し越しください。なお，このほかに財団法人原子力弘済会資料センター（〒319-1195　茨城県那珂郡東海村日本原子力研究所内）で複写による実費頒布をおこなっております。

This report is issued irregularly.
Inquiries about availability of the reports should be addressed to Research Information Division, Department of Intellectual Resources, Japan Atomic Energy Research Institute, Tokai-mura, Naka-gun, Ibaraki-ken, 319-1195, Japan.

編集兼発行　　日本原子力研究所

JAERI-Data/Code 99-027

原子力コードのVPP500におけるベクトル化，並列化及び移植 (移植編)
— 平成9年度作業報告書 —

日本原子力研究所計算科学技術推進センター
石附 茂 *・田邊 豪信 **・根本 俊行 *・川崎 信夫 ※
川井 渉 *・足立 将晶 ※・小笠原 忍 ※・渡辺 秀雄 *・久米 悦雄

(1999年4月6日受理)

本報告書は，平成9年度に計算科学技術推進センター情報システム管理課で行った原子力コードのVPP500における高速化作業のうち，移植作業部分について記述したものである．原子力コードのVPP500（一部AP3000）における高速化作業は，平成9年度に14件行われた．これらの作業内容は，今後同種の作業を行う上での参考となりうるよう，作業を大別して，「並列化編」，「ベクトル化編」及び「移植編」の3分冊にまとめた．

本報告書の「移植編」では，沸騰水型原子炉熱水力解析コードTRAC-BF1，連続エネルギー粒子輸送モンテカルロコードMCNP4AのAP3000への移植作業，及び汎用図形処理解析システムIPLOT用ライブラリの改良作業について記述している．別冊の「ベクトル化編」では，多次元二流体モデル構成方程式評価用コードACE-3D，原子核統計崩壊計算コードSD及びHENDEL炉内構造物実証試験部（T2）3次元熱伝導解析コードSSPHEATを対象に実施したベクトル化作業について記述している．また，別冊の「並列化編」では，円筒座標系直接数値解析コードCYLDNS44N，放射能粒子拡散予測コードWSPEEDI，拡張量子分子動力学コードEQMD及び三次元熱流体解析コードSTREAMを対象に実施した並列化作業について記述している．

日本原子力研究所（東海駐在）：〒319-1195 茨城県那珂郡東海村白方白根2-4
※ 外来研究員：富士通株式会社
* 富士通株式会社
** 株式会社ケーシーエス

JAERI-Data/Code 99-027

Vectorization,Parallelization and Porting of Nuclear Codes on the VPP500 System (Porting) - Progress Report Fiscal 1997 -

Shigeru ISHIZUKI*, Hidenobu TANABE**, Toshiyuki NEMOTO*, Nobuo KAWASAKI※, Wataru KAWAI*, Masaaki ADACHI※, Shinobu OGASAWARA※, Hideo WATANABE* and Etsuo KUME

Center for Promotion of Computational Science and Engineering (Tokai Site)
Japan Atomic Energy Research Institute
Tokai-mura, Naka-gun, Ibaraki-ken

(Received April 6 , 1999)

Several computer codes in the nuclear field have been vectorized, parallelized and transported on the FUJITSU VPP500 system and/or the AP3000 system at Center for Promotion of Computational Science and Engineering in Japan Atomic Energy Research Institute. We dealt with 14 codes in fiscal 1997. These results are reported in 3 parts, i.e., the vectorization part, the parallelization part and the porting part. In this report, we describe the porting.

In this porting part, the porting of transient reactor analysis code TRAC-BF1 and Monte Carlo radiation transport code MCNP4A on the AP3000 are described. In addition, a modification of program libraries for command-driven interactive data analysis plotting program IPLOT is described. In the vectorization part, the vectorization of multidimensional two-fluid model code ACE-3D for evaluation of constitutive equations, statistical decay code SD and three-dimensional thermal analysis code for in-core test section (T2) of HENDEL SSPHEAT are described. In the parallelization part, the parallelization of cylindrical direct numerical simulation code CYLDNS44N, worldwide version of system for prediction of environmental emergency dose information code WSPEEDI, extension of quantum molecular dynamics code EQMD and three-dimensional non-steady compressible fluid dynamics code STREAM are described.

Keywords : TRAC-BF1, IPLOT, MCNP4A, Porting, AP3000, VPP500, Nuclear Codes

※ On leave from FUJITSU, Ltd
* FUJITSU, Ltd
** KCS Corp.

目　　次

Contents

1. はじめに

計算科学技術推進センター情報システム管理課では，原研が保有する各種のスーパーコンピュータの効率的な運用とコンピュータ資源の有効利用を促進するため，計算需要の多い原子力コードをユーザに代わってスーパーコンピュータ上に整備し，それぞれのコードに最適な高速化を施す作業を実施している．この作業は，コンピュータの効率的利用を推進するのみならず，ユーザの計算待ち時間の短縮を通じてユーザの仕事の効率化へも貢献するものと思われる．

原子力コードのVPP500（一部AP3000）における高速化作業は，平成9年度に14件行われた．これらの作業内容は，今後同種の作業を行う上での参考となりうるよう，作業を大別して，「並列化編」，「ベクトル化編」及び「移植編」の3分冊にまとめた．

本報告書の「移植編」では，沸騰水型原子炉熱水力解析コードTRAC-BF1，連続エネルギー粒子輸送モンテカルロコードMCNP4AのAP3000への移植作業，及び汎用図形処理解析システムIPLOT用ライブラリの改良作業について記述している．別冊の「ベクトル化編」では，多次元二流体モデル構成方程式評価用コードACE-3D，原子核統計崩壊計算コードSD及びHENDEL炉内構造物実証試験部（T2）3次元熱伝導解析コードSSPHEATを対象に実施したベクトル化作業について記述している．また，別冊の「並列化編」では，円筒座標系直接数値解析コードCYLDNS44N，放射能粒子拡散予測コードWSPEEDI，拡張量子分子動力学コードEQMD及び三次元熱流体解析コードSTREAMを対象に実施した並列化作業について記述している．なお，ここで取り上げなかったいくつかのコードに関しては，ユーザとの連名により別途執筆予定であるので，そちらを参照されたい．

2章では，沸騰水型原子炉熱水力解析コードTRAC-BF1を対象とした移植作業について述べる．本コードは，研究室所有のWSで利用されてきたものであるが，計算時間の増大に伴いより高速な計算機環境への移行が必要となっており，今回AP3000への移植作業を実施した．

3章では，汎用図形処理解析システムIPLOT用ライブラリを対象とした改良作業について述べる．本作業では，EWS上に整備されたIPLOT用図形処理ライブラリ piflibに対し，フォント描画用のPostScript命令をファイルへ出力する機能の追加を行った．

4章では，連続エネルギー粒子輸送モンテカルロコードMCNP4Aを対象とした移植作業について述べる．本作業では，当該コードの並列版（PVM）のAP3000へのインストール及び動作環境の整備を実施した．

なお，本報告書の2章及び3章の作業は田邊が，4章の作業は石附が担当した．

2. TRAC-BF1 の AP3000 へのインストール

2.1 はじめに

TRAC-BF1 コードは，BWR(沸騰水型炉) の仮想事故，過渡及び，関連した実験施設の最適評価解析に使用される [1]．本コードの開発においては，PWR(加圧水型炉) に対する原子炉解析コード TRAC-PF1 の初期バージョンを出発点としているため，基礎方程式や，主要なコーディングの構造においては同等であるが，BWR 固有のコンポーネントモデルや，コードの改良による変更等により，現在のバージョンでは PWR 版と多くの点で異なっている．

日本原子力研究所 (以下，原研) においては，伝熱流動研究室により，PWR の大破断 LOCA 事象を対象として，最適評価コード J-TRAC が開発・整備されてきたが，J-TRAC コードを BWR の解析に使用するには，各種コンポーネントモデルの付加等，大幅な改良が必要であった．そこで，米国原子力規制委員会によって BWR の解析用に開発された TRAC-BF1 を用いて BWR の事故解析を行うこととなった．

これまで，TRAC-BF1 コードは，原研の大型計算機 (富士通製 M780/20 システム，以下 M780) 上に移植され，利用されてきたが，昨今の計算機環境の変化に伴い，UNIX 環境への移植を行うことが必要となった．

