FDPS講習会の手引

谷川衝、岩澤全規、細野七月、似鳥啓吾、村主崇行、牧野淳一郎

平成 27 年 7 月 23 日

目 次

1 プログラム

- 13:00 – 14:00 FDPS の講義
  - イントロダクション (牧野淳一郎)
  - 概要説明 (谷川衝)
  - FDPS 詳細 1 – API と内部構造 (岩澤全規)
  - FDPS 詳細 2 – サンプルコード解説 (細野七月)
  - Q&A
- 14:00 – 15:30 FDPS の実習
  - FDPS のインストール
  - サンプルコードの使用 1 (重力 N 体シミュレーションコード)
  - サンプルコードの使用 2 (SPH シミュレーションコード)
- 15:30 – 17:00 FDPS 使用に関する相談

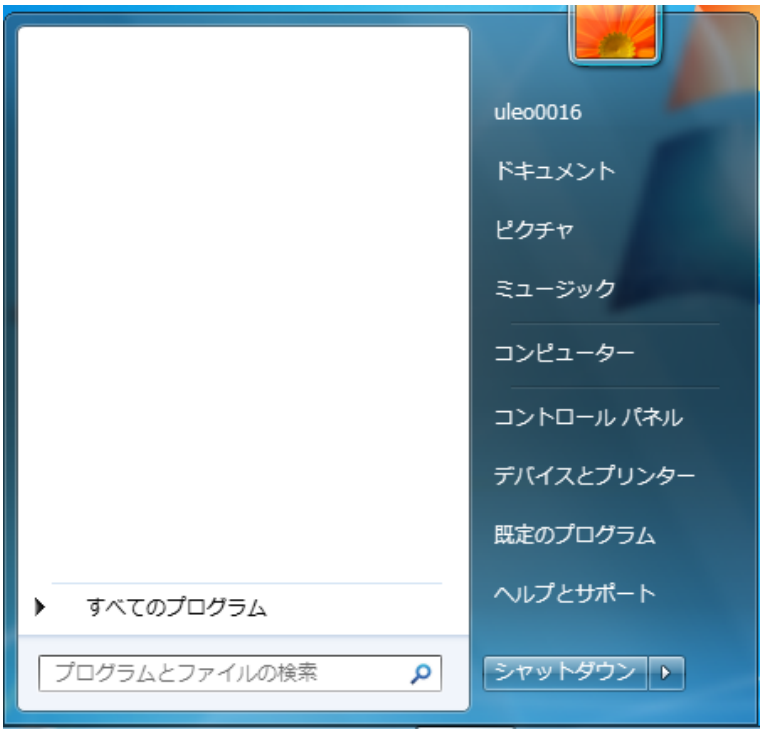

図 1:

2　概要

今回はこんな事をして FDPS を使ってもらいます

3　FDPS の実習

3.1　準備

3.1.1　FOCUS の計算機を用いる場合

まずは、FOCUS の計算機にログインするまでの手順を紹介する。講習用 PC の電源をつけたら、デスクトップ左下のスタートメニューボタンを押し、ウィンドウズメニューを開く。正しく開けたら図 1 のようになるはずである。

次に、Cygwin-X というフォルダを開き、XWin Server を起動する (図 2 参照)。起動に成功したら、XWin Server が図 3 の様に起動するはずである。

次に、計算用ノードにログインを行う。今回の実習では PuTTY というソフトウェアを使う。スタートメニューから、PuTTY というフォルダをクリックする (図 4)。フォルダが開くはずなので PuTTY をクリックし、起動する (図 5)。

PuTTY が起動すると、図 6 の様なメニューが開かれるはずである。この、“HostName (or IP address)” と書いてあるボックスに、「ff01.j-focus.jp」または「ff02.j-focus.jp」の好きな方を書く (図 7)。次に、左側のメニューの SSH の左側の+をクリックし、X11 を選択する。そ

