

ADVENTURE_Forge

HDDM-based rigid-plastic analysis considering contact and friction
between a billet and dies

Version : 0.40 (beta)

User's Manual

March 1, 2002

ADVENTURE Project

目 次

| | |
|---|----|
| 1 . 概要 | 1 |
| 1.1 ADVENTURE_Forge の仕様 | 1 |
| 1.2 動作環境 | 1 |
| 1.3 配布ファイル | 2 |
| 2 . 領域分割と並列化の方法 | 2 |
| 2.1 階層型領域分割 | 2 |
| 2.2 並列化の方法 (Grand, Parent, Child プロセッサの役割) | 3 |
| 3 . コンパイルとインストール | 3 |
| 3.1 コンパイル | 3 |
| 3.2 インストール | 4 |
| 4 . 解析手順 | 4 |
| 4.1 素材メッシュデータの作成 | 5 |
| 4.2 素材メッシュデータに対する境界条件の設定 | 5 |
| 4.2.1 メッシュの表面抽出 (bcGUI のためのデータ準備) | 5 |
| 4.2.2 bcGUI による境界条件の設定 | 6 |
| 4.2.3 材料パラメータデータファイルの作成 | 6 |
| 4.2.4 一体型モデルの生成 (ADVENTURE_Metis のためのデータ準備) | 7 |
| 4.3 素材データの領域分割 | 7 |
| 4.4 金型メッシュデータの作成 | 7 |
| 4.5 入力データファイルの配置とファイル名に関する注意点 | 8 |
| 4.6 advForge の実行 | 8 |
| 4.6.1 mpi run の起動オプション | 9 |
| 4.6.2 advForge の起動オプション | 10 |
| 4.6.3 advForge 実行コマンドの例 | 11 |
| 4.7 解析結果の可視化 | 12 |

付 録

解析ステップ数について

1 . 概要

本ドキュメントはADVENTURE Systemのメイン処理モジュールの1つであるADVENTURE_Forgeの使用マニュアルである。ADVENTURE_Forge は塑性加工（とりわけ鍛造加工）解析を目的とした剛塑性有限要素解析ソルバであり，大規模データを効率的に扱うための基盤技術であるADVENTURE I/Oを使用し，並列化に関しては階層型領域分割法(Hierarchical Domain Decomposition Method :HDDM)を採用している。ADVENTURE_ForgeはADVENTURE Systemの他のコアモジュール(ADVENTURE_BCtool, ADVENTURE_Metis, ADVENTURE_Visual)と連携することによって一連の解析が可能となるため，使用にあたっては他のモジュールのインストールや使用方法の習得が必要となる。他のモジュールのインストール方法や使用方法などは本マニュアルには記述されていないので，各モジュールの使用マニュアルを参考にされたい。なお，本マニュアル中の他のモジュールに関する記述はVer1.0を基準にしており，Ver Betaとは異なる場合があることを付記しておく。

1.1 ADVENTURE_Forge の仕様

ADVENTURE_Forge のおおまかな仕様は以下のようなものである。

- ・圧縮特性法に基づく3次元剛塑性有限要素法
- ・階層型領域分割法による並列計算（単一CPUでの実行も可能）
- ・素材（剛塑性体）と金型（剛体）の接触/摩擦解析
- ・素材は8節点六面体選択低減積分要素，金型表面形状は三角形パッチで表現

1.2 動作環境

- ・OS：UNIX (Linuxを含む)
- ・コンパイラ：C
- ・通信ライブラリ：MPI

(開発環境は CPU: Intel, OS: RedHat Linux7.1, CC: gcc2.96, MPI: mpich1.2.3)

ADVENTURE_Forge は並列計算機上での使用を前提としている。並列計算用の通信ライブラリとしてはMPIを使用するため，MPIのインストールされていないシステムでは使用できない。なお，ネットワークに接続されていない1プロセッサのパソコンであっても，MPIがインストールされていれば仮想的に並列計算を行うことによって実行は可能である(具体的な方法は4.6.1を参照)。

