

ADVENTURE SYSTEM

ADVENTURE_Metis

**Parallel domain decomposition of mesh into single layered
subdomains or two-layered parts and subdomains using
METIS and ParMETIS as kernel**

Version: 1.1

プログラム使用マニュアル

February 17, 2006

ADVENTURE Project

目次

1	概要	1
2	ファイル入出力	2
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	5
4	操作手順	6
4.1	実行の概要	6
4.2	必須引数	6
4.3	その他のオプション	7
5	ヒント	9
5.1	共有ディスクとローカルディスク	9
5.2	METIS ライブラリ による分割の機種依存性について	9
5.3	最適化リナンバリングについて	9
	Appendix	10
A	動作原理	10
B	出力ファイルフォーマット: HDDM 型解析モデルのデータ構造	11
B.1	ADVENTURE system 領域分割前後のデータの流れ	11
B.2	領域分割後の解析モデル (advhddm.in ファイル) のデータ構造	12
B.3	領域分割後の解析モデル (advhddm.in ファイル) における内部境界の取扱い	16
B.3.1	内部境界とは	16
B.3.2	内部境界を担当する親の決め方	16
B.3.3	内部境界を表現するためのデータ	17
B.4	領域分割後の解析モデル (advhddm.in ファイル) の実例	23
	参考文献	29

1 概要

ADVENTURE_Metis は、ADVENTURE Project [1] において開発中の、階層型領域分割法 (HDDM) を用いた並列処理による有限要素法解析システム (ADVENTURE system) のための領域分割ツール (Domain Decomposer) である。このソフトウェアは、以下の特徴を持っている。

- (1) メッシュを単層に、あるいは2階層に分割する。
- (2) ミネソタ大学の開発したグラフ分割ライブラリ METIS と ParMETIS [2] を用いている (これらは ADVENTURE_Metis ソースアーカイブに含まれており、別途入手する必要はない)。
- (3) FEM メッシュデータ (一体型解析モデル) を入力とし、階層型領域分割法 (HDDM) 用入力データ (HDDM 型解析モデル) を出力する。入出力のデータフォーマットは ADVENTURE file format である。
- (4) 並列処理が可能である。並列化に MPI [3] を用いている。コンパイルと実行には MPI が必要である。
- (5) ADVENTURE Project でリリースされるソルバに入力する解析モデル作成に用いることができる。

ADVENTURE システム全体における ADVENTURE_Metis の位置づけについては固体解析ソルバ ADVENTURE_Solid など、各ソルバのドキュメントを参照されたい。また、[4]、[5] も参照されたい。

2 ファイル入出力

ADVENTURE_Metis は、一体型 FEM 解析モデルファイルを入力し、領域分割型解析モデルを出力する。図 1 に、ADVENTURE_Metis のファイル入出力を図示した。

(1) 一体型 FEM 解析モデル

ADVENTURE_Metis の入力ファイルである。バイナリ形式の ADVENTURE Format ファイルである。ADVENTURE Format とは、FEM において使用されるデータを表現するための汎用フォーマットとして ADVENTURE Project によって提唱されたものである。くわしくは [5] を参照のこと。また、解析モデルの作成方法は ADVENTURE_BCtool のドキュメントを参照されたい。入力ファイルは引数で指定する (後述の必須オプション)。

一体型 FEM 解析モデルの概要については、ADVENTURE_Solid の使用マニュアルに説明があるので、参照されたい。また、ADVENTURE File Format を扱うためのツールについても ADVENTURE_Solid の使用マニュアルに説明がある。

(2) 領域分割型 FEM 解析モデル

ADVENTURE_Metis の出力ファイルである。これも、バイナリ形式の ADVENTURE Format ファイルである。領域分割後の FEM 解析モデルである。ADVENTURE_Metis は、2 段階に領域分割を行うが、出力するファイル数は 1 段階目の領域分割数に等しくなる。出力ファイルは、デフォルトでは advhddm_in_0.adv、advhddm_in_1.adv などとなる。末尾の数字は部分番号 (1 段階目の領域分割により生成した部分の番号) を表す。このファイルの詳細については、付録 B にて述べる。



図 1: ファイルの流れ

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、MPI のコンパイル環境と ADVENTURE_IO モジュールである。MPI がインストールされていない環境では、フリーソフトウェアの MPICH [6] 等を使用することが出来るので、あらかじめインストールしておく。

ADVENTURE_IO はあらかじめコンパイルしておく必要がある。

コンパイルの手順は、

- (1) `./configure [options]`
- (2) `make`

である。どちらもトップディレクトリにて行う。configure は環境に依存する部分を解決し、適切な Makefile を作成するためのシェルスクリプトである。

configure に渡せる主なオプションを以下に示す。ただし、以下で用いるディレクトリ名には絶対パスを指定すること。

- `--prefix=install_dir`
インストール先のトップディレクトリを *install_dir* にする。デフォルトは `$HOME/ADVENTURE` である。
- `--with-advio=directory`
ADVENTURE_IO が上記で指定する (あるいはデフォルトの) *install_dir* 以外にインストールされている場合に、そのインストール先ディレクトリを指定する。
- `--with-mpicc=command`
MPI C コンパイラ名を示す。デフォルトは `mpicc` である。ADVENTURE_Metis のソースのコンパイルには、すべてこれを用いる。見つからない場合はコンパイルできない。
- `--with-mpi-cflags=CFLAGS`
プログラムをコンパイルする場合必要な C のコンパイルオプションを指定する。例えば MPI のインクルードファイルを指定する必要がある場合は、
`--with-mpi-cflags="-I/usr/local/include/mpi"` などと指定する。下記の環境変数 `CFLAGS` と併せて用いる事ができる。
- `--with-mpi-libs=LIBS`
プログラムをリンクする場合必要なオプションを指定する。例えば MPI のライブラリを明示する必要がある場合は、`--with-mpi-libs="-L/usr/local/lib/mpi -lmpi"` などと指定する。下記の環境変数 `LIBS` と併せて用いる事ができる。

- `--enable-optimize`
コンパイル時に最適化を行う。これにより設定されるオプション以外の最適化オプションをつけてコンパイルしたい場合は、下の書式を使用する。
- `--enable-optimize=CFLAGS`
`CFLAGS` を最適化用のオプションとして、コンパイル時に最適化を行う。

また、コンパイルに使用される MPI C コンパイラを変更するには、以下の環境変数が使用できる。

- `MPICC`
MPI C コンパイラ名を設定する。
- `CFLAGS`
MPI C コンパイラに対するオプションを設定する。
- `LIBS`
リンクする必要があるライブラリが他にあれば指定する。

これらは、`./configure` 実行前に `./configure` を実行するシェル中にて設定しておく。

例えば、C shell の場合、

```
% setenv MPICC /usr/local/bin/mpicc
% setenv CFLAGS "-O2 -g -Wall"
% ./configure
```

等とし、Bourne shell の場合、

```
$ MPICC=/usr/local/bin/mpicc
$ export MPICC
$ CFLAGS="-O2 -g -Wall"
$ export CFLAGS
$ ./configure
```

等として設定する。

`configure` スクリプトを使用することで、多くの環境ではコンパイル可能と思われるが、うまくいかない場合には、`Makefile` のサンプルが各ディレクトリに用意してあるのでそれを用いてコンパイルする。それぞれのディレクトリにて `Makefile.sample` を `Makefile` にコピーする。また、トップディレクトリにある `Makefile.sample.in` を各環境に合わせて編集してトップディレクトリにて `make` して頂きたい。

3.2 インストール

configure スクリプトを用いてコンパイルした場合は、トップディレクトリにて、

```
% make install
```

とすることで、作成された実行ファイルおよびマニュアルが *install_directory* 以下にインストールされる。

configure を用いず Makefile.sample を Makefile にコピーしてコンパイルした場合も、トップディレクトリにて

```
% make install
```