このことから，伝熱流動研究室では，UNIX(SUN OS) 版のコードを整備し，ワークステーション上で利用していたが，CPU 性能の点で計算時間が長大なものとなっていた．そこで，平成 9 年度より，情報システム管理課に導入された，富士通製汎用 UNIX サーバ AP3000 への移植を行うこととなった [2]．

以下，その報告である．

2.2 インストール作業

ここでは，UNIX 版 TRAC-BF1 コードを AP3000 にインストールした際の作業について述べる．

2.2.1 コンパイル・リンク

AP3000 は，ノードコンピュータをネットワークで結合した，分散メモリ型の並列サーバで，1 台のノードは，UltraSPARC を搭載し，OS には Solaris2.5.1 (SUN OS) を採用したワークステーションに相当する．更に，UNIX 版 TRAC-BF1 コードは，SUN OS 用に作成されていることから，AP3000 への移植は比較的容易に行なえるものと思われる．

入手したソースファイル一式を AP3000 上に転送し，コンパイル・リンクを行ない，ロードモジュールを作成する．AP3000 では，富士通製コンパイラ [3] と，Sparc コンパイラの 2 種類の FORTRAN コンパイラが利用できるが，今回は，Sparc コンパイラを使用して作業を進めることとした．これは，本コードが SUN OS 用であり，通常のワークステーションで利用されていたことから，互換性も含めて，Sparc コンパイラを使用した方が有効であると判断したことによるものである．

ロードモジュールの作成は，makeコマンドを使用して行う．Makefileは，ソースファイル中に含まれていたものを使用した．Makefileの修正は，'TRACDIR'という定義マクロの部分に，ディレクトリ名を指定するようになっていたため，作業ディレクトリを定義したが，このマクロはMakefile中で引用されていないため，特に変更する必要はなかったものと思われる．これ以外の部分に関しては，特に修正すべき箇所は見当たらない．また，コンパイル及び，リンク時のエラーは，特に発生しなかった．

2.2.2 プログラムの実行

作成したロードモジュールを実行するにあたり，問題点が生じた．これは，入出力に使用するファイルの事前結合に関するものである．富士通製FORTRANコンパイラを使用した場合には，環境変数fuXX（XXは装置参照番号）を使用して，ファイル機番を指定できるが，今回はSparcコンパイラを使用しているため，このような指定が行なえない．

通常，Sparcコンパイラを使用してファイルの事前結合を行なう場合は，サブルーチンioinitを使用して，これを解決するが，本コードにおいては，このサブルーチンは使用されてはいなかった．このため，SUN OS上で本コードを実行する際には，何らかの工夫がなされているものと判断し，ユーザに確認したところ，別プログラムを介して，本コードを実行しているとの回答を得た．

そこで，このプログラム名subを入手し，利用することとした．

このsubプログラムは，FORTRANで記述されたソースプログラムとして入手したため，利用する際にはコンパイルしなければならない．Fig. 2.1にコンパイルするためのコマンドラインを示す．ここで，ソースファイル名は，sub.fとした．

次に，subコマンドを実行するにあたり，必要となるスクリプトファイルを用意する．このファイルは，UNIX上に一般に用いられるシェルスクリプトとは異なり，単にsubコマンドの入力データとして利用されるものである．Fig. 2.2に，その例を示す．

Fig. 2.2で，'WRD1='の行に指定されているのが，TRACコードのロードモジュールである．また，'UNIT='で始まる行は，入出力ファイルの機番とファイル名を指定しており，これをsubコマンドが解釈して，ファイルの装置参照番号を割り当て，TRACコードを実行する．なお，'#'以降の文字列はコメントとして扱われる．

subコマンドの実行は，コマンド名subに続けて，スクリプトファイル名を指定するだけである．Fig. 2.2に示したスクリプトファイルSTEADY.DATを使用して実行する場合のコマンドラインを，Fig. 2.3に示す．

通常，AP3000では，プログラムの実行をqsubコマンドを用いて，バッチジョブとして行う．この場合は，バッチジョブ用にシェルスクリプトを作成し，これにFig. 2.3に示したコマンドラインを記述すればよい．その例をFig. 2.4に示す．ここで，前述のsubコマンドは，カレントディレクトリに存在しているものとする．

ただし，バッチジョブとして実行する場合，subコマンドが解釈するスクリプトファイルに修正が必要となる．これは，Fig. 2.2の例で，24行目に記述されている

```
!tail -f ./output/steady.59 &
```

という部分である．これは，計算の出力をモニタリングするために，tail コマンドを使用している部分であるが，バックグラウンドでの処理となっているため，バッチジョブが終了しても，プロセスが残ってしまうことから，削除するか，コメント化する必要がある．バッチジョブ用に修正したスクリプトファイルを Fig. 2.5に示す．

2.3 作業結果

これまでの作業で，TRAC-BF1 コードを AP3000 上で実行することが可能となった．動作確認として，ユーザより頂いたテストデータ 4 件を用いて，プログラムを実行し，それぞれの計算結果を確認して頂いたところ，4 件とも正しく計算できているとの回答を得た．

2.4 おわりに

本作業においては，もともと SUN OS 上で動作していたものを，やはり，SUN OS 環境である AP3000 への移植であったため，特に障害となる問題も無く，スムースに移行を行うことができた．ファイルの事前結合に関しても，ユーザが利用していた方法をそのまま利用できたため，結果的に，研究室のワークステーションと同様の実行方法を提供できたことは，極めて有効であると思われる．

```
f77 -o sub sub.f
```

Fig. 2.1 Compilation for making 'sub' command.

```
#
#      TRAC-BF1/mod1
#      Test Calculation : STEADY.DAT
#
 WRD1=/dg06/g0342/j9136/trac/bin/tracbf1.exe
 WRD2=/dg06/g0342/j9136/trac/input
 WRD3=/dg06/g0342/j9136/trac/output
!cat /dev/null > ./output/steady.59
#
#
 UNIT=55,FILE=$WRD2/steady.dat,FORM=FORMATTED,STAT=OLD          #INPUT
 UNIT=60,FILE=$WRD3/steady.lst,FORM=FORMATTED,STAT=UNKNOWN      #OUTPUT
 UNIT=11,FILE=$WRD3/steady.grf,FORM=UNFORMATTED,STAT=UNKNOWN    #TRCGRF
 UNIT=12,FILE=$WRD3/steady.dmp,FORM=UNFORMATTED,STAT=UNKNOWN    #TRCDMP
#
 UNIT=09,FILE=$WRD3/steady.9,FORM=FORMATTED,STAT=UNKNOWN        #OUTPUT
 UNIT=14,FILE=$WRD3/steady.14,FORM=UNFORMATTED,STAT=UNKNOWN     #ICHAN
 UNIT=50,FILE=$WRD3/steady.50,FORM=FORMATTED,STAT=UNKNOWN       #OUTPUT
 UNIT=59,FILE=$WRD3/steady.59,FORM=FORMATTED,STAT=UNKNOWN       #OUTPUT
 UNIT=70,FILE=$WRD3/steady.70,FORM=FORMATTED,STAT=UNKNOWN       #WORK
#
 !date
#
!tail -f ./output/steady.59 &
#
 EXEC=$WRD1
#
#
 !date
#
```

Fig. 2.2 Script file for execution of 'sub' command (STEADY.DAT).

```
sub STEADY.DAT
```

Fig. 2.3 Execution of 'sub' command.

```
#!/bin/csh

cd $QSUB_WORKDIR
./sub STEADY.DAT
```

Fig. 2.4 Script file for batch job.

```
#
#      TRAC-BF1/mod1
#      Test Calculation : STEADY.DAT
#
 WRD1=/dg06/g0342/j9136/trac/bin/tracbf1.exe
 WRD2=/dg06/g0342/j9136/trac/input
 WRD3=/dg06/g0342/j9136/trac/output
!cat /dev/null > ./output/steady.59
#
#
 UNIT=55,FILE=$WRD2/steady.dat,FORM=FORMATTED,STAT=OLD          #INPUT
 UNIT=60,FILE=$WRD3/steady.lst,FORM=FORMATTED,STAT=UNKNOWN      #OUTPUT
 UNIT=11,FILE=$WRD3/steady.grf,FORM=UNFORMATTED,STAT=UNKNOWN    #TRCGRF
 UNIT=12,FILE=$WRD3/steady.dmp,FORM=UNFORMATTED,STAT=UNKNOWN    #TRCDMP
#
 UNIT=09,FILE=$WRD3/steady.9,FORM=FORMATTED,STAT=UNKNOWN        #OUTPUT
 UNIT=14,FILE=$WRD3/steady.14,FORM=UNFORMATTED,STAT=UNKNOWN     #ICHAN
 UNIT=50,FILE=$WRD3/steady.50,FORM=FORMATTED,STAT=UNKNOWN       #OUTPUT
 UNIT=59,FILE=$WRD3/steady.59,FORM=FORMATTED,STAT=UNKNOWN       #OUTPUT
 UNIT=70,FILE=$WRD3/steady.70,FORM=FORMATTED,STAT=UNKNOWN       #WORK
#
 !date
#
 EXEC=$WRD1
#
#
 !date
#
```

Fig. 2.5 Script file for execution of 'sub' command (for batch jobs).