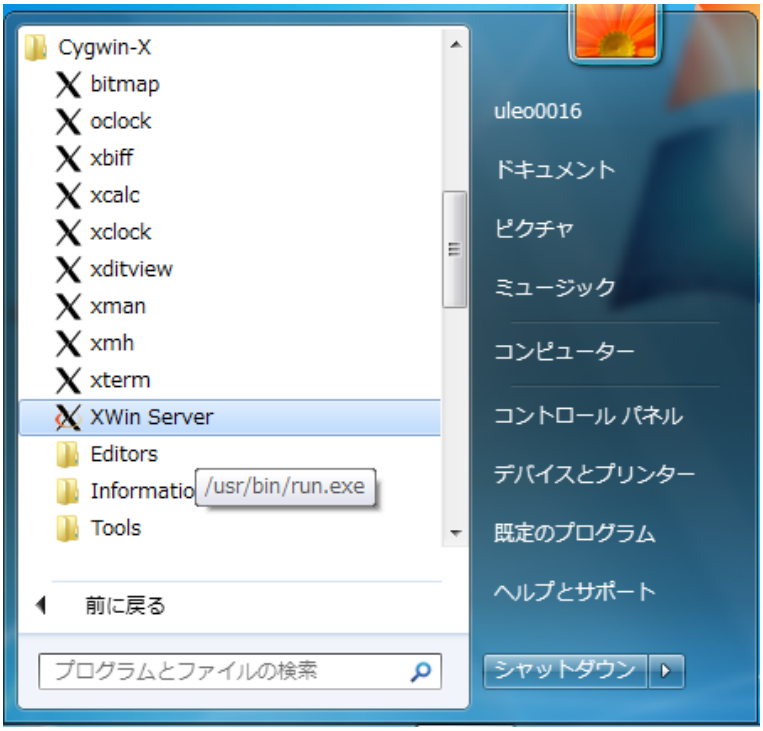
Cygwin-X
bitmap
oclock
xbiff
xcalc
xclock
xditview
xman
xmh
xterm
XWin Server
Editors
Informatio
/usr/bin/run.exe
Tools
前に戻る
プログラムとファイルの検索
uleo0016
ドキュメント
ピクチャ
ミュージック
コンピューター
コントロール パネル
デバイスとプリンター
既定のプログラム
ヘルプとサポート
シャットダウン

図 2:

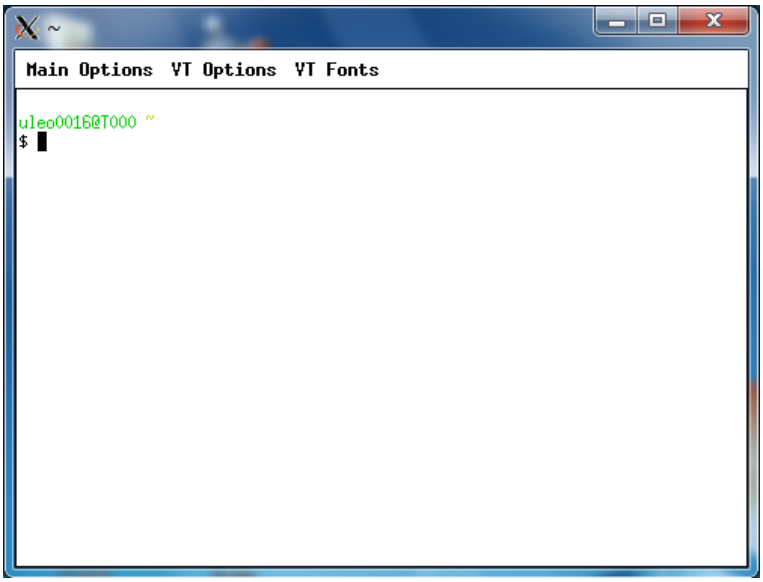
~
Main Options VT Options VT Fonts
uleo0016@T000 ~
$

図 3:

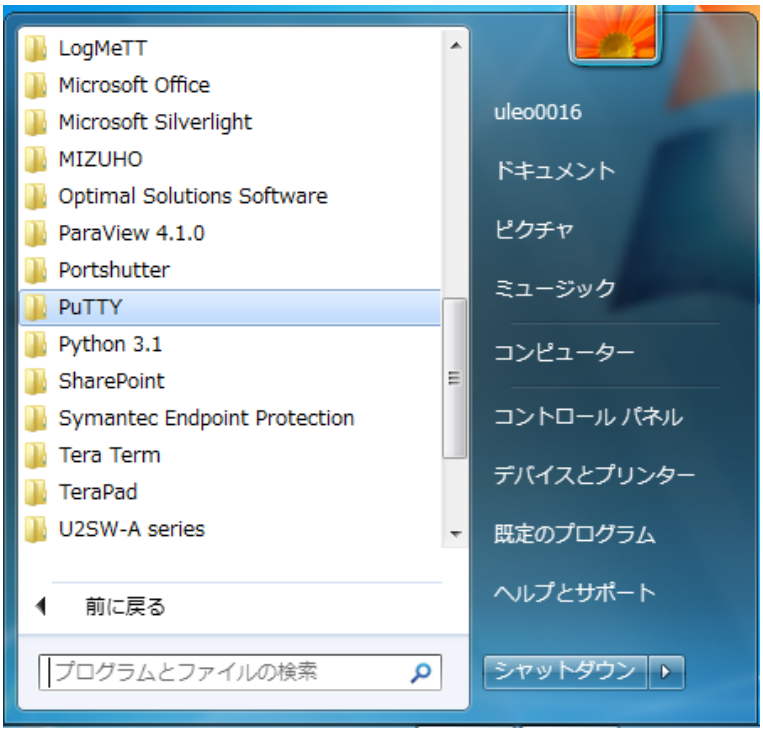
LogMeTT
Microsoft Office
Microsoft Silverlight
MIZUHO
Optimal Solutions Software
ParaView 4.1.0
Portshutter
PuTTY
Python 3.1
SharePoint
Symantec Endpoint Protection
Tera Term
TeraPad
U2SW-A series
前に戻る
プログラムとファイルの検索
uleo0016
ドキュメント
ピクチャ
ミュージック
コンピューター
コントロール パネル
デバイスとプリンター
既定のプログラム
ヘルプとサポート
シャットダウン