を行う。インストール先ディレクトリは Makefile.sample.in 中の INSTALL_BINDIR として定義されている。デフォルトでは、\$HOME/ADVENTURE/bin となっている。またドキュメントのインストール先ディレクトリは Makefile.sample.in 中の INSTALL_DOCDIR で定義され、デフォルトでは、\$HOME/ADVENTURE/doc/AdvMetis となっている。

4 操作手順

4.1 実行の概要

まず、MPI 環境における並列プログラムの一般的な実行の方法について述べる。

MPI を用いて並列化されたプログラムを実行する方法は、MPI の実装によって多少異なる。MPICH などでは、`mpirun` コマンドを用いて並列実行を行う仕様となっている。

```
% mpirun [mpioptions] program_name [options]
```

その他の MPI の実装においては、実行モジュールに対し、MPI のオプションを渡す場合もある。

```
% program_name [mpioptions] [options]
```

ADVENTURE_Metis を起動するには、3 つの必須引数 *model_filename*, *directory_name*, *div_num* を、その他のオプションの後に続けて記述する。従って、ADVENTURE_Metis の起動方法は、次のようになる。

```
% mpirun [mpioptions] ./adventure_metis [options] model_filename directory_name  
div_num
```

あるいは

```
% ./adventure_metis [mpioptions] [options] model_filename directory_name div_num
```

このように、実行方法や MPI のオプション (上例の [*mpioptions*]) の指定方法は実装によって異なるので、それぞれの環境にあった方法を参照されたい。

また、MPICH の実行方法 (ホストの指定方法など) については ADVENTURE_Solid の使用マニュアルの付録に簡単な説明があるので、そちらを参照されたい。

4.2 必須引数

必須引数は、*model_filename*, *directory_name*, *div_num* の 3 つである。実行時には、これらをオプションの後に順に並べて記述する。

- *model_filename*
入力する一体型解析モデルファイルを指定するための引数である。
- *directory_name*
領域分割型解析モデルファイルを出力するディレクトリのトップディレクトリ名を指定するための引数である。ADVENTURE_Metis を実行したディレクトリからの相対パスで指定する。もし、ここで指定したディレクトリが存在しない場合、ADVENTURE_Metis は、指定したディレクトリを自動的に作成する。また、ここで指定したディレクトリ中に `model` とい

う名前のサブディレクトリを作成する。領域分割後の解析モデルは、サブディレクトリ `model` 中に出力される。

- *div_num*

2段階目の領域分割数を指定するための引数である。ADVENTURE_Metis が行う領域分割は、FEM 解析のためのものであり、領域分割数の決め方やその意味は解析と独立に説明することはできない。したがって、領域分割の意味や領域分割数の決め方については ADVENTURE_Solid 使用マニュアルを参照していただきたい。(「ADVENTURE_Solid 使用マニュアル section2 並列方法と領域分割」を参照していただきたい。)ここでは、領域分割数の指定の仕方のみ説明する。

ADVENTURE_Metis では、2段階の階層型の領域分割が可能である。

1段階目の領域数(これを「部分数」とこのドキュメントでは呼び慣わす)は、ADVENTURE_Metis を動作させる際のプロセス数により、指定する。

2段階目の領域数(これを「領域数」とこのドキュメントでは呼び慣わす)は、ADVENTURE_Metis の3番目の必須オプション *div_num* によって指定する。

なお、領域数(2段階目の分割数)は、各部分(1段階目の分割後の領域)あたりの数を指定する仕様となっている。

したがって、例えば部分数を4とし、(各部分あたりの)領域数100として分割する場合、

```
% mpirun -np 4 ./adventure_metis some_model.adv some_model 100
```

と、コマンドラインにおいて指定すればよい。

なお、この例では、一体型解析モデルファイル名(入力)が `some_model.adv` であり、領域分割後の解析モデルファイルを入れるトップディレクトリが `some_model` となるように指定している。また、この例は MPICH(Ver.1.2.0以降)の `mpirun` コマンドを用いたものである。

補足：ADVENTURE_Meits による単層型の領域分割は、1段階目、あるいは2段階目の分割数を1とすることにより、実現可能である。

4.3 その他のオプション

- `-HDDM`

標準的な階層型領域分割を行う(デフォルト)。ADVENTURE_Solid への入力には、このモードを用いる。

- `-MAGNETIC`

ADVENTURE_Magnetic への入力向けの階層型領域分割を行う。

- `-difn n`
領域境界上節点の自由度数を n とする。ADVENTURE_Thermal への入力の場合には 1 を指定する。デフォルト値は 3 である。
- `-decomposed-file filename`
領域分割後のファイル名を指定する。ただし、部分番号を表す数字とその前のアンダーバーと拡張子 (.adv) は除いたものを指定する。
デフォルトは、advhddm_in である。
- `-subdir_name directory`
ファイルを出力するディレクトリのサブディレクトリ名を指定する。領域分割後のファイルはこのサブディレクトリ中に出力される。デフォルトは、model である。
- `-memlimit n`
各プロセッサが使用できるメモリ量を MByte 単位で指定する。動的に確保されたメモリの総計から解放された分を引いた値がこの値を超えるとプログラムは強制終了する。デフォルトでは無制限になっている。
- `-ls filename`
各プロセッサから出力されるログの出力先を *filename* に指定する。デフォルトでは 起動プロセッサの標準エラー出力に出力するようになっている。
- `-ll n`
ll は、Log Level の略である。このオプションは、現在の処理の段階や要した時間などについてのログの詳細の度合を制御する。

-ll 0 : ログを出力しない。
-ll 1 : おおまかなルーチン毎についてのみログを出力する。
-ll 2 : さらに細かいサブルーチンについてもログを出力する。
-ll 3 : 主なループ作業について進行状況をリアルタイムに表示する。
デフォルトでは -ll 2 となっている。
- `-nin n`
入力ファイルを読み込むプロセス数を指定する。デフォルトでは 1 つのプロセスが入力ファイルを読み込むようになっている。
- `-r, -nr`
-r オプションをつけると、部分領域を生成するときに最適化リナンバリングを行う。-nr オプションをつけると最適化リナンバリングは行わない。デフォルトでは行うようになっている。

- -h, -help

簡単なヘルプが表示される。

5 ヒント

5.1 共有ディスクとローカルディスク

ADVENTURE_Metis では、ファイル出力を各プロセスが同時に行う仕様となっている。従って、特に大きな問題をたくさんのプロセッサを用いて分割する場合、NFS 等による共有ディスクに結果を出力するとアクセス集中によって非常に遅くなることがあり得る。このような時には出力ディレクトリをローカルディスクへのシンボリックリンクにするか、出力先のディレクトリをローカルディスクに指定していただきたい。

5.2 METIS ライブラリ による分割の機種依存性について

METIS は実行環境によって、それぞれ異なる分割を返すことがある。ただし ParMETIS では同様の現象はないようである。

結果として、ADVENTURE_Metis による領域分割には、2 段階目の分割において、実行環境依存性がある。

5.3 最適化リナンバリングについて

デフォルトでは、部分領域ごとの最適化リナンバリングが行われる。特に 1 部分領域当りの節点数が多いとき、リナンバリングにかなりの時間を要することがある。最適化リナンバリングを行わない場合は -nr オプションを使っていただきたい。

Appendix

A 動作原理

ADVENTURE_Metis は、基本的なアルゴリズムは SIMD 型の並列実行プロセスである。つまり、全ての並列プロセスが、適時同期を取りつつほぼ同じインストラクションを異なるデータに対して行うことで、分散並列化を実現している。ただし、起動プロセスだけは例外的に master として特殊な仕事を行う。基本的に入力ファイルにアクセスするのは master のみで、データを頭からプロセッサの数で等分した分だけ読んでいき、順次他のプロセスに通信で配分する。これは、入力ファイルが 1 ファイルで供給されるので、全プロセスによる集中アクセスを防ぐためである。(用語「部分」を一段階目の領域分割後の解析モデルについて用い、用語「領域」を二段階目の領域分割後の解析モデルについて用いている。この用法は、このドキュメントを通して用いられている)。