参 考 文 献

[1] Akimoto.H and Ohkuni.A, ASSESSMENT OF TRAC-BF1 ID REFLOOD MODEL WITH CCTF AND SCTF DATA, JAERI M-93-045, 1993.

[2] 情報システム管理課,AP3000 システム利用手引き, 平成 9 年 4 月.

[3] FUJITSU, Fortran90 利用手引書 v2 用, 1994 年 10 月.

3. IPLOT 用図形処理ライブラリの改良

3.1 はじめに

汎用図形処理解析システム IPLOT [1] では，図形の描画に使用する処理をカルコンプライブラリ [2] を用いて実現している．現在，IPLOT はワークステーション (以下 EWS) への移行が行われており，ここで使用する図形処理ライブラリとして，カルコンプ互換ライブラリ piflib [3] が採用された．

カルコンプの仕様では，SYMBOL ルーチンを使用して文字列を描画する場合，SYMBOL ルーチン内部に保持している文字パターンデータを用いて行っている．このことは，互換ライブラリである piflib も同様である．しかし，piflib を使用した場合には，PostScript ファイルへの出力を行うことができることから，PostScript 内部のフォントを用いることができれば，より美しい出力を得ることができる．

本作業では，PostScript 内部のフォントを用いて文字列の描画を行うことができるように，piflib ライブラリの改良を行った．

3.2 作業内容

piflib ライブラリでは，描画した図形の出力先として 4 通りの方法が選択できる．本作業では，これらの出力先の中から PostScript ファイルへの出力部分について改良を行った．改良の目的は，PostScript 内部のフォントを用いて文字列を描画できるようにすることである．そもそも PostScript はページ記述言語であり，これを用いて描画を行う場合には，言語仕様に従って適切な命令を発行することが必要となる．従って，今回の作業で行った改良とは，フォントを描画するための適切な PostScript 命令をファイルに出力する機能を実現することを意味する [4].

3.2.1 文字列描画部分の構造

カルコンプ仕様のライブラリにおいて文字列の描画を行うには，SYMBOL ルーチンが使用される．piflib においても同様であるが，piflib の SYMBOL ルーチンでは，そこから図形の出力先に応じて専用のサブルーチンが呼び出されている．Fig. 3.1 に piflib の SYMBOL ルーチンを示す．ここで，item[0] の値によって，それぞれの出力先に応じたサブルーチンに分岐が行われている．PostScript ファイルへの出力は，この値が 2 の時であり，この時 pssymbol ルーチンが呼び出される．従って，今回の作業ではこのサブルーチンを新しいものと置き換えることによって処理を実現することとなる．

3.2.2 サブルーチンの作成

ここで，今回の作業について仕様を述べる．目的は先に述べたように，PostScript のフォントを用いて文字列の描画を行えるように機能を拡張することである．そこで必要となる機能とし

ては,

1 フォントの選択.

2 PostScript のフォントを用いた文字列の描画.

3 センターシンボル [2] の描画.

の 3 種となる. この中で文字列の描画とセンターシンボルの描画は一つのサブルーチンで実現することが必要である. これは, カルコンプライブラリ中の SYMBOL ルーチンの仕様である. 従って, フォントの選択を行うサブルーチンと文字列やセンターシンボルの描画を行うサブルーチンの二つを作成することとした. この二つのサブルーチン名をそれぞれ **setfnt**, **psfont** とする.

3.2.2.1 PostScript 言語の命令

ここでは, PostScript 言語を用いて文字列の描画を行う場合の処理について述べる. 文字列の描画には次の様な記述が必要となる.

font_name **findfont**
num **scalefont setfont**
(*string*) **show**

まず, **findfont** によって, *font_name* で指定されたフォントを辞書から取り出し, **scalefont** で *num* ポイントの大きさに調整し, その状態を **setfont** によってカレントフォントとして設定する. そのカレントフォントを使用して, **show** により () 内の文字列 *string* を描画する. この時, 描画を開始する座標は, その時のカレント座標である. 描画開始座標の指定は **moveto** を使用して行う. 書式は次の通りである.

x y **moveto**

これによって, カレント座標が (x,y) に設定される. ちなみに, カレント座標から指定した座標までの直線を描く場合は **lineto** を用いる. 書式は **moveto** と同様である.

また, SYMBOL ルーチンでは, 描画する文字列を水平軸に対して回転させることが可能である. 座標軸の回転を PostScript で扱う場合は, **rotate** を用いて行う. 書式は次の通りである.

angle **rotate**

ここで, *angle* は座標軸を回転させる角度である. つまり, この記述によって, 現在の座標軸に対して *angle* 度の回転が行われる.

本作業では, これらの命令を使用して適切な処理が行えるように PostScript ファイルへの出力機能を実装することとなる.

3.2.2.2 setfnt ルーチン

サブルーチン **setfnt** は，使用する PostScript フォントを設定するためのものである．本ルーチンは，一つの整数型変数を引数とする．この引数によって使用する PostScript フォントを指定する．使用可能なフォントの一覧を Table 3.1に示す．ここで，各フォント名に対応する数字が本ルーチンへの引数となる．なお，デフォルト値は 1 とする．

先に述べたように，カレントフォントを設定する場合には，**setfont** を使用することになるが，この段階では，描画する大きさについての情報が得られていないため，指定されたフォントの種類 (引数の値) だけを共通変数に格納しておくようにした．Fig. 3.2に **setfnt** ルーチンを示す．ここで，変数 **currentfontID** が共通変数である．

3.2.2.3 psfont ルーチン

psfont ルーチンは， Fig. 3.1 に示した **SYMBOL** ルーチンから呼び出される **pssymbol** ルーチンの代りに使用する．よって， **SYMBOL** ルーチンを Fig. 3.3 のように変更した．

本ルーチンの引数は **SYMBOL** ルーチンのものと完全に同等である．ここで，6 番目の引数 **nchar** の符合に応じて文字列の描画またはセンターシンボルの描画が行われる． **nchar** の値が正である場合，先に述べた PostScript の文法に従って，文字列の描画を行うための命令が PostScript ファイルに記述される．該当部分を Fig. 3.4 に示す．

nchar の値が負であった場合，センターシンボルの描画が行われるが， **SYMBOL** ルーチンの仕様に合致したシンボルが PostScript のフォント中に存在しないため，独自にシンボルパターンを定義し，これを利用してシンボルの描画を行うこととした．

まず，シンボルパターンの定義を行う．座標 (0,0) を原点とする一辺の長さが 1 である正方形を考える (Fig. 3.5参照)． **SYMBOL** ルーチンで使用されるセンターシンボルの 0 番を描画する場合，この正方形の中心 ((0,0) 座標) から y 軸方向に直線を引き (座標 (0,0.5) まで)，そこから正方形の外周に沿って直線を引くことで行うことができる (Fig. 3.6参照)．この処理を PostScript 言語で記述し，使用する際に大きさを調整することでセンターシンボルの描画を行うことができる．なお，定義するシンボルパターンは PostScript のユーザ辞書に登録しておき，使用する際にこれを呼び出して処理を行うこととする．このシンボルパターンの定義部分を Fig. 3.7 から Fig. 3.19 に示す．また， Fig. 3.20 に，これを利用してセンターシンボルを描画するための PostScript ファイルへの出力部分を示す．