図 4:

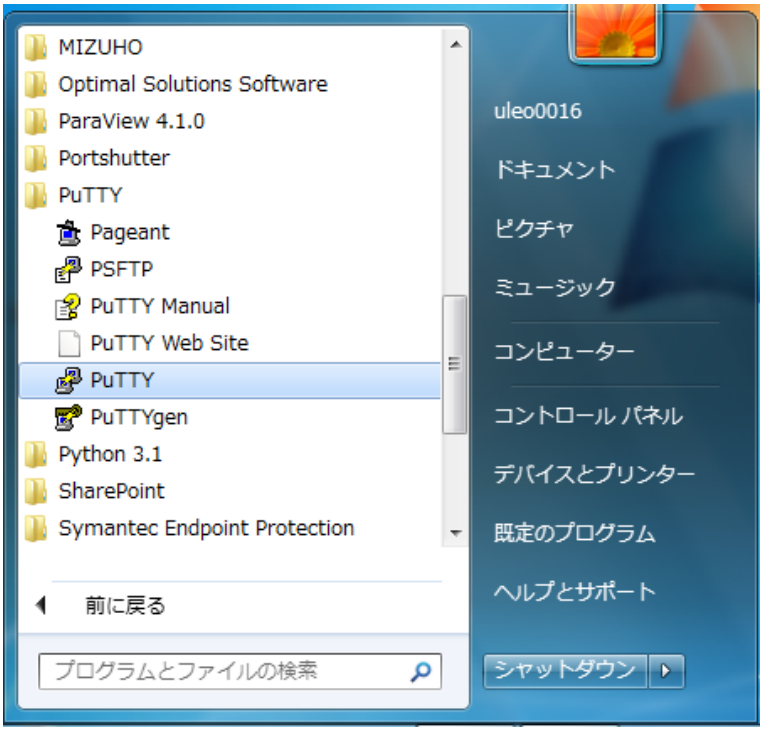
MIZUHO
Optimal Solutions Software
ParaView 4.1.0
Portshutter
PuTTY
Pageant
PSFTP
PuTTY Manual
PuTTY Web Site
PuTTY
PuTTYgen
Python 3.1
SharePoint
Symantec Endpoint Protection
前に戻る
プログラムとファイルの検索
uleo0016
ドキュメント
ピクチャ
ミュージック
コンピューター
コントロール パネル
デバイスとプリンター
既定のプログラム
ヘルプとサポート
シャットダウン

図 5:

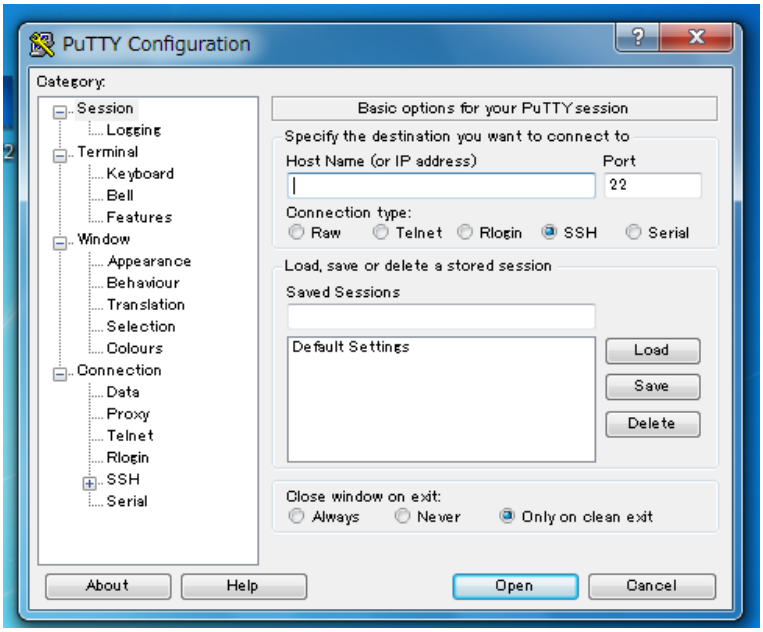

図 6:

こにある “Enable X11 forwarding” の左のチェックボックスをオンにする (図 8)。これらが終わったら、Open ボタンを押す。

Open を押すとログイン画面が開かれる (図 9)。ユーザー名とパスワードを入力すればログインは完了である。

FOCUS の計算機にログインしたら、以下のコマンドを実行する。

```
$ module load gnu/openmpi165
```

以下のコマンドを実行し、FDPS を自分のカレントディレクトリにコピーする (「$」はコマンドプロンプトであるので、「$」を打ち込む必要はない)。

```
$ cp -r ../share/FDPS-master .
```

以上により、ディレクトリ FDPS-master がコピーされる。以下ではディレクトリ FDPS-master があるディレクトリの名前を fdps とする。

3.1.2 自分で用意した計算機で実行する場合

https://github.com/FDPS/FDPS から FDPS の最新版をダウンロードし、好きなディレクトリ下で解凍する。これによってディレクトリ FDPS-master が出来る。以下ではディレクトリ FDPS-master があるディレクトリの名前を fdps とする。

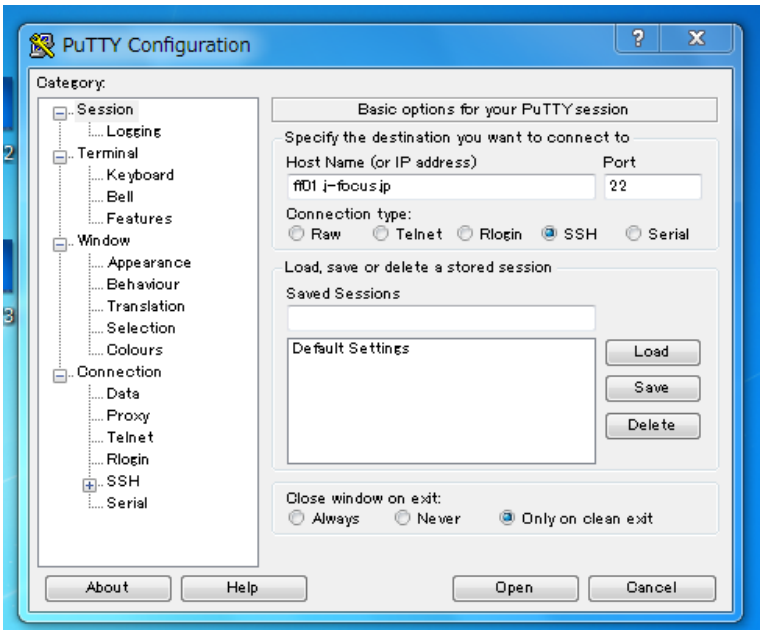
PuTTY Configuration
Category:
Session
Logging
Terminal
Keyboard
Bell
Features
Window
Appearance
Behaviour
Translation
Selection
Colours
Connection
Data
Proxy
Telnet
Rlogin
SSH
Serial
Basic options for your PuTTY session
Specify the destination you want to connect to
Host Name (or IP address)
Port
ff01 j-focus.jp
22
Connection type:
Raw
Telnet
Rlogin
SSH
Serial
Load, save or delete a stored session
Saved Sessions
Default Settings
Load
Save
Delete
Close window on exit:
Always
Never
Only on clean exit
About
Help
Open
Cancel

図 7:

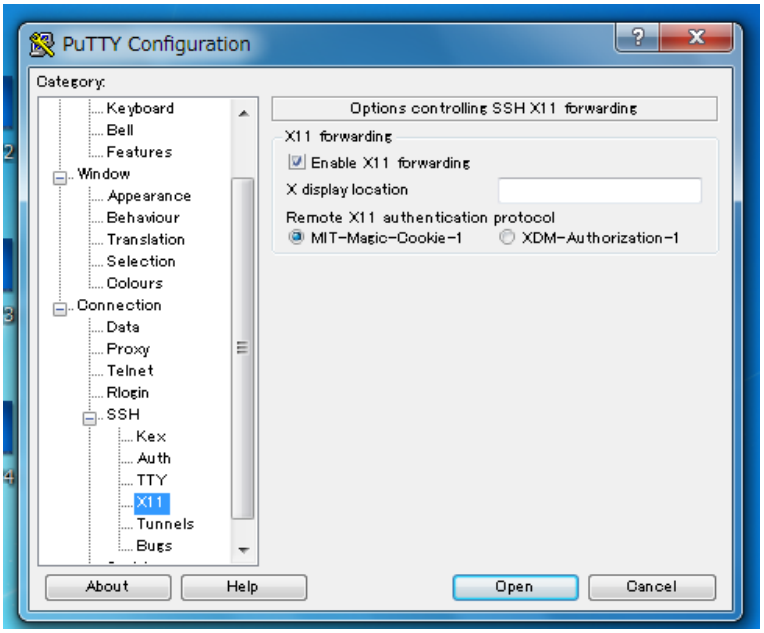
PuTTY Configuration
Category:
Keyboard
Bell
Features
Window
Appearance
Behaviour
Translation
Selection
Colours
Connection
Data
Proxy
Telnet
Rlogin
SSH
Kex
Auth
TTY
X11
Tunnels
Bugs
Options controlling SSH X11 forwarding
X11 forwarding
Enable X11 forwarding
X display location
Remote X11 authentication protocol
MIT-Magic-Cookie-1
XDM-Authorization-1
About
Help
Open
Cancel

図 8:

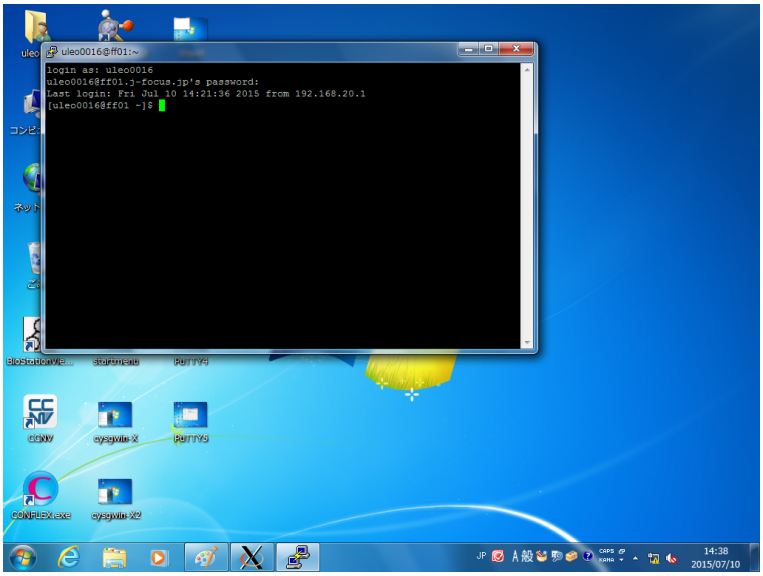

図 9:

3.2 実習本番

実習で行うことは、FDPS を使って実装された重力 N 体シミュレーションコードと SPH シミュレーションコードを使用することである。最初に重力 N 体シミュレーションコード、次に SPH シミュレーションコードを使用する。

なお、実習の際に `Makefile` を更新し、コンパイルし直すという作業を何回か行うが、ここで注意しなくてはならないのは、**`Makefile` を編集しただけでは実行ファイルの再作成は行われない**ということである。この場合、きちんと前回作った実行ファイルを明示的に "`$ rm ./nbody.out`" などで消す必要がある。これを忘れた場合、「`make: ‘nbody.out’` は更新済みです」と出る。

3.2.1 重力 N 体シミュレーションコード

ここでは、重力 N 体シミュレーションコードでの cold collapse を、並列環境無し、OpenMP を用いた並列計算環境、OpenMP + MPI を用いた並列計算環境の 3 つで行う。後者の 2 つに関しては FOCUS スパコンに計算を用いる。

3.2.1.1 概要

ここでは、用意された重力 N 体シミュレーションコードを動かしてみよう。このコードは、重力多体系のコールドコラプスを計算する。この節でまず行うことは、シリアルコードのコンパイルと実行、出て来た結果の解析である。次にシリアルコードを Phantom-GRAPE を用いて高速化して、その速さを体験しよう。最後に OpenMP や MPI を利用して、さらにコードを高速化する。