以下に ADVENTUER_Metis の動作の流れを簡単に示す。

- (1) 引数処理、各種初期化
- (2) ファイルオープン
出力ファイルのオープンもこの時点で行う。これは異常終了せざるを得ない状況を、なるべく計算を始める前の早い段階で検出するため。
- (3) Pre-Filter
FEM 入力データから要素コネクティビティを読み込み、FEM 入力データをグラフに変換する。
- (4) Partition1
ParMETIS による分割をおこなう (1 段階目の分割)。
- (5) Converter
Partition1 の結果に従い、要素データの再分配をおこなう。グラフデータを部分固有に修正する。
- (6) Partition2
部分のグラフを METIS により分割する (2 段階目の分割)。
- (7) Post-Filter
ここでは、以下の処理を行う。
 - (a) 入力ファイルから節点データを読み込み、Partition1 の結果に従い節点データを再配分する。また、入力ファイルから指定境界条件を読み込み、同様に再配分する。

- (b) 部分間の内部境界条件を作成する。
節点番号を部分固有の番号にリナンバリングし、部分データを作成する。領域間の内部境界条件作成も行う。
- (c) 部分データを出力する。
- (d) 領域データを出力する。
- (e) 終了作業を行う。
ファイルのクローズ、MPI プロセスの終了などを行う。

B 出力ファイルフォーマット: HDDDM 型解析モデルのデータ構造

B.1 ADVENTURE system 領域分割前後のデータの流れ

ここでは、ADVENTURE_Metis の出力する HDDDM 型解析モデルのデータ構造のファイルフォーマットについて述べる。なお、このファイルを便宜上「advhddm_in ファイル」と呼ぶ。

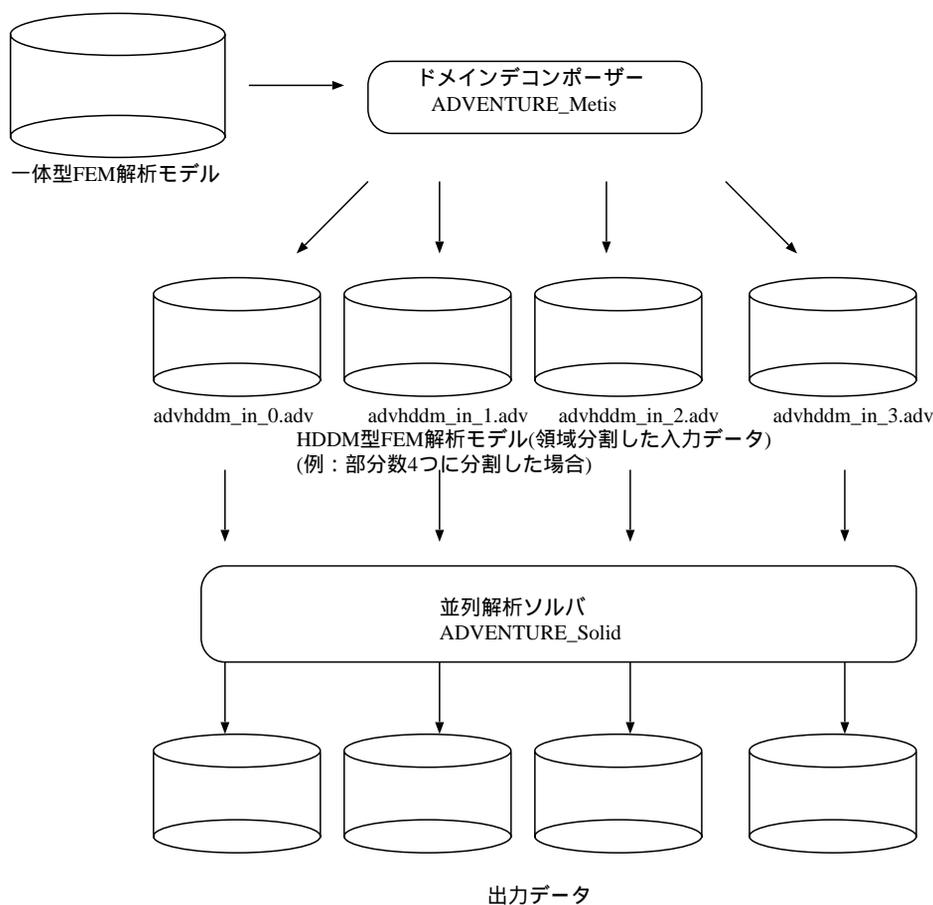


図 2: ADVENTURE_Metis 周辺でのデータの流れ

B.2 領域分割後の解析モデル (advhddm.in ファイル) のデ - タ構造

本システムの FEM 解析ソルバ ADVENTURE_Solid は、階層型領域分割法 (HDDM) を用いている。その入力ファイルを作るため、ADVENTURE_Metis は二段階の領域分割を行う。すなわち、全体を「部分」に分け、次に部分を「領域」に分ける。(用語「部分」を一段階目の領域分割後の解析モデルについて用い、用語「領域」を二段階目の領域分割後の解析モデルについて用いている。この用法は、このドキュメントを通して用いられている)。一つの advhddm.in ファイル、(一段階目の領域分割に相当する) は、一つの「部分」についての情報を持つ。したがって、advhddm.in ファイル一つは、複数の「領域」のデ - タを持つ。

advhddm.in ファイルは ADVENTURE File Format を用いて記述されている。

advhddm.in ファイルでは、上述のように一つのファイルで複数の「領域」のデータを表現するため、「ドキュメント」も複数の領域を表現できるように拡張されている(「ドキュメント」は ADVENTURE FILE Format の用語である)。

以下に、HDDM 型解析モデルのフォーマットを模式的に示す。

HDDM 型解析モデルの含むドキュメントは、次の通りである。

Document の用途	content_type
節点座標	Node
部分節点座標から全体節点座標へのインデックス	FEGenericAttribute
内部境界自由度部分間接続情報	HDDM_InterfaceDOF
要素コネクティビティ	HDDM_Element
領域節点番号から部分節点番号へのインデックス	HDDM_FEGenericAttribute
領域節点番号から全体節点番号へのインデックス	HDDM_FEGenericAttribute
内部境界条件番号	HDDM_FEGenericAttribute
変位境界条件	HDDM_FEGenericAttribute
その他、物性値など	HDDM_FEGenericAttribute

次に、これらのドキュメントのデータ構造を示す。なお、このフォーマットの解説中、'content type'、'Property' といった、ADVENTURE File Format 特有の用語については、参考文献 [5] を参照いただきたい。

また、このフォーマットの解説中、'infree'、'outfree'、'midfree'、'opnu'、'opbd' といった用語は、HDDM における内部境界自由度の取り扱いに関する用語であり、B.3節「領域分割後の解析モデル (advhddm.in ファイル) における、内部境界の取り扱い」にて、詳述する。