3.3 サンプルプログラムと出力結果

Fig. 3.21に PostScript のフォントを用いて描画を行うためのサンプルプログラムを示す．ここで，作成した **setfnt** ルーチンを使用してフォント種類の指定を行っている．また，実際に PostScript ファイルへの出力を行う **psfont** ルーチンは **SYMBOL** ルーチン内で呼び出されるため，ユーザプログラムで使用することはない．

このプログラムを実行し，作成した PostScript ファイルを印刷した結果を Fig. 3.22に示す．更に，センターシンボルを描画するためのサンプルプログラムを Fig. 3.23 に示し．その印刷結果を Fig. 3.24 に示す．

3.4 おわりに

piflib ライブラリは UNIX 環境におけるカルコンプライブラリの互換ルーチンとして，原研内で広く利用されている．今回の作業では，PostScript ファイルへの出力時のみフォントを用いて文字列を描画することができるように改良を行った．しかし，X Window System への描画時には，依然として従来の文字パターンデータを使用している．本来ならば，X Window System への描画時にも X のフォントを利用できるようにするべきであるが，この X のフォントと PostScript フォントとの間で同一のフォントを指定するのは非常に困難である．従って，表示先によって指定の異なるフォントを利用する場合に，如何にして似通ったフォントを指定するかが今後の課題となるであろう．

Table 3.1 PostScript fonts.

フォントID	フォント名
1	Times-Roman
2	Times-Italic
3	Times-Bold
4	Times-BoldItalic
5	Helvetica
6	Helvetica-Oblique
7	Helvetica-Bold
8	helvetioca-BoldOblique
9	Courier
10	Courier-Oblique
11	Courier-Bold
12	Courier-BoldOblique

```
symbol(x, y, hh, ibcda, angle, nchar )

    float *x, *y, *hh, *angle;
    int *nchar;
    int *ibcda;
{
/*  item[0] --  1 : X Window System.
                2 : PostScript File.
                3 : EPS File.
                4 : Graphic Data File.    */

    switch( item[0] ){
      case '1':
xsymbol(x, y, hh, ibcda, angle, nchar );
break;
      case '2':
pssymbol(x, y, hh, ibcda, angle, nchar );
break;
      case '3':
texsymbol(x, y, hh, ibcda, angle, nchar );
break;
      case '4':
gdsymbol( x, y, hh, ibcda, angle, nchar );
break;
    }
}
```

Fig. 3.1 Subroutine SYMBOL (Original Version).

```
setfnt_( font_ID )
  long *font_ID;
{
   currentfontID = *font_ID;
}
```

Fig. 3.2 Subroutine setfnt.

```
symbol(x, y, hh, ibcda, angle, nchar )

    float *x, *y, *hh, *angle;
    int *nchar;
    int *ibcda;
{
/*  item[0] --  1 : X Window System.
                2 : PostScript File.
                3 : EPS File.
                4 : Graphic Data File.    */

    switch( item[0] ){
      case '1':
xsymbol(x, y, hh, ibcda, angle, nchar );
break;
      case '2':
psfont(x, y, hh, ibcda, angle, nchar );
break;
      case '3':
texsymbol(x, y, hh, ibcda, angle, nchar );
break;
      case '4':
gdsymbol( x, y, hh, ibcda, angle, nchar );
break;
    }
}
```

Fig. 3.3 Subroutine SYMBOL (New Version).

```
fprintf(fp, "/%s findfont\n", font_name[currentfontID-1] );
fprintf(fp, "%ld scalefont setfont\n", (long)ps_h);
if( *x == 999.0 || *y == 999.0 ){
   fprintf( fp,"currentpoint moveto\n");
}else{
   fprintf(fp, "%ld %ld moveto\n",(long)ps_x,(long)ps_y );
}
fprintf(fp, "%ld rotate\n", (long)*angle );
fprintf(fp, "(%s) show\n", buf);
fprintf(fp, "-%ld rotate\n",(long)*angle );
```

Fig. 3.4 Drawing sentence of string (subroutine psfont).

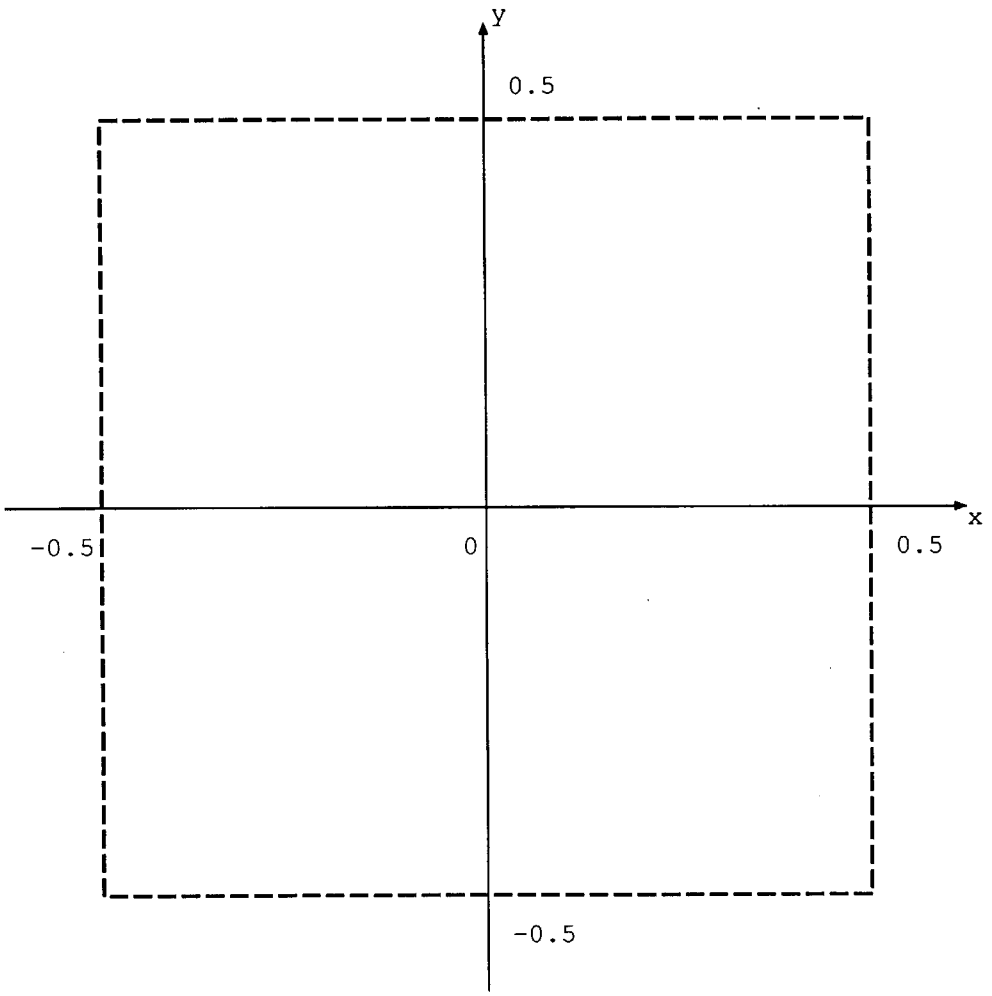


Fig. 3.5 Squere that the origin is coordinate (0,0) and the length of one side is 1.