1.3 配布ファイル

```
AdvForge-0.4b/
+-- copyright <---- copyright 説明書
+-- README.jp
+-- src      <---- ソースコード格納ディレクトリ
|   +-- advForge.h
|   +-- advForgeMain.c
|   +-- advForgeFEM.c
|   +-- advForgeCG.c
|   +-- advForgeCommu.c
|   +-- advForgeContact.c
|   +-- advForgeI0.c
|   +-- advForgeFunc.c
|   +-- Makefile
|
+-- samples <---- サンプルデータ格納ディレクトリ
|   +-- model
|   |   +-- advhddm_in_0.adv
|   |   +-- advhddm_in_1.adv
|   |   +-- tool_in.dat
|   |
|   +-- sample.msh
|   +-- material.dat
|   +-- rc_ex      <---- advForge 実行スクリプトの例
|
+-- doc      <---- マニュアル格納ディレクトリ
    +-- manual-jp.pdf <--- 本ファイル
```

2 . 領域分割と並列化の方法

2.1 階層型領域分割

ADVENTURE System のメイン処理モジュールの多くは効率的な並列計算を可能にするため階層型領域分割法を用いている。階層型領域分割については ADVENTURE_Solid のマニュアル第 2 章に詳しいので、まずはそちらを参照されたい。解析領域の階層型への分割として、ADVENTURE System では一階層目の大きな分割単位を「部分 (Part)」と呼び、二階層目の細かい分割単位を「部分領域 (Subdomain)」と呼んでおり、ADVENTURE_Forge もそれに準拠している。したがって、扱える要素が六面体一次要素のみである点を除けば、入力データ形式は他の ADVENTURE System のメイン処理モジュールとほぼ共通である。

2.2 並列化の方法 (Grand, Parent, Child プロセッサの役割)

ADVENTURE_Solid と同じく階層型領域分割法を採用しているが、並列化の方法は若干異なる。ADVENTURE_Forge では動的負荷分散は考慮されておらず、全てのプロセッサは解析前に Grand, Parent, Child の 3 種類のどれかに分類され (詳細は以下参照のこと)、解析途中で親子関係や担当する subdomain が変化することはない。

Grand プロセッサ:

常に 1 台必要。C G 法により subdomain 間境界節点の反復解を求める。C G 法の過程では Parent プロセッサとのみ通信を行う。

Parent プロセッサ:

最低 1 台必要。台数は階層型領域分割における part 数と同じであり、起動時にオプションで指定する (4.6.2 を参照)。解析スタート時点で担当する Child プロセッサが決定され、解析中に担当する Child プロセッサが変化することはない。主に Grand プロセッサと Child プロセッサのデータコミュニケーションの橋渡し役。

Child プロセッサ:

最低限 Parent プロセッサと同じ台数必要。台数は (mpirun -np で指定されたプロセッサ数 - 1 - Parent プロセッサ数) で自動的に決まる。C G 法の過程では、subdomain 間境界節点のデータを Parent プロセッサから受け取り、各 subdomain の有限要素計算を行い、結果を Parent プロセッサに返す。また、計算速度を優先するため、解析中は三角分解後の剛性マトリックスの値は常に保持される。

3. コンパイルとインストール

3.1 コンパイル

コンパイルに必要なものは、C コンパイラと MPI のコンパイル環境、ADVENTURE_IO のモジュールである。ADVENTURE_IO はあらかじめコンパイルしておく必要がある。具体的な手順としては、まず、AdvForge-0.4b/src/ というディレクトリに Makefile と ADVENTURE_Forge のソースファイルがあるので、そこに移動する。

```
% cd AdvForge-0.4b/src/
```

次に、make コマンドで実行形式 (advForge) を作成する。

```
% make
```

なお, make の際に advsys-config というスクリプトが自動的に起動されるが, デフォルトで advsys-config のディレクトリは \$(HOME)/ADVENTURE/bin に設定されているので advsys-config が \$(HOME)/ADVENTURE/bin 以外のディレクトリにある場合は, Makefile 内の

```
ADVBINDIR = $(HOME)/ADVENTURE/bin
```

という部分を事前に書き換えておく. 問題なくコンパイルできれば, advForge という実行形式が出来上がる.