< HDDM 型解析モデルデータ構造 >

```
#####
節点座標 (部分節点番号順)
#####
[Properties]
  str <content_type> = "Node"
  int32 <num_items> = この部分の節点数
  int32 <num_items_orig> = 全体節点数
```

```

    int32 <dimension> = 次元
[Data]
    ((float64 座標値) * 次元) * 節点数
-----
#####
部分節点番号から全体節点番号へのインデックス
#####
[Properties]
    str <content_type> = "FEGenericAttribute"
    int32 <num_items> = この部分の節点数
    int32 <num_items_orig> = 全体節点数
    str <fega_type> = AllNodeVariable
    str <format> = i4
    str <label> = NodeIndex_PartToGlobal
    int32 <index_byte> = 4

[Data]
    (int32 全体節点番号) * 節点数
-----
#####
部分要素番号から全体要素番号へのインデックス
#####
ない。
-----
#####
内部自由度部分間接続情報
#####
[Properties]
    str <content_type> = "HDDM_InterfaceDOF"
    int32 <num_parts> = 全部分数
    int32 <dimension> = 節点あたりの内部境界節点自由度
    int32 <num_subdomains> = 部分内領域数
    int32 <part_number> = この部分番号
    int32 <num_outdom> = n_outdom
    int32 <total_num_infree> = t_infree
    int32 <total_num_outfree> = t_outfree
    int32 <total_num_midfree> = t_midfree

[Data]
    ( (int32 <op.n_mofree>) Part 中の op.ibid 数
      ((int32 <op.ibid>) x op.n_mofree)
    ) x num_parts
-----
#####
要素コネクティビティ
#####
[Properties]
    str <content_type> = "HDDM_Element"
    int32 <num_subdomains> = この部分の領域数
    str <element_type> = 要素タイプ
    int32 <num_nodes_per_element> = 1 要素の節点数
    int32 <element_dimension> = 要素の次元
    int32 <element_num_items> = この部分の要素数

[Data]
    ( 領域要素数
      (
        (int32 節点番号) x 1 要素の節点数
      ) x 領域要素数
    ) x 領域数
-----
#####
領域節点番号から部分節点番号へのインデックス
#####
[Properties]
    str <content_type> = "HDDM_FEGenericAttribute"
    int32 <num_subdomains> = この部分の領域数
    int32 <sum_items> = この部分ののべ節点数
    str <fega_type> = AllNodeVariable
    str <format> = i4
    str <label> = NodeIndex_SubdomainToPart
    int32 <index_byte> = 4

[Data]
    ( 領域節点数
      (int32 部分節点番号) x 領域節点数
    ) x 領域数
-----
#####

```

```

領域要素番号から全体要素番号へのインデックス
#####
[Properties]
  str <content_type> = "HDDM_FEGenericAttribute"
  int32 <num_subdomains> = この部分の領域数
  int32 <sum_items> = この部分ののべ要素数
  str <fega_type> = AllElementVariable
  str <format> = i4
  str <label> = ElementIndex_SubdomainToGlobal
  int32 <index_byte> = 4
[Data]
  ( 領域要素数
    ( int32 全体要素番号 ) x 領域要素数
  ) x 領域数
-----
#####
内部境界条件番号
#####
[Properties]
  str <content_type> = "HDDM_FEGenericAttribute"
  int32 <num_subdomains> = この部分の領域数
  int32 <sum_items> = この部分ののべ内部境界条件数
  str <fega_type> = NodeVariable
  str <format> = i4i4i4i4
  str <label> = InterfaceDOF
  int32 <ifd_dimension> = 3
  int32 <index_byte> = 4
[Data]
  ( 領域内部境界条件数
    ( index 領域節点番号
      int32 <coordinate>
      int32 担当部分番号
      int32 担当部分での内部境界条件番号
      int32 この領域が属する部分での内部境界条件番号
    ) x 領域内部境界条件数
  ) x 領域数
-----
#####
変位境界条件
#####
[Properties]
  str <content_type> = "HDDM_FEGenericAttribute"
  int32 <num_subdomains> = この部分の領域数
  int32 <num_items_orig> = 分割前の num_items (全体でのトータル)
  int32 <sum_items> = 部分内ののべアイテム数
  str <fega_type> = NodeVariable
  str <format> = i4f8
  str <label> = ForcedDisplacement
  int32 <index_byte> = 4
[Data]
  ( 領域中のアイテム数
    ( index 領域節点番号
      int32 <coordinate>
      float64 値
    ) x 領域中のアイテム数
  ) x 領域数
-----
#####
HDDM_FEGenericAttribute の一般形
#####
[Properties]
  str <content_type> = "HDDM_FEGenericAttribute"
  int32 <num_subdomains> = この部分の領域数
  int32 <num_items_orig> = 分割前の num_items (全体でのトータル)
  str <fega_type> = (All)(Node|Element)(Constant|Variable)
  str <format> = <value> のフォーマット
  str <label> = 意味のある名前
  int32 <index_byte> = 4
[Data]
All(Node|Element)Variable のとき
  ( 領域中のアイテム数
    <value> x 領域中のアイテム数
  ) x 領域数

All(Node|Element)Constant のとき
  <value>

(Node|Element)Variable のとき

```

(領域中のアイテム数
 (<index>, <value>) x 領域中のアイテム数
) x 領域数

(Node|Element)Constant のとき
 <value>
 (領域中のアイテム数
 <index> x 領域中のアイテム数
) x 領域数

B.3 領域分割後の解析モデル (advhddm_in ファイル) における内部境界の取扱い

B.3.1 内部境界とは

複数の領域によって共有されている節点の自由度を内部境界自由度と呼ぶ。

B.3.2 内部境界を担当する親の決め方

部分間で共有される内部境界自由度については、どの部分が担当するか決めておくことにする。本ソルバの場合、最も部分番号の若い部分が部分間内部境界を担当することに決めておく。図3に、内部境界の取扱いのための例を示す。図3は、48個の四角形要素からなるモデルを4部分8領域に二段階で領域分割した様をあらわす。図3の例では、部分0と部分1の境界上の節点の自由度は部分0が担当する。また、部分1と部分3の境界上の節点の自由度は部分3が担当する。

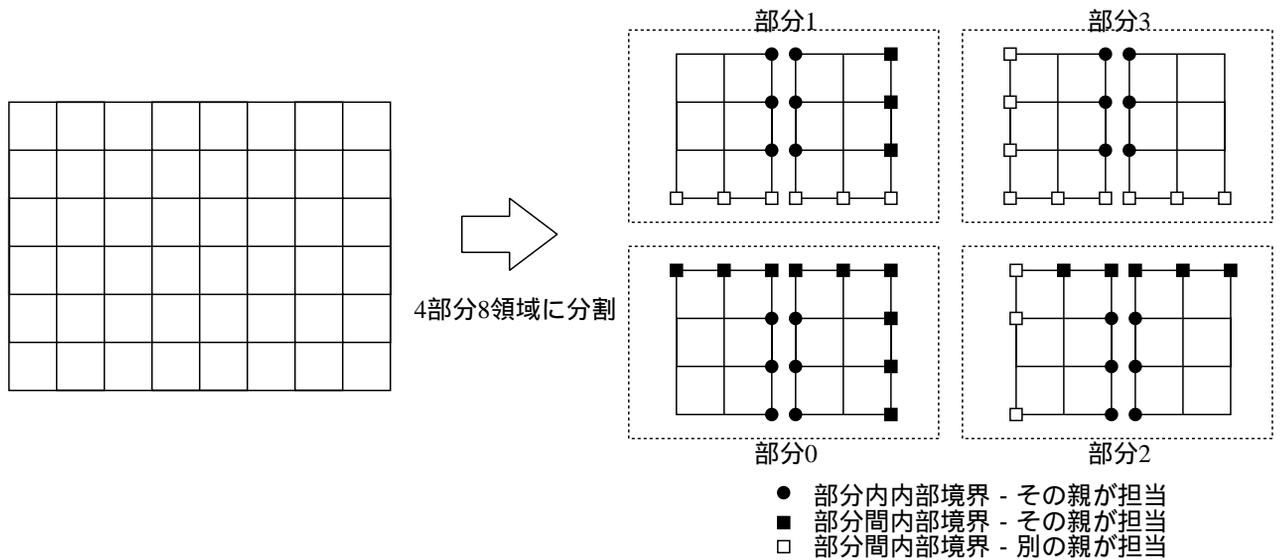


図 3: HDDM における分割と内部境界条件取扱い

B.3.3 内部境界を表現するためのデータ

advhddm_in ファイルにおいては、次表のデータで内部境界自由度を扱う。

変数名 t_infree
変数型 int
説明： 部分の担当する内部境界条件数

変数名 t_outfree
変数型 int
説明： この部分内にあり他の部分の担当する内部境界条件自由度数

変数名 t_midfree
変数型 int
説明： 他の部分内の内部境界条件で、この部分が担当する自由度数

変数名 sum_ninbd
変数型 int
説明： 部分内の各領域の持つ内部境界条件自由度の合計 (重複あり)