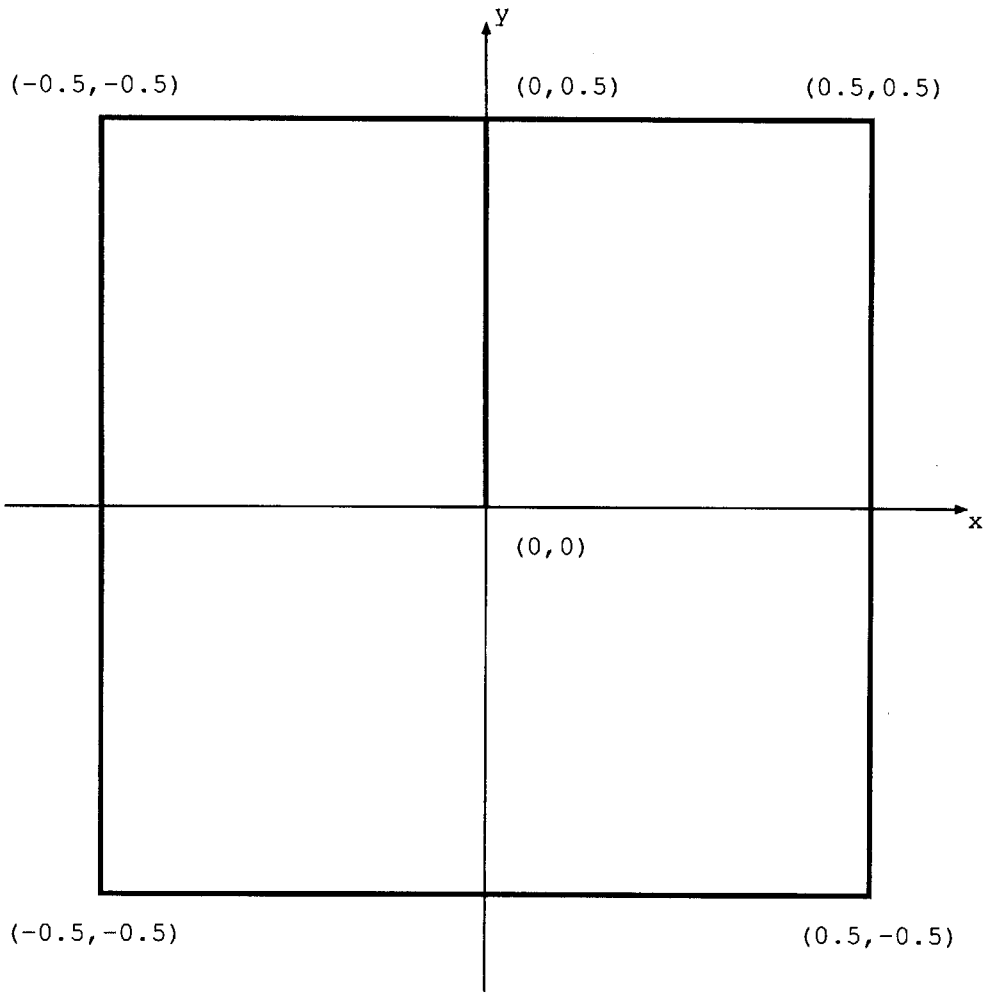


Fig. 3.6 Drawing procedure of definition of symbol 0.

```
/Sym0 { newpath
        0 0 M
        0 0.5 L
       -0.5 0.5 L
       -0.5 -0.5 L
        0.5 -0.5 L
        0.5 0.5 L
        0 0.5 L
        closepath}def
```

Fig. 3.7 Definision of symbol 1.

```
/Sym1 { newpath
        0 0 M
        0 0.5 L
       -0.25 0.5 L
       -0.5 0.25 L
       -0.5 -0.25 L
       -0.25 -0.5 L
        0.25 -0.5 L
        0.5 -0.25 L
        0.5 0.25 L
        0.25 0.5 L
        0 0.5 L
        closepath }def
```

Fig. 3.8 Definision of symbol 2.

```
/Sym2 { newpath
        0 0 M
        0 0.5 L
       -0.5 -0.5 L
        0.5 -0.5 L
        0 0.5 L
        closepath }def
```

Fig. 3.9 Definision of symbol 3.

```
/Sym3 { newpath
        0 0.5 M
        0 -0.5 L
       -0.5 0 M
        0.5 0 L
        closepath }def
```

Fig. 3.10 Definision of symbol 4.

```
/Sym4 { newpath
       -0.5 0.5 M
        0.5 -0.5 L
       -0.5 -0.5 M
        0.5 0.5 L
        closepath }def
```

Fig. 3.11 Definision of symbol 5.

```
/Sym5 { newpath
        0 0 M
        0 0.5 L
       -0.5 0 L
        0 -0.5 L
        0.5 0 L
        0 0.5 L
        closepath }def
```

Fig. 3.12 Definision of symbol 6.

```
/Sym6 { newpath
        0 -0.5 M
        0 0.5 L
       -0.5 0 L
        0.5 0 L
        0 0.5 L
        closepath }def
```

Fig. 3.13 Definision of symbol 7.

```
/Sym7 { newpath
        -0.5 -0.5 M
         0.5 0.5 L
        -0.5 0.5 L
         0.5 -0.5 M
        -0.5 0.5 L
         closepath }def
```

Fig. 3.14 Definision of symbol 8.

```
/Sym8 { newpath
        -0.5 0.5 M
         0.5 0.5 L
        -0.5 -0.5 L
         0.5 -0.5 L
        -0.25 0 M
         0.25 0 L
         closepath }def
```

Fig. 3.15 Definision of symbol 9.

```
/Sym9 { newpath
        -0.5 0.5 M
         0 0 L
         0 -0.5 L
         0 0 M
         0.5 0.5 L
         closepath }def
```

Fig. 3.16 Definision of symbol 10.

```
/Sym10 { newpath
         0 0 M
         0.5 0.5 L
        -0.5 0.5 M
        -0.25 0.25 L
        -0.5 -0.5 M
        -0.25 -0.25 L
         0.5 -0.5 M
         0.25 -0.25 L
         0.25 0.25 L
        -0.25 0.25 L
        -0.25 -0.25 L
         0.25 -0.25 L
         closepath }def
```

Fig. 3.17 Definision of symbol 11.

```
/Sym11 { newpath
         0 0.5 M
         0 -0.5 L
        -0.5 0.5 M
         0.5 -0.5 L
        -0.5 0 M
         0.5 0 L
         0.5 0.5 M
        -0.5 -0.5 L
         closepath }def
```

Fig. 3.18 Definision of symbol 12.

```
/Sym12 { newpath
        -0.5 -0.5 M
         0.5 0.5 L
        -0.5 0.5 L
         0.5 -0.5 L
        -0.5 -0.5 L
         closepath }def
```

Fig. 3.19 Definision of symbol 13.

```
sym_pt = (long *)str;
fprintf(fp,"gsave\n");
fprintf(fp,"0 setlinewidth\n");
fprintf(fp,"[] 0 setdash\n");
fprintf(fp,"%ld %ld translate\n",(long)ps_x,(long)ps_y );
fprintf(fp,"%ld %ld scale\n",(long)ps_h,(long)ps_h);

switch( *sym_pt ){
case 0:
   fprintf(fp,"Sym0\n");
   break;
case 1:
   fprintf(fp,"Sym1\n");
   break;
case 2:
   fprintf(fp,"Sym2\n");
   break;
case 3:
   fprintf(fp,"Sym3\n");
   break;
case 4:
   fprintf(fp,"Sym4\n");
   break;
case 5:
   fprintf(fp,"Sym5\n");
   break;
case 6:
   fprintf(fp,"Sym6\n");
   break;
case 7:
   fprintf(fp,"Sym7\n");
   break;
```

Fig. 3.20 Output sentence to PostScript files to draw symbols (1/2).

```
        case 8:
           fprintf(fp,"Sym8\n");
           break;
        case 9:
           fprintf(fp,"Sym9\n");
           break;
        case 10:
           fprintf(fp,"Sym10\n");
           break;
        case 11:
           fprintf(fp,"Sym11\n");
           break;
        case 12:
           fprintf(fp,"Sym12\n");
           break;
        }
        fprintf(fp,"stroke\n");
        fprintf(fp,"grestore\n");
```

Fig. 3.20 Output sentence to PostScript files to draw symbols (2/2).

```
      call plots(0.0, 0.0 )
      call setfnt(1)
      call symbol(10.0,130.0,5.0,'ABCDEFGH',0.0,8)
      call setfnt(2)
      call symbol(10.0,120.0,5.0,'ABCDEFGH',0.0,8)
      call setfnt(3)
      call symbol(10.0,110.0,5.0,'ABCDEFGH',0.0,8)
      call setfnt(4)
      call symbol(10.0,100.0,5.0,'ABCDEFGH',0.0,8)
      call setfnt(5)
      call symbol(10.0,90.0,5.0,'ABCDEFGH',0.0,8)
      call setfnt(6)
      call symbol(10.0,80.0,5.0,'ABCDEFGH',0.0,8)
      call setfnt(7)
      call symbol(10.0,70.0,5.0,'ABCDEFGH',0.0,8)
      call setfnt(8)
      call symbol(10.0,60.0,5.0,'ABCDEFGH',0.0,8)
      call setfnt(9)
      call symbol(10.0,50.0,5.0,'ABCDEFGH',0.0,8)
      call setfnt(10)
      call symbol(10.0,40.0,5.0,'ABCDEFGH',0.0,8)
      call setfnt(11)
      call symbol(10.0,30.0,5.0,'ABCDEFGH',0.0,8)
      call setfnt(12)
      call symbol(10.0,20.0,5.0,'ABCDEFGH',0.0,8)
      call plot(0.0,0.0,999)
      stop
      end
```

Fig. 3.21 Sample program to draw string.