3.2.1.2 シリアルコード

以下の手順で本コードを使用できる。

- ディレクトリ fdps/FDPS-master/sample/nbody に移動
- make を実行
- ジョブの投入
- 結果の解析
- OpenMP/MPI の利用 (オプション)

3.2.1.2.1 ディレクトリ移動

ディレクトリ fdps/FDPS-master/sample/nbody に移動する。

```
$ cd fdps/FDPS-master/sample/nbody
```

3.2.1.2.2 make の実行

make コマンドを実行する。

```
$ make
```

3.2.1.2.3 計算の実行

まずは、インタラクティブ実行で計算を実行する。これは、生成された実行ファイルの名前をそのまま実行すればよい。

```
$ ./nbody.out
```

正しくジョブが終了すると、標準入出力の最後には以下のようなログが出力されるはずである。energy error は絶対値で 1×10^{-3} のオーダーに収まっていればよい。

```
time:  9.5000000 energy error: -3.804653e-03
time:  9.6250000 energy error: -3.971175e-03
time:  9.7500000 energy error: -3.822343e-03
time:  9.8750000 energy error: -3.884310e-03
******** FDPS has successfully finished. ********
```

ただし、後述する Phantom-GRAPE を用いた場合、energy error 数値は変わるので注意する。

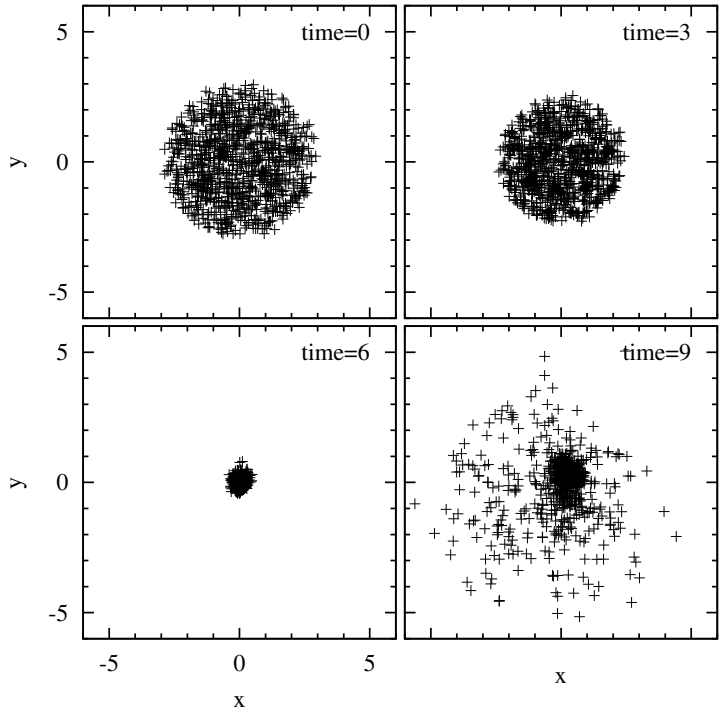

図 10:

3.2.1.2.4 結果の解析

ディレクトリ `result` に粒子分布を出力したファイル"000*.dat"ができている。*は 0 から 9 の値で、時刻を表す。出力ファイルフォーマットは 1 列目から順に粒子の ID, 粒子の質量、位置の x, y, z 座標、粒子の x, y, z 軸方向の速度である。

ここで実行したのは、粒子数 1024 個からなる一様球 (半径 3) のコールドコラプスである。コマンドライン上で以下のコマンドを実行すれば、時刻 9 における xy 平面に射影した粒子分布を見ることができる。

```
$ gnuplot
$ plot "result/0009.dat" using 3:4
```

他の時刻の粒子分布をプロットすると、一様球が次第に収縮し、その後もう一度膨張する様子を見ることができる (図 10 参照)。

3.2.1.3 Phantom-GRAPE の利用

以下では、相互作用計算に Phantom-GRAPE を使う場合について、記述する。この場合、ユーザーはまずは Phantom-GRAPE のコンパイルを行わなければならない。今回使う Phantom-GRAPE のソースコードは、`fdps/FDPS-master/src/phantom_grape_x86/G5/newton/libpg5/` 以下に存在するので、そこまで移動し、コンパイルを行う。以下は、現在 `sample/nbody/` に

居る場合の例である。

```
$ cd ../../src/phantom_grape_x86/G5/newton/libpg5/
$ make
```

コンパイルが成功したら、元のディレクトリに戻る。次に、Makefile の修正を行う。Makefile の 11 行目に Phantom-GRAPE を使用するか否かを決定しているスイッチが存在している。このスイッチはデフォルトではコメントアウトされて no になっているいるため、以下のようにしてコメントアウトを解除する。

```
use_phantom_grape_x86 = yes
```