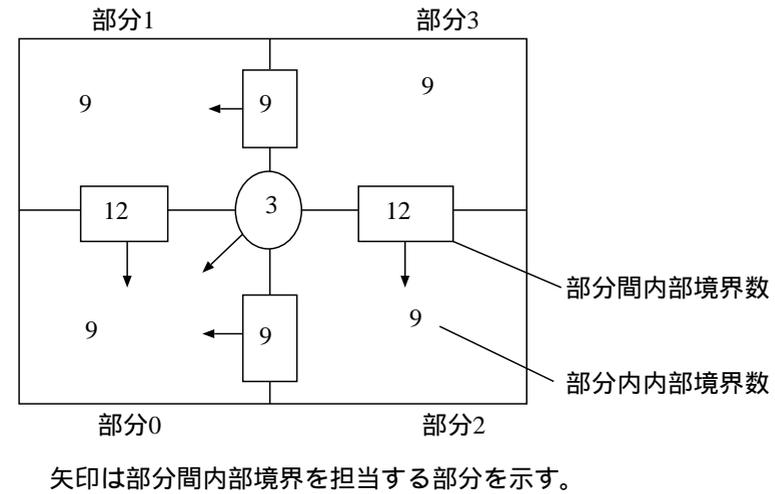
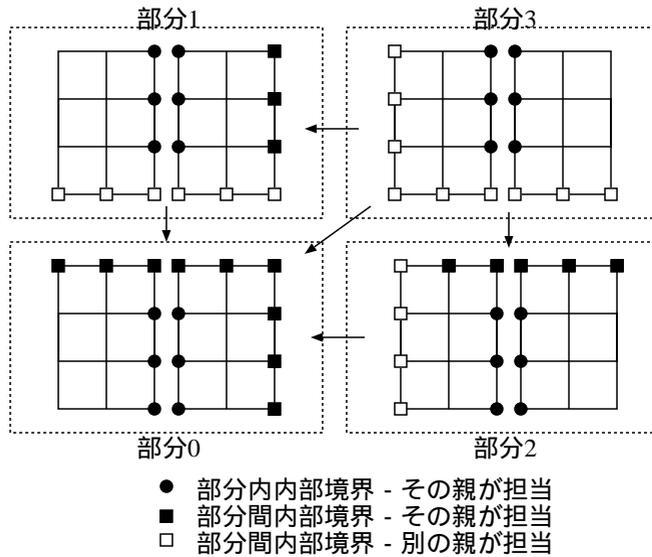
変数名 op[j].n_mofree
変数型 int
説明： 部分 j と共有する内部境界条件自由度数 自分自身と共有する内部境界条件自由度数は 0 であることに注意する

変数名 op[j].ibid[k]
変数型 int
説明： 部分 j と共有する内部境界条件自由度の番号。n_mofree 個存在する。 この番号はこの部分内の通し番号である。 共有する番号を担当するかないにかかわらず数えられる。

なお、変数名と変数型はソルバが読み込む際に用いる変数のものである。

また、上記のデータのうち、t_infree、t_outfree、t_midfree、および op.n_mofree の数え方を具体的に図示した。

図 4: 内部境界自由度数の数え方



内部境界自由度数の数えかた：

- 左図のように領域分割した場合の内部境界自由度の数えかたを右図で模式的にあらわしている。
1. 部分内の内部境界は各部分中に3節点ある。このモデルでは拘束がないため、内部境界自由度は各部分中に $3 \times 3 = 9$ 個存在する。
 2. 2つ以上の部分に共有される内部境界は四角で囲って書いた。1.と同様に、拘束がないため自由度数は節点数 $\times 3$ である。
 3. 3つの部分に共有される節点が一つあり、その節点のもつ自由度数は3である。t_infree、t_outfreeを数える際にこの節点のみ扱いが異なるので注意が必要である。

t_infree: 担当する内部境界自由度数(他の部分と共有しないものも含む)

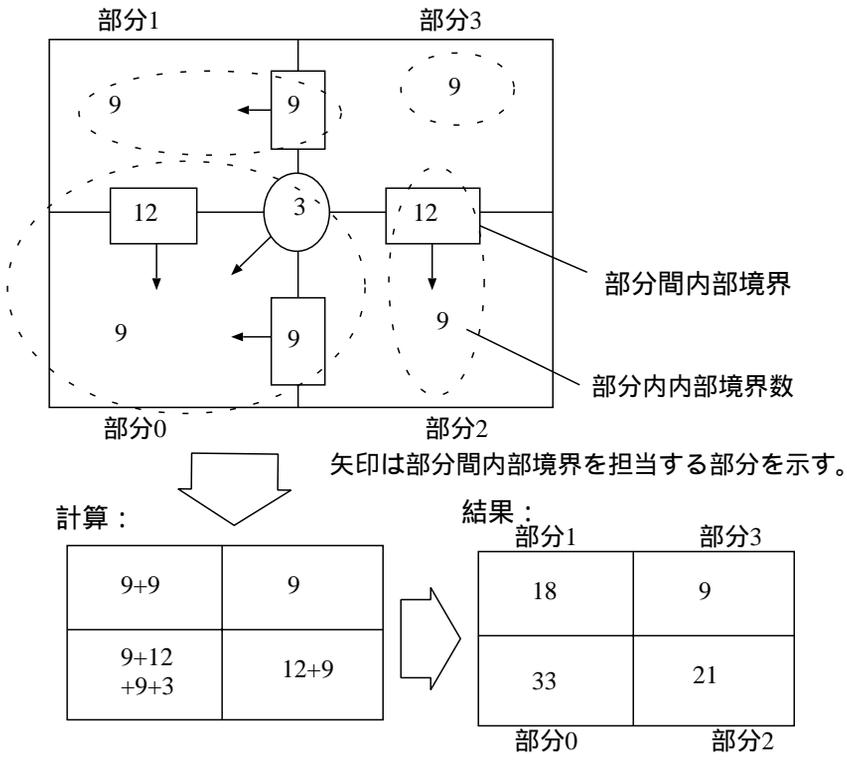


図 5: t_infree の数え方

t_outfree: 他の部分に担当させる内部境界自由度数

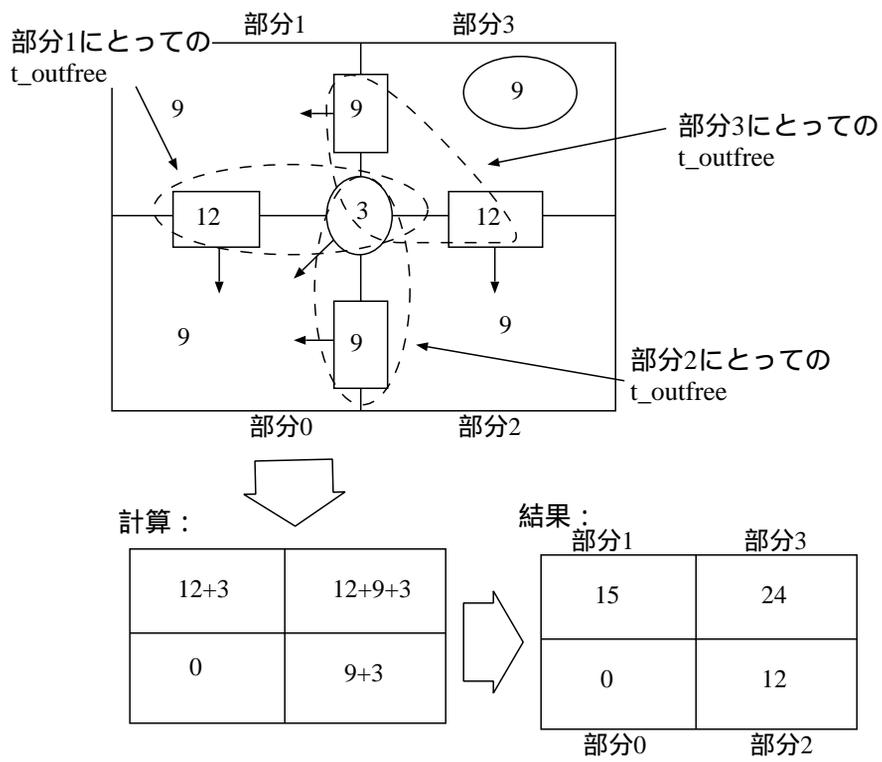


図 6: $t_{outfree}$ の数え方

t_midfree: 自分の担当となる、他の部分上の内部境界自由度数の和(重なりあいあり)