ABCDEFGH

ABCDEFGH

ABCDEFGH

ABCDEFGH

ABCDEFGH

ABCDEFGH

ABCDEFGH

ABCDEFGH

ABCDEFGH

ABCDEFGH

ABCDEFGH

ABCDEFGH

Fig. 3.22 Printed results.

```
      call plots(0.0, 0.0 )
      call symbol(10.0,150.0,5.0,1,0.0,-1)
      call symbol(10.0,140.0,5.0,2,0.0,-1)
      call symbol(10.0,130.0,5.0,3,0.0,-1)
      call symbol(10.0,120.0,5.0,4,0.0,-1)
      call symbol(10.0,110.0,5.0,5,0.0,-1)
      call symbol(10.0,100.0,5.0,6,0.0,-1)
      call symbol(10.0,90.0,5.0,7,0.0,-1)
      call symbol(10.0,80.0,5.0,8,0.0,-1)
      call symbol(10.0,70.0,5.0,9,0.0,-1)
      call symbol(10.0,60.0,5.0,10,0.0,-1)
      call symbol(10.0,50.0,5.0,11,0.0,-1)
      call symbol(10.0,40.0,5.0,12,0.0,-1)
      call plot(0.0,0.0,999)
      stop
      end
```

Fig. 3.23 Sample program to draw symbols.

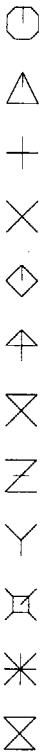


Fig. 3.24 Printed results.

参 考 文 献

[1] 汎用図形処理解析プログラム IPLOT 使用説明書, 1996 年 2 月.

[2] CALCOMP ソフトウェアマニュアル グラフィック COM プログラミング – ベーシックソフトウェア–, 吉沢ビジネスマシンズ (株),1972 年 6 月.

[3] 情報システム管理課, pifiib 利用手引書, 1996 年 3 月.

[4] Adobe Systems,PostScript リファレンス・マニュアル, アスキー出版局, 1993 年 1 月.

4. MCNP4A コードの AP3000 へのインストール

4.1 コード概要

ＭＣＮＰ（Monte Carlo N-Particle transport code system）は，ロスアラモス国立研究所（ＬＡＮＬ）において開発された遮蔽計算，臨界計算用の連続エネルギー粒子輸送モンテカルロコードである．このコードの特徴は，取り扱うことのできるソース分布の種類が豊富で，種々の分散低減法が用意されており，問題の体系の記述に対して柔軟性の高い幾何形状記述方式を採用していることである．さらに，バージョン４以降は，従来の中性子，光子の輸送計算機能に加えて電子輸送計算機能，並列版の選択機能，計算実行中にタリーを定期的にプロットできる機能などが追加されている [1,2].

4.2 コードの構造

ＭＣＮＰ４Ａは大きく分けて以下に示す６つのサブルーチン群から成り立っている． mcrun は主に output,trnspt の２つの部分から構成されており，粒子の輸送計算は trnspt 以下の hstory で行なっている．構成概要図を Fig. 4.1 に示す.

（1） ｉｍｃｎ
　入力データを読み込み，セルの体積計算など粒子輸送計算の前準備をする.
（2） ｘａｃｔ
　入力データで指定されたエネルギー範囲の断面積ライブラリを読み込む.
（3） ｍｃｒｕｎ
　粒子の輸送計算の実行，中間結果の編集出力をする.
（4） ｏｕｔｐｕｔ
　各種の統計量を集計し出力する.
（5） ｔｒｎｓｐｔ
　ヒストリー数，計算時間，ランダムウオークの追跡の途中で幾何学的矛盾を生じた迷子粒子数，中間結果の出力間隔，リスタート用データの出力間隔などを制御する.
（6） ｈｓｔｏｒｙ
　粒子の輸送計算を行うルーチン群であり，粒子のランダムウオークの追跡を行う.

4.3 モンテカルロ法

放射線の物質内輸送を扱うモンテカルロ法について述べる．モンテカルロ法を適用し，ある体系に入射または体系内で発生した放射線（光子）が，その体系内で衝突を繰り返して散乱，吸収等を受けて，位置，方向，速度（エネルギー）を変えながら移動する過程を追跡し，ある特定領域に到達する量や体系内で吸収された量等を求めている．モンテカルロ法のフローチャートを Fig. 4.2 に示す.

4.4 インストール

インストール作業は，ロスアラモス研究所より提供されているインストールシェルを用いて行なった．このシェルは，あらゆるＯＳのマシンにインストールできるように工夫されている．また，異常が発生した場合のサポートのことも考慮し添付シェルを使用した．関連ファイルは下に示すユーザ共有のディレクトリ配下に格納されている．

```
/dg06/center/codelib/mcnp4a
```

ＭＣＮＰ４Ａには計算機能だけではなく，会話型図形表示機能が備わっている．また，ＰＶＭライブラリを利用した並列処理機能も装備している．今回のインストールでは，通常のＭＣＮＰ４Ａ版に加えて，会話型図形表示版も同時にインストールした．しかし，ＡＰ３０００の運用制限により，すべてのプロセッサ上で会話型図形表示機能を使用できるわけではない．使用できるのは，会話処理サーバとアプリケーションサーバの２台だけである．通常の計算機能については全てのプロセッサ上で実行可能である．

4.4.1 ソースプログラム格納位置

ソースプログラムは， mcnp4a 共有ディレクトリ配下の以下のディレクトリに格納してある．

```
通常版　　：/dg06/center/codelib/mcnp4a/singl/source/*.f
図形表示版：/dg06/center/codelib/mcnp4a/xlib/source/*.f
```

4.4.2 ロードモジュール格納位置

ロードモジュールは， mcnp4a 共有ディレクトリ配下の以下のディレクトリに格納してある．

```
通常版　　：/dg06/center/codelib/mcnp4a/NQS/mcnp.si
図形表示版：/dg06/center/codelib/mcnp4a/NQS/mcnpx
```

4.4.3 実行用シェルスクリプト（雛型）格納位置

バッチ処理用の実行シェルスクリプトの雛型を以下のディレクトリに格納してある．また，会話型図形表示機能はバッチ処理では使用できない．

```
通常版　　：/dg06/center/codelib/mcnp4a/NQS/mcnp_sh
```

4.4.4 断面積ライブラリ格納位置

ＭＣＮＰ用断面積ライブラリは，以下のディレクトリに格納してある．また，断面積ライブラリは，会話型図形表示版も同様であるため，両者共同一場所となっている．

```
/dg06/center/codelib/mcnp4a/mcnplib
```

4.4.5 断面積ライブラリのインストール

ＭＣＮＰ４Ａ版に添付されている基本的な断面積ライブラリに加え，Ｍｏｎｔｅ－４上で使用していた断面積ライブラリについてもＡＰ３０００上にインストールした．インストールした断面積ライブラリのリストを以下に示す．

```
531DOS-2    DRE5-2     FNDF5T-2    FENDL1-2
FXJ32A-2    MCPLIB-2   RMCCS-2     532DOS-2
DRMCCS-2    ENDF5U-2   FSXDOS-2    FXJ32B-2
MGXSNP-2    RMCCSA-2   BMCCS-2     EL-2
ENDL85-2    FSXJFF-2   FXJ32C-2    NEWXS-2
THERXS-2    D9-2       ENDF5P-2    EPRIXS-2
FSXLJ3-2    LLLDOS-2   NEWXSD-2    TMCCS-2
```

4.5 実行方法

4.5.1 会話型図形表示機能

会話型図形表示機能を利用する場合の実行方法 [3] について記述する．

（１）ＡＰ３０００にログインする．
（２）グラフィックを出力するディスプレイを指定する．
　　（例：% setenv DISPLAY host-name:0.0 ）
（３）コマンドラインより mcnp を実行する．
　　（例：% mcnp ip inp=inp01 ）
　　(以降は通常のＭＣＮＰと同様の操作)