無事にコンパイルが通れば、以降の実行・解析の手順は同様である。実行直後に次のような表示がされれば、正しく実行ができている。

```
******** FDPS has successfully begun. ********
./result/t-de.dat
Number of processes: 1
Number of threads per process: 1
rsqrt: MSE = 1.158186e-04,  Bias = 8.375360e-08
(以下省略)
```

3.2.1.4 OpenMP/MPI の利用

OpenMP や MPI を利用する場合について以下に記述する。

以降では、計算の実行の際にはジョブの投入を行うことにする。今回は以下の様なコマンドでジョブの投入を行う。これは OpenMP 実行の際の例である。

```
$ sbatch runOMP.sh
```

正しくジョブが投入されている場合、squeue コマンドを実行すると、例えば以下のようにログが出力される。

```
 JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
200402     d006h    nbody uleo0016 PD       0:00      1 (None)
```

ジョブのキャンセルは以下のコマンドで実行できる。

```
$ scancel JOBID
```

JOBID は上の squeue コマンドで表示された JOBID の番号である (上で言えば、200402)。正しくジョブが終了すると、ファイル stdout.log の最後にログが出力される。

- OpenMP のみ使用の場合

- Makefile の編集
  - マクロ CC に OpenMP 対応の C++コンパイラを代入する。今回の実習の環境では変更する必要は無い。
  - “CFLAGS += -DPARTICLE_SIMULATOR_THREAD_PARALLEL -fopenmp” の行のコメントアウトを外す
- make コマンドを実行する。
- sbatch runOMP.sh と打って実行する。正しく実行された場合、stdout.log に以下のように表示されるはずである。

```
******** FDPS has successfully begun. ********
./result/t-de.dat
Number of processes: 1
Number of threads per process: 4
rsqrt: MSE = 1.158186e-04,  Bias = 8.375360e-08
(以下省略)
```

  見て分かる通り、Number of threads per process: 4 となっている。これで、4 スレッドでの並列計算が行われている事が確認できた。

- OpenMP と MPI の同時使用の場合
  - Makefile の編集
    - マクロ CC に MPI 対応の C++コンパイラを代入する。今回の実習の環境では mpic++にする。
    - “CFLAGS += -DPARTICLE_SIMULATOR_THREAD_PARALLEL -fopenmp” の行のコメントアウトを外す (インテルコンパイラの場合は-fopenmp を外す)
    - “CFLAGS += -DPARTICLE_SIMULATOR_MPI_PARALLEL” の行のコメントアウトを外す
  - make コマンドを実行する。
  - sbatch runMPI.sh と打って実行する。正しく実行された場合、stdout.log 以下のように表示されるはずである。

```
(省略)
******** FDPS has successfully begun. ********
./result/t-de.dat
./result/t-de.dat
Number of processes: 2
Number of threads per process: 4
rsqrt: MSE = 1.158186e-04,  Bias = 8.375360e-08
rsqrt: MSE = 1.158186e-04,  Bias = 8.375360e-08
(以下省略)
```

見て分かる通り、Number of processes: 2となっている。これで、2プロセス4スレッドでの並列計算が行われている事が確認できた。

3.2.2 SPHシミュレーションコード

3.2.2.1 概要

ここでは、SPHシミュレーションコードを動かす。用意されているコードは、断熱スフィアコラプスの計算を行う。この節でまず行うことは、シリアルコードのコンパイルと実行、出て来た結果の解析である。最後にOpenMPやMPIを利用して、さらにコードを高速化する。

3.2.2.2 シリアルコード

以下の手順で本コードを使用できる。

- ディレクトリ fdps/FDPS-master/sample/nbodysph に移動
- make を実行
- ジョブの投入
- 結果の解析
- OpenMP/MPIの利用 (オプション)

3.2.2.2.1 ディレクトリ移動

ディレクトリ fdps/FDPS-master/sample/nbodysph に移動に移動する。

3.2.2.2.2 make の実行

make コマンドを実行する。

3.2.2.2.3 計算の実行

まずは、インタラクティブ実行で計算を実行する。これは、生成された実行ファイルの名前をそのまま実行すればよい。

```
$ ./sph.out
```

正しくジョブが終了すると、標準入出力の最後には以下のようなログが出力されるはずである。

```
(省略)
//================================
time = 7.6861124696015781e-01, dt = 3.5727841270539857e-03
step = 88
//================================
9.9999999999997635e-01
6.6872858041836647e-01
9.7700092080239072e-04
9.7700092080317243e-04
9.7700092080318761e-04
******** FDPS has successfully finished. ********
```