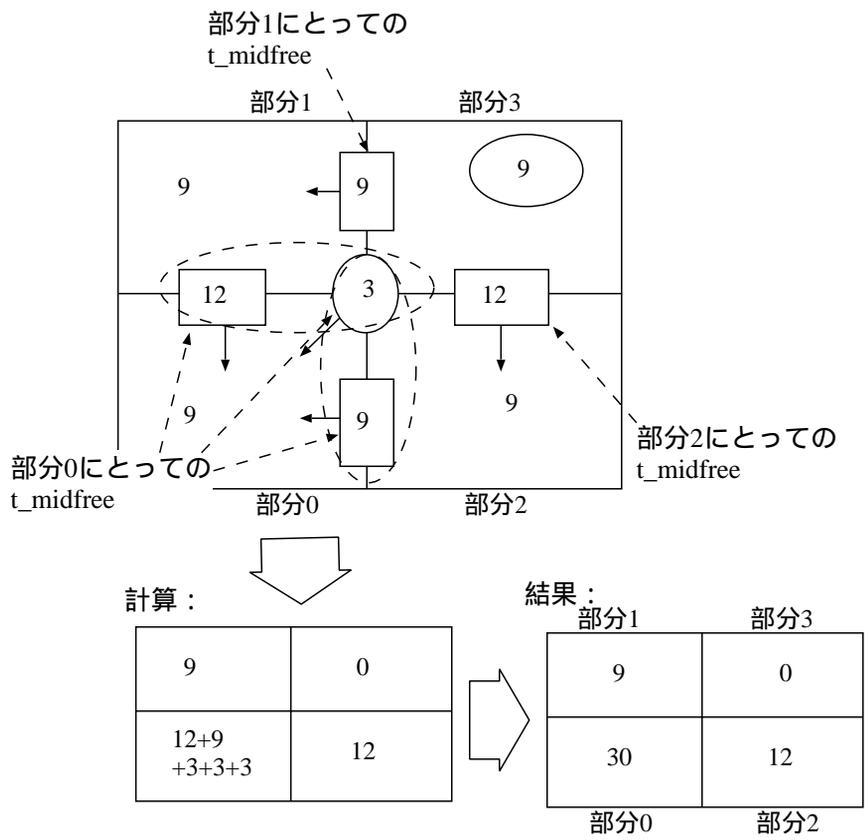


図 7: t_midfree の数え方

t_infree: 担当する内部境界自由度数(他の部分と共有しないものも含む)

9+9	9	=	18	9
9+12 +9+3	12+9		33	21

t_outfree: 他の部分に担当させる内部自由度数

12+3	12+9+3	=	15	24
0	9+3		0	12

t_midfree: 自分の担当となる他の部分上の内部境界自由度の和(重なりあいがある)

9	0	=	9	0
12+9 +3+3+3	12		30	12

op[j],n_mofree (of part No.i) : 部分jと共有する内部境界条件の数
(ただし、どちらかが担当している自由度に限る)

j \ i	0	1	2	3	
0	0	15]	12]	3]	t_outfree(of i)
1	15]	0	0]	9]	
2	12]	0]	0	12]	
3	3]	9]	12]	0	
					t_midfree (of i)

図 8: n_mofree の数え方

B.4 領域分割後の解析モデル (advhddm.in ファイル) の実例

この節では実際の領域分割後の解析モデル (advhddm.in ファイル) ファイルの例を示す。ここに示す例は、ADVENTURE Format のファイルを見るためのツール advshow を用いて advhddm.in ファイルを見た場合のイメージである。このモデルは、ADVENTURE_Solid のアーカイブに含まれている解析モデルと同じである。小さな立方体の解析モデルを、部分数 2、部分あたりの領域数 2 に領域分割した結果である。すでに説明したように、一つのファイルには一つの部分が含まれるので、ここに示したファイル (一つ) には、一つの部分が含まれ、その中には 2 つの領域が含まれる。

```
[Document]
/export/work/adventure/AdvSolid-0.8beta/sample_data/cube_p2d2/model/advhddm.in_0.adv?A8138FE:Node@HDDM_Part[0]:5A85:39D89249
size: 1800
```

```
[Properties]
1: content_type=Node
2: num_items=75
3: num_items_orig=125
4: dimension=3
```

```
[Data]
0: 0.000000e+00 0.000000e+00 0.000000e+00
1: 2.500000e+00 0.000000e+00 0.000000e+00
2: 5.000000e+00 0.000000e+00 0.000000e+00
3: 0.000000e+00 2.500000e+00 0.000000e+00
4: 2.500000e+00 2.500000e+00 0.000000e+00
5: 5.000000e+00 2.500000e+00 0.000000e+00
6: 0.000000e+00 5.000000e+00 0.000000e+00
7: 2.500000e+00 5.000000e+00 0.000000e+00
8: 5.000000e+00 5.000000e+00 0.000000e+00
9: 0.000000e+00 7.500000e+00 0.000000e+00
10: 2.500000e+00 7.500000e+00 0.000000e+00
11: 5.000000e+00 7.500000e+00 0.000000e+00
12: 0.000000e+00 1.000000e+01 0.000000e+00
13: 2.500000e+00 1.000000e+01 0.000000e+00
```

(中略)

```
64: 2.500000e+00 2.500000e+00 1.000000e+01
65: 5.000000e+00 2.500000e+00 1.000000e+01
66: 0.000000e+00 5.000000e+00 1.000000e+01
67: 2.500000e+00 5.000000e+00 1.000000e+01
68: 5.000000e+00 5.000000e+00 1.000000e+01
69: 0.000000e+00 7.500000e+00 1.000000e+01
70: 2.500000e+00 7.500000e+00 1.000000e+01
71: 5.000000e+00 7.500000e+00 1.000000e+01
72: 0.000000e+00 1.000000e+01 1.000000e+01
73: 2.500000e+00 1.000000e+01 1.000000e+01
74: 5.000000e+00 1.000000e+01 1.000000e+01
```

```
[Document]
/export/work/adventure/AdvSolid-0.8beta/sample_data/cube_p2d2/model/advhddm.in_0.adv?1C2655B3:NodeIndex_PartToGlobal@HDDM_Part[0]:5A85:39D89249
size: 300
```

```
[Properties]
1: content_type=FEGenericAttribute
2: num_items=75
3: num_items_orig=125
4: fega_type=AllNodeVariable
5: format=14
6: label=NodeIndex_PartToGlobal
7: index_byte=4
```

```
[Data]
0: 0
1: 1
2: 2
3: 5
4: 6
5: 7
6: 10
7: 11
8: 12
9: 15
```