3.2 インストール

advForge は実行パスのとあったディレクトリにコピーしておく方が使い易い make install すると, advForge は \$(HOME)/ADVENTURE/bin/ へ, マニュアル類(AdvForge-0.4b/README.jp と本マニュアル) は \$(HOME)/ADVENTURE/doc/AdvForge/ へコピーされる.

```
% make install
```

make install を行わず, cp コマンドなどを用いても差し支えない.

4 . 解析手順

ADVENTURE_Forge の実行過程の概略は以下のようなものである.

- (1) 素材メッシュデータの作成
- (2) 素材データに対する境界条件の設定
- (3) 素材データの領域分割
- (4) 金型メッシュデータの作成
- (5) 入力データファイルの配置
- (6) advForge の実行
- (7) 計算結果の可視化

ADVENTURE_Forge は塑性加工(とりわけ鍛造)解析を目的としており, 加工される材料(素材)に関するデータと素材を加工する工具(金型)に関するデータの両方が必要である. したがって advForge を実行するに当たって必要となる入力データファイルは (a) 階層型領域分割された 1 つあるいは複数の素材データファイルと (b) 1 つの金型データファイルである.(a) は (1) ~ (3) の過程を経て作成され,(b) は (4) で作成する.(5) でそれらを適当なディレクトリに配置し,(6) で解析スタートとなる. 以下で, 上記(1) ~ (7)の詳細と具体的な手順について説明する.

4.1 素材メッシュデータの作成

ASCII 形式のテキストファイルとして素材の全領域に関するメッシュファイルを作成する。素材のメッシュデータに必要な情報は

要素数

(各要素の) コネクティビティ

節点数

(各節点の) 節点座標

であるため、このデータは市販のメッシュジェネレータなどを用いて比較的容易に作成できる。なお、ADVENTURE_Forge がサポートしているのは六面体一次要素のみであり、その他の要素は使用できない。また、節点番号は 1 からではなく、0 から始まることに注意する。以下にその例として AdvForge-0.4b/samples ディレクトリにある sample.msh の内容を示す。

```
1500
  0    66    67     1    11    77    78    12
  1    67    68     2    12    78    79    13
.....
.....
1967 2033 2034 1968 1978 2044 2045 1979
2046
0.0000E+00 -0.5000E+01 0.0000E+00
0.0000E+00 -0.4000E+01 0.0000E+00
.....
.....
0.3000E+02 0.3536E+01 0.3536E+01
```

4.2 素材データに対する境界条件の設定

4.2.1 メッシュの表面抽出 (bcGUI のためのデータ準備)

msh2pch を用いて上記のメッシュデータ (.msh) から .fgr, .pcg, .pch, .trn の 4 つのファイルを作成する (これらのファイルは次の 4.2.2, 4.2.4 の過程で使用される)。各ファイルと mesh2pch の使用法の詳細については ADVENTURE_BCtool のマニュアルを参照されたい。例えば、sample.msh という素材メッシュデータの表面を 45 度の角度でグループ分けしたい場合には以下のようにする (45 = 180 / 4 なので、最後のパラメータとして 4 を入力する)。

```
% msh2pch sample.msh 4
```

こうすると sample_4.fgr, sample_4.pcg, sample_4.pch, sample_4.trn が生成される。

4.2.2 bcGUI による境界条件の設定

bcGUI を用いて境界条件を付加する。素材と金型の接触 / 摩擦は ADVENTURE_Forge の中で自動的に考慮されるようになっており、それ以外の境界条件、すなわち、変形対称面、固定点などの条件をここで付加する。なお、bcGUI は ADVENTURE System の汎用境界条件設定モジュールであり、面荷重、体積力などの境界条件も付与できるが、ADVENTURE_Forge では入力データ中の節点速度以外の境界条件を無視する設定になっているので、ここで付加できる境界条件は節点速度のみであることに注意する。なお、解きたい問題が「素材と金型の接触のみで、特に対称面や固定点などがない場合」は境界条件を設定する必要はないが、領域分割用のデータを作成するためにこの手順は必須である。

```
% bcGUI sample_4.pch sample_4.pcg
```

bcGUI の強制節点変位を設定するための手順で設定された値を、ADVENTURE_Forge では節点速度に読み替える。具体的な手順としては、節点速度を与える面を選択し、“Add Displacement” で強制的な節点速度を与え、“File” メニューの “save Condition” を選択し、sample_4 と入力して “OK” ボタンを押すと sample_4.cnd ファイルが生成される。