3.2.2.2.4 結果の解析

ディレクトリ `result` にファイルが出力されている。ファイル名は"00**.dat"(* には数字が入る) となっている。ファイル名は時刻を表す。出力ファイルフォーマットは 1 列目から順に粒子の ID、粒子の質量、位置の x, y, z 座標、粒子の x, y, z 軸方向の速度、密度、内部エネルギー、圧力である。

これは 3 次元の Evrard sphere である。以下のコマンドを実行すれば、横軸に r、縦軸に密度の logscale での図が作成される。

```
$ gnuplot
$ set logscale
$ plot "result/0040.dat" using (sqrt($3**2 + $4**2 + $5**2)):9
```

正しい答が得られれば、図 11 のような図を描ける。

3.2.2.3 OpenMP/MPI の利用

OpenMP や MPI を利用する場合を以下に示す。

- OpenMP のみ使用の場合

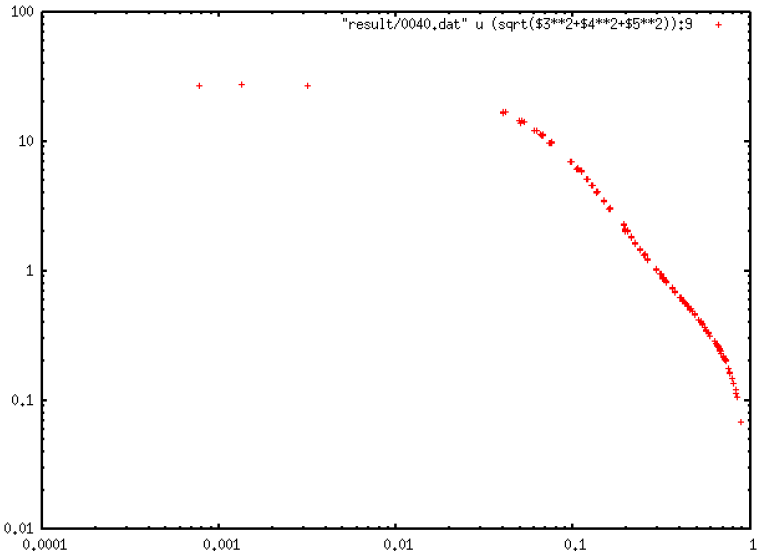

図 11:

- Makefile の編集
  - マクロ CC に OpenMP 対応の C++コンパイラを代入する
  - “CFLAGS += -DPARTICLE_SIMULATOR_THREAD_PARALLEL -fopenmp” の行のコメントアウトを外す
- make コマンドを実行する。
- sbatch runOMP.sh と打って実行する。正しく実行された場合、stdout.log 以下のように表示されるはずである。

```
******** FDPS has successfully begun. ********
//====================================\\
||                                    ||
|| ::::::: ::::::. ::::::. .::::::. ||
|| ::      ::    : ::    : ::       ||
|| ::::::  ::    : ::::::'  `::::. ||
|| ::      ::::::' ::      `.....' ||
||     Framework for Developing     ||
||        Particle Simulator        ||
\\====================================//
//====================================
This is a sample program of Smoothed Particle Hydrodynamics on FDPS!
# of proc is   1
# of thread is 4
//====================================
```

- OpenMP と MPI の同時使用の場合
  - Makefile の編集
    - マクロ CC に MPI 対応の C++コンパイラを代入する
    - “CFLAGS += -DPARTICLE_SIMULATOR_THREAD_PARALLEL -fopenmp” の行のコメントアウトを外す (インテルコンパイラの場合は-fopenmp を外す)
    - “CFLAGS += -DPARTICLE_SIMULATOR_MPI_PARALLEL” の行のコメントアウトを外す
  - make コマンドを実行する。
  - sbatch runMPI.sh と打って実行する。正しく実行された場合、stdout.log 以下のように表示されるはずである。

```
******** FDPS has successfully begun. ********
//==================================\\
||                                  ||
|| ::::::: ::::::. ::::::. .:::::::. ||
|| ::      ::    : ::    : ::       ||
|| ::::::  ::    : ::::::'  `:::::. ||
|| ::      ::::::' ::      `.....' ||
||     Framework for Developing     ||
||        Particle Simulator        ||
\\==================================//
//======================================
This is a sample program of Smoothed Particle Hydrodynamics on FDPS!
# of proc is   2
# of thread is 4
//======================================
```