(中略)

```
71: 117
72: 120
73: 121
74: 122
```

```
[Document]
/export/work/adventure/AdvSolid-0.8beta/sample_data/cube_p2d2/model/advhddm.in_0.adv?56A68AB7:HDDM_InterfaceDOF@HDDM_Part[0]:5A85:39D89249
size: 308
```

```
[Properties]
1: content_type=HDDM_InterfaceDOF
2: num_parts=2
3: dimension=3
```

```
4: num_subdomains=2
5: part_number=0
6: num_outdom=4
7: total_num_infree=105
8: total_num_outfree=0
9: total_num_midfree=75
```

```
[Data]
# Part[0]
# num items in part
0
# Part[1]
# num items in part
75
0: 0
1: 1
2: 2
3: 3
4: 4
5: 5
6: 6
7: 7
8: 8
9: 9
```

(中略)

```
68: 68
69: 69
70: 70
71: 71
72: 72
73: 73
74: 74
```

```
[Document]
/export/work/adventure/AdvSolid-0.8beta/sample_data/cube_p2d2/model/advhddm_in_0.adv?341B2827:HDDM_Element@HDDM_Part[0]:5A85:39D89249
size: 1032
```

```
[Properties]
1: content_type=HDDM_Element
2: num_subdomains=2
3: element_type=3DLinearHexahedron
4: num_nodes_per_element=8
5: dimension=3
6: sum_items=32
7: num_items_orig=64
```

```
[Data]
# SubDomain[0]
# num items in SubDomain
16
0: 27 29 37 33 28 31 41 35
1: 29 30 38 37 31 32 42 41
2: 33 37 40 34 35 41 44 36
3: 37 38 39 40 41 42 43 44
4: 28 31 41 35 18 19 23 21
5: 31 32 42 41 19 20 24 23
6: 35 41 44 36 21 23 26 22
7: 41 42 43 44 23 24 25 26
8: 18 19 23 21 9 10 14 12
9: 19 20 24 23 10 11 15 14
10: 21 23 26 22 12 14 17 13
11: 23 24 25 26 14 15 16 17
12: 9 10 14 12 3 4 6 5
13: 10 11 15 14 4 1 2 6
14: 12 14 17 13 5 6 7 8
15: 14 15 16 17 6 2 0 7
```

```
# SubDomain[1]
# num items in SubDomain
16
0: 27 29 37 33 28 31 41 35
1: 29 30 38 37 31 32 42 41
2: 33 37 40 34 35 41 44 36
3: 37 38 39 40 41 42 43 44
4: 28 31 41 35 18 19 23 21
5: 31 32 42 41 19 20 24 23
6: 35 41 44 36 21 23 26 22
7: 41 42 43 44 23 24 25 26
8: 18 19 23 21 9 10 14 12
9: 19 20 24 23 10 11 15 14
10: 21 23 26 22 12 14 17 13
11: 23 24 25 26 14 15 16 17
12: 9 10 14 12 3 4 6 5
13: 10 11 15 14 4 1 2 6
14: 12 14 17 13 5 6 7 8
15: 14 15 16 17 6 2 0 7
```

```
[Document]
/export/work/adventure/AdvSolid-0.8beta/sample_data/cube_p2d2/model/advhddm_in_0.adv?42A1930E:HDDM_NodeIndex@HDDM_Part[0]:5A85:39D89249
size: 368
```

```
[Properties]
1: content_type=HDDM_FEGenericAttribute
2: num_subdomains=2
3: sum_items=125
4: feqa_type=AllNodeVariable
5: format=i4
6: label=NodeIndex_SubdomainToPart
7: index_byte=4
```

```
[Data]
# SubDomain[0]
# num items in SubDomain
45
```

```

0: 68
1: 62
2: 65
3: 60
4: 61
5: 63
6: 64
7: 67
8: 66
9: 45

(中略)

38: 5
39: 8
40: 7
41: 19
42: 20
43: 23
44: 22
# SubDomain[1]
# num items in SubDomain
45
0: 74
1: 68
2: 71
3: 66
4: 67
5: 69
6: 70
7: 73
8: 72
9: 51

(中略)

39: 14
40: 13
41: 25
42: 26
43: 29
44: 28

[Document]
/export/work/adventure/AdvSolid-0.8beta/sample_data/cube_p2d2/model/advhddm_in_0.adv?7F8062F5:InterfaceDOF@HDDM_Part[0]:5A85:39D89249
size: 3008

[Properties]
1: content_type=HDDM_FEGenericAttribute
2: num_subdomains=2
3: feqa_type=NodeVariable
4: format=I4I4I4I4
5: label=InterfaceDOF
6: index_byte=4
7: ifd_dimension=3
8: sum_items=150

[Data]
# SubDomain[0]
# num items in SubDomain
75
0: 0 0 0 66 66
1: 0 1 0 67 67
2: 0 2 0 68 68
3: 1 0 0 60 60
4: 1 1 0 61 61
5: 1 2 0 62 62
6: 2 0 0 63 63
7: 2 1 0 64 64
8: 2 2 0 65 65
9: 7 0 0 102 102

(中略)

70: 43 1 0 22 22
71: 43 2 0 23 23
72: 44 0 0 84 84
73: 44 1 0 85 85
74: 44 2 0 86 86
# SubDomain[1]
# num items in SubDomain
75
0: 0 0 0 72 72
1: 0 1 0 73 73
2: 0 2 0 74 74
3: 1 0 0 66 66
4: 1 1 0 67 67
5: 1 2 0 68 68
6: 2 0 0 69 69
7: 2 1 0 70 70
8: 2 2 0 71 71
9: 3 0 0 99 99

(中略)

68: 39 2 0 14 14
69: 42 0 0 24 24
70: 42 1 0 25 25
71: 42 2 0 26 26
72: 43 0 0 27 27
73: 43 1 0 28 28
74: 43 2 0 29 29

[Document]
/export/work/adventure/AdvSolid-0.8beta/sample_data/cube_p2d2/model/advhddm_in_0.adv?5D9FFE7F:HDDM_FEGA@HDDM_Part[0]:5A85:39D89249

```

```

size: 872

[Properties]
1: content_type=HDDM_FEGenericAttribute
2: num_subdomains=2
3: num_items_orig=75
4: feqa_type=NodeVariable
5: format=i4f8
6: label=ForcedDisplacement
7: index_byte=4
8: sum_items=54

[Data]
# SubDomain[0]
# num items in SubDomain
27
0: 27 0 0.000000e+00
1: 27 1 0.000000e+00
2: 27 2 0.000000e+00
3: 29 0 0.000000e+00
4: 29 1 0.000000e+00
5: 29 2 0.000000e+00
6: 30 0 0.000000e+00
7: 30 1 0.000000e+00
8: 30 2 0.000000e+00
9: 33 0 0.000000e+00

(中略)

24: 39 0 0.000000e+00
25: 39 1 0.000000e+00
26: 39 2 0.000000e+00
# SubDomain[1]
# num items in SubDomain
27
0: 27 0 0.000000e+00
1: 27 1 0.000000e+00
2: 27 2 0.000000e+00
3: 29 0 0.000000e+00
4: 29 1 0.000000e+00
5: 29 2 0.000000e+00
6: 30 0 0.000000e+00
7: 30 1 0.000000e+00
8: 30 2 0.000000e+00
9: 33 0 0.000000e+00

(中略)

24: 39 0 0.000000e+00
25: 39 1 0.000000e+00
26: 39 2 0.000000e+00

[Document]
/export/work/adventure/AdvSolid-0.8beta/sample_data/cube_p2d2/model/advhddm_in_0.adv?1CA6FOC7:HDDM_FEGA@HDDM_Part[0]:5A85:39D89249
size: 296

[Properties]
1: content_type=HDDM_FEGenericAttribute
2: num_subdomains=2
3: num_items_orig=25
4: feqa_type=NodeVariable
5: format=i4f8
6: label=Load
7: index_byte=4
8: sum_items=18

[Data]
# SubDomain[0]
# num items in SubDomain
9
0: 3 2 -1.000000e-01
1: 4 2 -1.000000e-01
2: 1 2 -1.000000e-01
3: 5 2 -1.000000e-01
4: 6 2 -1.000000e-01
5: 2 2 -1.000000e-01
6: 8 2 -1.000000e-01
7: 7 2 -1.000000e-01
8: 0 2 -1.000000e-01
# SubDomain[1]
# num items in SubDomain
9
0: 3 2 -1.000000e-01
1: 4 2 -1.000000e-01
2: 1 2 -1.000000e-01
3: 5 2 -1.000000e-01
4: 6 2 -1.000000e-01
5: 2 2 -1.000000e-01
6: 8 2 -1.000000e-01
7: 7 2 -1.000000e-01
8: 0 2 -1.000000e-01

[Document]
/export/work/adventure/AdvSolid-0.8beta/sample_data/cube_p2d2/model/advhddm_in_0.adv?17791560:HDDM_FEGA@HDDM_Part[0]:5A85:39D89249
size: 8

[Properties]
1: content_type=HDDM_FEGenericAttribute
2: num_subdomains=2
3: num_items_orig=1
4: feqa_type=AllElementConstant
5: format=f8
6: label=YoungModulus
7: index_byte=4
8: sum_items=1