4.2.3 材料パラメータデータファイルの作成

ADVENTURE_Forge では硬化則として、以下のような Swift 型硬化則を速度依存形に拡張したものを採用している。

$$\bar{\sigma} = f (\bar{\epsilon}_0 + \bar{\epsilon})^n \dot{\bar{\epsilon}}^m$$

ここで、余ひずみ ϵ_0 は 0.002、ひずみ速度感受性指数 m は 0.01 にあらかじめ設定されており、入力データとして与えることのできる材料パラメータは基準変形抵抗 f と、加工硬化指数 n の 2 つである。以下にその指定方法を説明する。

```
YieldStress          40.0  
HardeningParameter  0.3
```

以上のような ASCII 形式のテキストファイルをエディタ等を用いて作成し、適当な名前（以下では material.dat）で保存する。ここで YieldStress は f を、HardeningParameter は n を指定するためのキーワードである（ f の物理的意味は Yield Stress（降伏応力）とは異なるが、便宜上このキーワードを用いている）。なお、ADVENTURE_Solid は異なる構成式を用いており、同じキーワードでも意味が異なるので注意する。また、 f は応力の単位系を持つが、SI 単位系（MPa）でも工学単位系（kgf/mm²）でもかまわない。しかし、解析の結果得られる応力や反力などの単位系はここで入力した単位系に依存するので注意が必要である。

4.2.4 一体型モデルの生成 (ADVENTURE_Metis のためのデータ準備)

上記までで作成してきた様々なファイルを1つの ADVENTURE I/O 形式のバイナリファイル (.adv ファイル: 以下の例では sample_4.adv) にまとめる。なお, 以下のコマンド入力例は紙面の都合上 2 行にわたっているが, 実際には 1 行に続けて入力する。

```
% makefem sample.msh sample_4.fgr sample_4.cnd material.dat
   sample_4.adv -t sample_4.trn
```

4.3 素材データの領域分割

ADVENTURE_Metis を用いて, 上記で作成したデータ sample_4.adv の階層型領域分割を行う。詳しくは ADVENTURE_Metis のマニュアルを参照されたい。以下の例は全領域を 20 個の subdomain に領域分割し, 10 個の subdomain を 1 つの part として合計 2 part のデータを作成する場合の例である (mpirun コマンドに関しては 4.6 を参照されたい)。

```
% mpirun -np 2 -machinefile ~/machines
   ~/ADVENTURE/bin/adventure_metis ./sample_4.adv . 10
```

こうすることによって, デフォルトでは ./model というディレクトリに advhddm_in_0.adv と advhddm_in_1.adv という 2 つ (作成されるファイルの数は part の数と同じ) のファイルが作成される。

4.4 金型メッシュデータの作成

ASCII 形式のテキストファイルとして金型表面形状のメッシュファイルを作成する。金型データに必要な情報は以下のものである。

解析ステップ数, 金型の最終移動量

金型の数

各金型の移動速度 (x, y, z)

各金型の摩擦係数

各金型の要素数

(各要素の) コネクティビティ

各金型の節点数

(各節点の) 節点座標 (x, y, z)

なお, ADVENTURE_Forge がサポートしている金型の要素は 3 次元三角形一次要素 (三角形パッチ) のみである。その際, コネクティビティに沿って右手系で表現した要素の法線ベクトル方向が金型の外側へ向いていなければならないことと, 節点番号 0 から始まることに注意する。以下にその例を示す (AdvForge-0.4b/samples ディレクトリにある tool_in.dat)。

```

#NumberOfSteps_&_PunchStroke
10  2.0
#DieCavity
1
0.0  0.0  -1.0
0.3
2
  0    1    3
  1    2    3
4
10.0 -10.0  5.0
10.0  10.0  5.0
20.0  10.0  5.0
20.0 -10.0  5.0