4.5.2 計算のみの実行

次に，計算のみ（タリー出力を含む）の場合の実行方法について記述する．

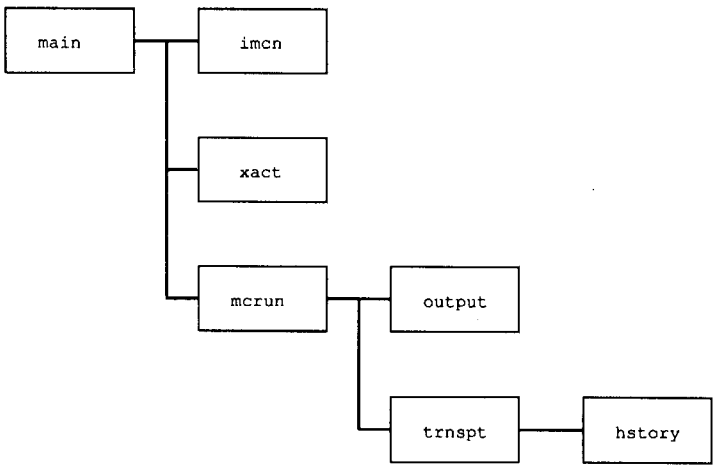
（１）ＡＰ３０００にログインする．
（２）実行用シェルを用意する．
（３）コマンドラインより qsub コマンドを使用し計算サーバへジョブを投入する．
　　（例：% qsub go.sh ）

4.5.3 実行用シェル

計算サーバへジョブを投入する時には，実行用シェルが必要となる．実行用シェル内に，実行環境やＭＣＮＰへの入力データ，オプション等を指定する．実行用シェルの例を Fig. 4.3 に示す．

4.6 おわりに

ＭＣＮＰは世界的に広く使用されているコードであり，現在でもＬＡＮＬにおいてバージョンアップ作業が行なわれている．今後は，さらなる機能強化と共に，並列版における処理方法の改善も予定されている．加えて，旧バージョンにおける不具合点が修正される場合もあるため，新バージョンが提供された時点でバージョンアップすることが望ましい．対応ＯＳも幅広く，ＵＮＩＸやＤＯＳやＩＲＩＸ等様々なＯＳに対応している．次に，断面積ライブラリの整備については現在のところＦＥＮＤＬのみとなっている．ＪＥＮＤＬの整備については，今後の課題として残されている．



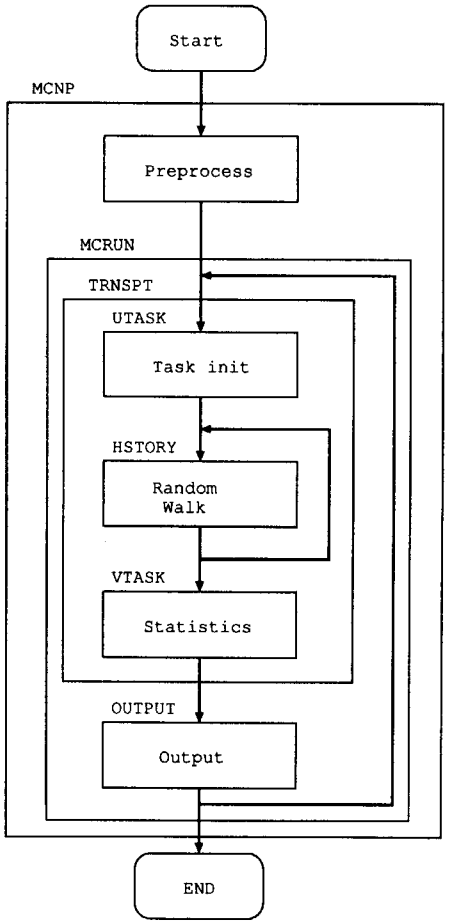


Fig. 4.1 Composition and flow chart of MCNP4A code.

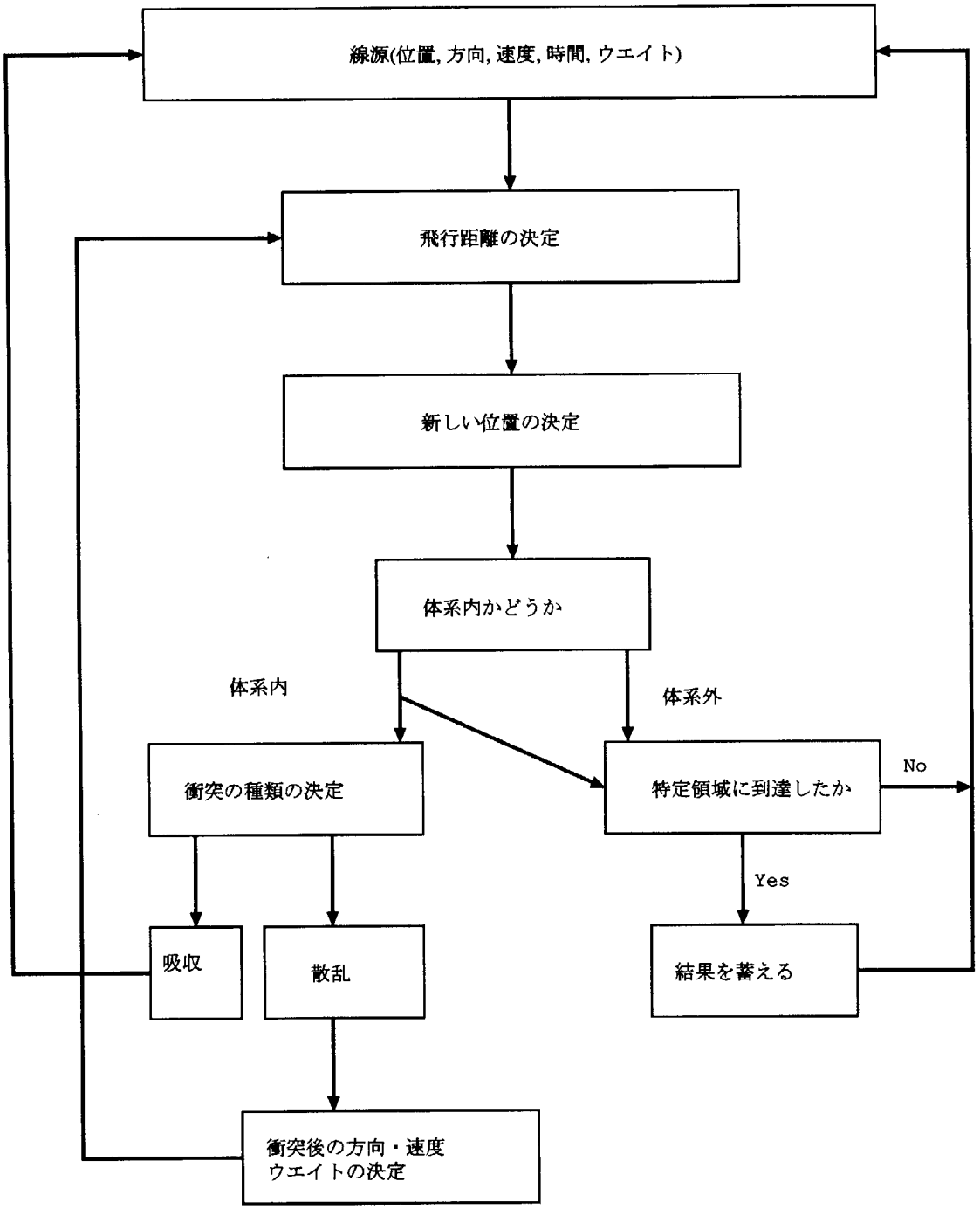


Fig. 4.2 Montecalro calculation for radiation transport.

```
#
#  MCNP execute shell
#
#!/bin/csh
#@$-q l
#@$-eo
#@$-C MCNP
# 断面積ライブラリ・ロードモジュールへのパスの指定

setenv MCNPLIB /dg06/center/codelib/mcnp4a/mcnplib
setenv MCNPLM  /dg06/center/codelib/mcnp4a/NQS

#
# カレントディレクトリの移動

cd $QSUB_WORKDIR

#
# 断面積ライブラリ・ロードモジュールをカレントディレクトリへリンクする

ln -s $MCNPLIB/xsdir $QSUB_WORKDIR/xsdir
ln -s MCNPLM/mcnp $QSUB_WORKDIR/mcnp

#
# MCNP の実行

mcnp name=inp01
```

Fig. 4.3 Shell of execution.