```

```

[Data]
2.100000e+04

[Document]
/export/work/adventure/AdvSolid-0.8beta/sample_data/cube_p2d2/model/advhddm_in_0.adv?22B722E9:HDDM_FEGA@HDDM_Part[0]:5A85:39D89249
size: 8

[Properties]
1: content_type=HDDM_FEGenericAttribute
2: num_subdomains=2
3: num_items_orig=1
4: feqa_type=AllElementConstant
5: format=f8
6: label=PoissonRatio
7: index_byte=4
8: sum_items=1

[Data]
4.000000e-01

[Document]
/export/work/adventure/AdvSolid-0.8beta/sample_data/cube_p2d2/model/advhddm_in_0.adv?D53DCE0:HDDM_FEGA@HDDM_Part[0]:5A85:39D89249
size: 8

[Properties]
1: content_type=HDDM_FEGenericAttribute
2: num_subdomains=2
3: num_items_orig=1
4: feqa_type=AllElementConstant
5: format=f8
6: label=HardeningParameter
7: index_byte=4
8: sum_items=1

[Data]
1.000000e+03

[Document]
/export/work/adventure/AdvSolid-0.8beta/sample_data/cube_p2d2/model/advhddm_in_0.adv?33E08E29:HDDM_FEGA@HDDM_Part[0]:5A85:39D89249
size: 8

[Properties]
1: content_type=HDDM_FEGenericAttribute
2: num_subdomains=2
3: num_items_orig=1
4: feqa_type=AllElementConstant
5: format=f8
6: label=YieldStress
7: index_byte=4
8: sum_items=1

[Data]
5.000000e+02

[Document]
/export/work/adventure/AdvSolid-0.8beta/sample_data/cube_p2d2/model/advhddm_in_0.adv?77CED430:HDDM_FEGA@HDDM_Part[0]:5A85:39D89249
size: 8

[Properties]
1: content_type=HDDM_FEGenericAttribute
2: num_subdomains=2
3: num_items_orig=1
4: feqa_type=AllElementConstant
5: format=f8
6: label=Density
7: index_byte=4
8: sum_items=1

[Data]
7.600000e+02

[Document]
/export/work/adventure/AdvSolid-0.8beta/sample_data/cube_p2d2/model/advhddm_in_0.adv?26873E3:HDDM_FEGA@HDDM_Part[0]:5A85:39D89249
size: 24

[Properties]
1: content_type=HDDM_FEGenericAttribute
2: num_subdomains=2
3: num_items_orig=1
4: feqa_type=AllElementConstant
5: format=f8f8f8
6: label=Gravity&Acceleration
7: index_byte=4
8: sum_items=1

[Data]
0.000000e+00 0.000000e+00 -9.800000e+00

[Document]
/export/work/adventure/AdvSolid-0.8beta/sample_data/cube_p2d2/model/advhddm_in_0.adv?4852AF07:HDDM_FEGA@HDDM_Part[0]:5A85:39D89249
size: 136

[Properties]
1: content_type=HDDM_FEGenericAttribute
2: num_subdomains=2
3: num_items_orig=64
4: feqa_type=AllElementVariable
5: format=i4
6: label=ElementIndex_SubdomainToGlobal
7: index_byte=4
8: sum_items=32

[Data]
# SubDomain[0]
# num items in SubDomain
16

```

```

0: 0
1: 1
2: 4
3: 5
4: 16
5: 17
6: 20
7: 21
8: 32
9: 33
10: 36
11: 37
12: 48
13: 49
14: 52
15: 53
# SubDomain[1]
# num items in SubDomain
16
0: 8
1: 9
2: 12
3: 13
4: 24
5: 25
6: 28
7: 29
8: 40
9: 41
10: 44
11: 45
12: 56
13: 57
14: 60
15: 61

[Document]
/export/work/adventure/AdvSolid-0.8beta/sample_data/cube_p2d2/model/advhddm_in_0.adv?70B8D367:DocumentList[0]:5A85:39D89249
size: 759

[Properties]
1: content_type=DocumentList
2: num_items=15
3: num_subdomains=2
4: num_parts=2
5: part_number=0
6: label=HDDM_FEA_Model

[Data]
0: A8138FE:Node@HDDM_Part[0]:5A85:39D89249
1: 1C2655B3:NodeIndex_PartToGlobal@HDDM_Part[0]:5A85:39D89249
2: 56A68AB7:HDDM_InterfaceDOF@HDDM_Part[0]:5A85:39D89249
3: 341B2827:HDDM_Element@HDDM_Part[0]:5A85:39D89249
4: 42A1930E:HDDM_NodeIndex@HDDM_Part[0]:5A85:39D89249
5: 7F8062F5:InterfaceDOF@HDDM_Part[0]:5A85:39D89249
6: 5D9FFE7F:HDDM_FEGA@HDDM_Part[0]:5A85:39D89249
7: 1CA6F0C7:HDDM_FEGA@HDDM_Part[0]:5A85:39D89249
8: 17791560:HDDM_FEGA@HDDM_Part[0]:5A85:39D89249
9: 22B722E9:HDDM_FEGA@HDDM_Part[0]:5A85:39D89249
10: D5C0CE0:HDDM_FEGA@HDDM_Part[0]:5A85:39D89249
11: 33E08E29:HDDM_FEGA@HDDM_Part[0]:5A85:39D89249
12: 77CED430:HDDM_FEGA@HDDM_Part[0]:5A85:39D89249
13: 26873E3:HDDM_FEGA@HDDM_Part[0]:5A85:39D89249
14: 4852AF07:HDDM_FEGA@HDDM_Part[0]:5A85:39D89249

```

参考文献

- [1] ADVENTURE Project Home Page: <<http://adventure.q.t.u-tokyo.ac.jp/>>
- [2] METIS Home Page: <<http://www-users.cs.umn.edu/~karypis/metis/>>
- [3] MPI Home Page: <<http://www-unix.mcs.anl.gov/mpi/>>
- [4] G. Yagawa and R. Shioya: Massively Parallel Finite Element Analysis, Asakura-Shoten, (1998) (in Japanese).
- [5] T. Miyamura, S. Tanaka, H. Takubo, S. Yoshimura and G. Yagawa: Standardization of Input/Output Data in Large Scale Parallel Computational Mechanics System, *Internet Transactions of Japan Society for Computational Engineering and Science (JSCES)*, No. 20000028 (2000) (in Japanese),
<<http://homer.shinshu-u.ac.jp/jscs/trans/trans2000/No20000028.pdf>>.
- [6] MPICH Home Page: <<http://www-unix.mcs.anl.gov/mpi/mpich/>>
- [7] G. Yagawa and R. Shioya: Parallel Finite Elements on a Massively Parallel Computer with Domain Decomposition, *Computing Systems in Engineering*, **4**, Nos. 4-6 (1993), 495-503.