```

4.5 入力データファイルの配置とファイル名に関する注意点

advForge を実行するに当たって必要な入力データは「1つあるいは複数の素材データファイル」と「1つの金型データファイル」であり、上記までで作成した例では以下のものである。

```

(素材データ) advhddm_in_0.adv, advhddm_in_1.adv
(金型データ) tool_in.dat

```

各入力データファイルは、そのファイルを読み込むプロセッサが決められているので、これらのデータファイルをプロセッサが読むことのできる適当な場所に配置しなければならない。全てのプロセッサがファイルサーバなどに接続されており、ファイルがシェアリングされていれば、この作業は容易になる。

素材データはいくつかの part に分割されているが、これらは Parent プロセッサが読み込む。ここで上記 advhddm_in_0.adv というファイル名の advhddm_in の部分に関してはユーザーが適当な名称に変更しても差し支えないが、_0.adv の部分に変更されると advForge は正常に動作しないので注意されたい。金型データは Grand プロセッサが読む。このデータファイル名は拡張子も含めてユーザーが全て自由に決めることができる。

4.6 advForge の実行

advForge は並列計算機上での使用を前提としており、通信ライブラリとしては MPI を使用している。したがって、基本的には mpirun コマンドを用いて以下のような方法で起動する。

```
% mpirun mpirun_options ~/ADVENTURE/bin/advForge advForge_options
```

ここで、*mpirun_options* は *mpirun* に対する起動オプション、*advForge_options* は *advForge* に対する起動オプションである。ただし、*advForge* のバージョンを表示するだけの場合は *mpirun* を用いないで、以下のようにする。

```
% advForge -version
```

4.6.1 *mpirun* の起動オプション

ホームページ <http://www-unix.mcs.anl.gov/mpi/www/www1/mpirun.html> などに詳しい記述があるので、ここでは最低限必要な起動オプションについてのみ紹介する。

```
-np n
```

使用するプロセッサの台数を *-np* オプションで指定する。*n* は使用するプロセッサの総数であり、1 + Parent プロセッサ数 + Child プロセッサ数である。Parent プロセッサ数=1 ならば *n* は 3 以上、Parent プロセッサ数=2 ならば *n* は 5 以上でなければならないことに注意する（詳細は 2.2 参照）。

```
-machinefile machinefileName
```

何台目にどのプロセッサを割り当てるかを *-machine_file* オプションで指定する。*machinefileName* は割り当てるプロセッサ名を記述したファイルの名前である。例えば全部で 3 台のプロセッサを用い、1 台目として *pe1*、2 台目として *pe2*、3 台目として *pe3* を使いたい場合は以下のように記述したファイルをエディタ等で作成する。

```
pe1  
pe2  
pe3
```

1 台のスタンドアロンマシン *pe1* でプログラムを動かしたい場合は、以下のように全て同じプロセッサ名を書いておくことで実現できる。

```
pe1  
pe1  
pe1
```

4.6.2 advForge の起動オプション

実行時に設定可能なオプションは以下のとおりである .

`-nparent n`

Parent プロセッサの台数を指定する . このオプションは必須である . 各 Parent プロセッサは素材入力データファイルを読み込むので *n* が領域分割された素材入力データファイルの数と一致していなければエラーとなる . また , 8 台以上の Parent プロセッサを使用する場合には , [AdvForge-0.4b/src/advForge.h の 18 行目 :NPA 8] の部分をさらに大きな数に変更してコンパイルし直す必要がある .

`-output-interval m`

解析ステップ *m* 回毎に解析結果をファイルに出力する . デフォルトでは解析ステップの最終回のみ解析結果ファイルを出力する .

`-cg-tol x`

反復法の収束判定のためのトレランスを指定する . デフォルトは 0.00001 .

`-model-file fileName`

素材の入力データファイル名の `_i.adv` (ここで *i* は Part を示す数字) より前の部分を指定する . 最終的に読み込まれる素材入力データファイル名は `fileName_i.adv` になる (ファイル名については 4.5 参照) . なお , `fileName` にディレクトリ名を含めたファイル名を指定すれば , カレントディレクトリ以外にあるファイルも指定できる . デフォルトは `./advhddm_in` である .