参 考 文 献

[1] Judith F.Briesmeister, MCNP - A General Monte Carlo N-Particle Transport Code, Version 4A.

[2] 第１０回原子力コード開発利用ワークショップ，ＲＩＳＴ，1998 年 5 月.

[3] Computer 情報 No.65, 1998 年 6 月.

5. おわりに

計算科学技術推進センター情報システム管理課で実施しているVPP500 (一部AP3000) 向けの原子力コードの高速化作業は，毎年10数件を順調にこなし，平成9年度に14件の作業を完了，平成10年度にも15件の作業が計画されている．

本編では，上記作業の内，平成9年度に導入されたAP3000への移植作業を中心に記述した．本作業は，原研の計算機へのコードのインストール及び計算機環境に合わせたコードの改良 / 修正を行うもので，コンパイラ固有のサービスルーチンや指示行の変更，計算機アーキテクチャの違いによる計算精度のチェック等が主な作業である．

本報告書では，沸騰水型原子炉熱水力解析コードTRAC-BF1，連続エネルギー粒子輸送モンテカルロコードMCNP4AのAP3000への移植作業，及び汎用図形処理解析システムIPLOT用ライブラリの改良作業について記述した．本報告書がこれらの仕事に携わる人々に多少なりとも参考になれば幸いである．

謝　辞

本作業を行う上で，作業を依頼された 伝熱流動研究室 大貫晃氏 (2章)，伝熱流動研究室 秋本肇氏（3章）には，コード内容の把握に際し御協力頂きました．また，本報告書の作成に当り数値実験グループの渡辺正氏には御指導と御助言をいただきました．さらに，本作業を円滑に遂行するための各種事務処理については山田圭子氏に御協力を頂きました．ここにこれらの方々に感謝の意を表します．最後に，本報告書を執筆する機会を与えて下さいました計算科学技術推進センター長竹田辰興氏，情報システム管理課長藤井実氏，(株) 富士通R&Dシステム部長平沢健一氏に感謝致します．

This is a blank page.

国際単位系（SI）と換算表

表1　SI基本単位および補助単位

量	名称	記号
長さ	メートル	m
質量	キログラム	kg
時間	秒	s
電流	アンペア	A
熱力学温度	ケルビン	K
物質量	モル	mol
光度	カンデラ	cd
平面角	ラジアン	rad
立体角	ステラジアン	sr

表2　SIと併用される単位

名称	記号
分，時，日	min，h，d
度，分，秒	°，′，″
リットル	l，L
トン	t
電子ボルト	eV
原子質量単位	u

1 eV=1.60218×10^{-19}J

1 u=1.66054×10^{-27}kg

表3　固有の名称をもつSI組立単位

量	名称	記号	他のSI単位による表現
周波数	ヘルツ	Hz	s^{-1}
力	ニュートン	N	$m\cdot kg/s^2$
圧力，応力	パスカル	Pa	N/m^2
エネルギー，仕事，熱量	ジュール	J	N·m
工率，放射束	ワット	W	J/s
電気量，電荷	クーロン	C	A·s
電位，電圧，起電力	ボルト	V	W/A
静電容量	ファラド	F	C/V
電気抵抗	オーム	Ω	V/A
コンダクタンス	ジーメンス	S	A/V
磁束	ウェーバ	Wb	V·s
磁束密度	テスラ	T	Wb/m^2
インダクタンス	ヘンリー	H	Wb/A
セルシウス温度	セルシウス度	℃	
光束	ルーメン	lm	cd·sr
照度	ルクス	lx	lm/m^2
放射能	ベクレル	Bq	s^{-1}
吸収線量	グレイ	Gy	J/kg
線量等量	シーベルト	Sv	J/kg

表4　SIと共に暫定的に維持される単位

名称	記号
オングストローム	Å
バーン	b
バール	bar
ガル	Gal
キュリー	Ci
レントゲン	R
ラド	rad
レム	rem

1 Å=0.1nm=10^{-10}m

1 b=100fm^2=10^{-28}m^2

1 bar=0.1MPa=10^{5}Pa

1 Gal=1cm/s^2=10^{-2}m/s^2

1 Ci=3.7×10^{10}Bq

1 R=2.58×10^{-4}C/kg

1 rad=1cGy=10^{-2}Gy

1 rem=1cSv=10^{-2}Sv

表5　SI接頭語

倍数	接頭語	記号
10^{18}	エクサ	E
10^{15}	ペタ	P
10^{12}	テラ	T
10^{9}	ギガ	G
10^{6}	メガ	M
10^{3}	キロ	k
10^{2}	ヘクト	h
10^{1}	デカ	da
10^{-1}	デシ	d
10^{-2}	センチ	c
10^{-3}	ミリ	m
10^{-6}	マイクロ	μ
10^{-9}	ナノ	n
10^{-12}	ピコ	p
10^{-15}	フェムト	f
10^{-18}	アト	a

(注)

1．表1－5は「国際単位系」第5版，国際度量衡局1985年刊行による。ただし，1 eVおよび1 uの値はCODATAの1986年推奨値によった。

2．表4には海里，ノット，アール，ヘクタールも含まれているが日常の単位なのでここでは省略した。

3．barは，JISでは流体の圧力を表わす場合に限り表2のカテゴリーに分類されている。

4．EC閣僚理事会指令ではbar，barnおよび「血圧の単位」mmHgを表2のカテゴリーに入れている。

換算表

力	N(=10^{5}dyn)	kgf	lbf
	1	0.101972	0.224809
	9.80665	1	2.20462
	4.44822	0.453592	1

粘度　1 Pa·s(N·s/m^2)=10 P(ポアズ)(g/(cm·s))

動粘度　1m^2/s=10^{4}St(ストークス)(cm^2/s)

圧力	MPa(=10bar)	kgf/cm^2	atm	mmHg(Torr)	lbf/in^2(psi)
	1	10.1972	9.86923	7.50062×10^3	145.038
	0.0980665	1	0.967841	735.559	14.2233
	0.101325	1.03323	1	760	14.6959
	1.33322×10^{-4}	1.35951×10^{-3}	1.31579×10^{-3}	1	1.93368×10^{-2}
	6.89476×10^{-3}	7.03070×10^{-2}	6.80460×10^{-2}	51.7149	1

エネルギー・仕事・熱量	J(=10^7 erg)	kgf·m	kW·h	cal(計量法)	Btu	ft·lbf	eV
	1	0.101972	2.77778×10^{-7}	0.238889	9.47813×10^{-4}	0.737562	6.24150×10^{18}
	9.80665	1	2.72407×10^{-6}	2.34270	9.29487×10^{-3}	7.23301	6.12082×10^{19}
	3.6×10^6	3.67098×10^5	1	8.59999×10^5	3412.13	2.65522×10^6	2.24694×10^{25}
	4.18605	0.426858	1.16279×10^{-6}	1	3.96759×10^{-3}	3.08747	2.61272×10^{19}
	1055.06	107.586	2.93072×10^{-4}	252.042	1	778.172	6.58515×10^{21}
	1.35582	0.138255	3.76616×10^{-7}	0.323890	1.28506×10^{-3}	1	8.46233×10^{18}
	1.60218×10^{-19}	1.63377×10^{-20}	4.45050×10^{-26}	3.82743×10^{-20}	1.51857×10^{-22}	1.18171×10^{-19}	1

1 cal= 4.18605J（計量法）

= 4.184J（熱化学）

= 4.1855J（15℃）

= 4.1868J（国際蒸気表）

仕事率　1 PS(仏馬力)

= 75 kgf·m/s

= 735.499W

放射能	Bq	Ci
	1	2.70270×10^{-11}
	3.7×10^{10}	1

吸収線量	Gy	rad
	1	100
	0.01	1

照射線量	C/kg	R
	1	3876
	2.58×10^{-4}	1

線量当量	Sv	rem
	1	100
	0.01	1

(86年12月26日現在)

原子力コードのVPP500におけるベクトル化、並列化及び移植（移植編）
―平成9年度作業報告書―