`-result-file fileName`

解析結果ファイル名の先頭部分を指定する . 最終的な出力ファイル名は `fileName_i_j.adv` (ここで *i* は Part を示す数字 , *j* は解析ステップを示す数字) が付加されたものになる . `fileName` にディレクトリ名を含めることもできる . デフォルトは `./advhddm_out` である .

`-tool-file fileName`

金型データのファイル名を指定する . なお , このファイル名は `-model-file` , `-result-file`

と異なり,拡張子なども含めたフルネームで指定する *fileName* にディレクトリ名を含めることもできる. デフォルトは `./tool_in.dat` である.

```
-avs-file fileName
```

デフォルトでは ADVENTURE_Visual 用の解析結果ファイルだけでなく, MicroAVS の「複数ステップ非構造格子型データ」形式に対応した解析結果ファイルも出力される. このオプションでそのファイル名を指定する. MicroAVS の複数ステップ非構造格子型データファイルの拡張子は `.inp` でなければならないので, *fileName* で指定するのは拡張子より前の部分である. *fileName* にディレクトリ名を含めることもできる. デフォルトは `./advForge_avs` である.

```
-no-output-avs
```

MicroAVS 用の解析結果ファイルを出力しないようにする.

4.6.3 advForge 実行コマンドの例

以下に advForge の実行コマンドの一例 (AdvForge-0.4b/samples/rc_ex) を示す. Grand プロセッサ 1 台, Parent プロセッサ 2 台, Child プロセッサ 4 台の合計 7 台を用いて計算し, データは全て `./model` というディレクトリに置き, 計算結果は ADVENTURE_Visual と MicroAVS のファイルを 5 ステップ毎に出力する場合の例である. ただし, ホームディレクトリに `machines` という名前の MPI の `machinefile` がなかったり, `mpirun` に対してパスが通っていないと正常に動作しない. また, `mpirun` によって起動されるプログラム名 (以下の例では `~/ADVENTURE/bin/advForge`) は, そのプログラムが置いてある場所を含めたフルパス名でなければならない. したがって, `mpirun` を起動するカレントディレクトリに `advForge` が存在しない場合は `advForge` へパスが通っていても `~/ADVENTURE/bin/advForge` を `advForge` のように省略することはできないので注意する. 以下の例は 5 行にわたっているが, 実際には 1 行に続けて書く.

```
% mpirun -np 7 -machinefile ~/machines ~/ADVENTURE/bin/advForge
  -nparent 2 -model-file ./model/advhddm_in
  -result-file ./model/advhddm_out
  -tool-file ./model/tool_in.dat
  -avs-file ./model/advForge -output-interval 5
```

4.7 解析結果の可視化

先にも述べたように ADVENTURE_Visual と MicroAVS 用の解析結果ファイルを出力することが可能である。ADVENTURE_Visual を用いた可視化については ADVENTURE_Visual のマニュアルを参照されたい。MicroAVS を用いた可視化では、MicroAVS を起動し、「ファイル」メニューから「データの読み込み」を選択し、ファイルを読み込む。

付 録

解析ステップ数について

ADVENTURE_Forge では金型のトータル移動量と解析ステップ数を入力データとして与えることにより増分ステップが決まる。解析ステップ数が少ないと計算時間は短くなるが、体積変化が大きくなったり、素材の金型への食い込み量が大きくなったりする。したがって解析ステップ数は問題の種類によって最適に設定する必要がある。しかし、剛塑性有限要素法では弾塑性有限要素法と異なり、各解析ステップにおける応力が応力増分の積分によって求められるのではないため、弾塑性有限要素法と比較してかなり大きな増分ステップをとることができる。極端に大きな増分ステップ(極端に少ない解析ステップ数)をとらない限りは、解析精度に大きな支障は出ない。

解析ステップ数の目安としては1ステップあたりの要素のひずみが2～3%程度にすると良い場合が多い。例えば圧縮率50%の単軸の均一圧縮の場合であれば15～30ステップとなる。圧縮率50%であっても変形の不均一性が大きい場合、場所によってはひずみが100%を超えることも少なくないので50ステップ程度必要となることもある。