ADVENTURE_Thermal

Steady / Non-steady Heat Conduction Analysis with HDDM

Version: 1.0

User's Manual

July, 2005

ADVENTURE Project

Contents

List of Figures

List of Tables

1. Introduction

The current document contains information on the ADVENTURE_Thermal finite element analysis solver designed in ADVENTURE Project [1] for analysis of steady and non-steady heat conduction in solid using Hierarchical Domain Decomposition Method with parallel data processing techniques.

1.1. Program Features

ADVENTURE_Thermal has the following features.

- ADVENTURE_Thermal supports the dynamic load distribution of CPUs in parallel computing environments using the Hierarchical Domain Decomposition method (HDDM).
- ADVENTURE_Thermal supports the Balancing Domain Decomposition (BDD)[8] as CG preconditioner for the HDDM solver.
- ADVENTURE_Thermal supports the single version where all calculations are performed as a single process.
- ADVENTURE_Thermal supports steady and non-steady heat conduction analyses.
- ADVENTURE_Thermal supports linear tetrahedral elements and quadratic tetrahedral elements.
- ADVENTURE_Thermal operates in UNIX and Linux environments.
- ADVENTURE_Thermal uses the Message Passing Interface (MPI) library [6] for parallel data processing.

1.2. Operational Environments

The ADVENTURE_Thermal operates in the following operational environments.

Operating system	Unix, Linux
Data processing library	MPI

1.3. Program Compilation and Installation

To compile the ADVENTUR_Thermal module, you need properly installed MPI

environment and ADVENTURE_IO libraries on your computer. The following procedure should be followed to compile the ADVENTURE_Thermal module.

1.3.1 File Extraction from Archive

The necessary data are contained in AdvThermal-1.0.tar.gz. The directories described in subsection 1.3.2 will be created after decompressing the archive file by using the following command.

gunzip -c AdvThermal-1.0.tar.gz | tar xvf -

1.3.2. Substructures of Directories

After decompressing the AdvThermal-1.0.tar.gz archive file, the directory AdvThermal-1.0 will be created. The contents of AdvThermal-1.0 are shown in the *Table 1*.

Table 1. Contents of Directories

Subdirectory Name	Contents
hddmsrc	Source file of ADVENTURE_Thermal
doc	Documents (Including User's Manual)
tools	Tools for setting up boundary conditions
libfem	Library for finite element method
sample_data	Sample data

Except the directories mentioned in *Table 1*, some files will be created in AdvThermal-1.0 directory for auto configuration.

1.3.3. Compilation Method

(1). Install the ADVENTURE_IO module according to its User manual.
(2). Go to the top directory and execute the following command:

% ./configure
% make
After execution of shell script configure, all necessary computing environment will be recorded into the Makefile.

The shell script configure uses the following options. The absolute path to the top directory should be mentioned.

--with-advio=directory
This option is used to define the top directory of ADVENTURE_IO. Default is

“$HOME/ADVENTURE”.

--with-mpicc=command
This option is used to define the C compiler for MPI. The default is mpicc.
Parallel versions of ADVENTURE_Thermal will not be compiled if the C compiler for MPI is not found.

--prefix=install_dir
This option is used to define the top directory specified by install_dir for program installation. Only the executable modules will be installed in the directory install_dir/bin. The default directory is /$HOME/ADVENTURE.

Other configure options will be described in Chapter 4.

1.3.4 Installation of Executable Module

Execute the command `make install`.

```
% make install
```

The default directory for installation is `$(HOME)/ADVENTURE/`. To change the directory for installation, execute the command

```
% make install prefix=<install_dir>
```

where the option <*install_dir*> should include a full path to the directory for installation.

The following files will be installed.

`bin/advthermal-s`	← Executable module
`bin/advthermal-p`	← Executable module
`bin/advthermal-h`	← Executable module
`bin/mkbc4th`	← Boundary condition setup tool
`bin/makefem_thermal`	← Tool for entire FEA model data
`bin/so2th`	← Data converter tool
`doc/AdvThermal/manual-jp.pdf`	← User’s Manual in Japanese
`doc/AdvThermal/manual-eg.pdf`	← User’s Manual in English
`doc/AdvThermal/README.eucJP`	← Brief information in Japanese
`doc/AdvThermal/README`	← Brief information in English
`doc/AdvThermal/copyright`	← Copyright agreement

1.4. Program Execution

The ADVENTURE_Thermal module can be executed in 3 versions. You do not need mpirun to execute the single mode of ADVENTURE_Thermal. The command of execution of 3 versions is described below.

Single mode

`% advthermal-s` *[options] data_dir*

Parallel mode with static job distribution using MPI

`% mpirun` [*options for* `mpirun]` `advthermal-p` [*options*] *data_dir*

Parallel mode with dynamic job distribution using MPI

`% mpirun` [*options for* `mpirun]` `advthermal-h` [*options*] *data_dir*

The options [*options for* `mpirun]` are specified for the `mpirun`. The options [*options*] are specified for the ADVENTURE_Thermal executable (see *Section 5.2* of the current manual for details). The option *data_dir* should contain a name of the top directory with data files for analysis (input/output directory).

Necessary options (mpirun)

-np ***n*** : the number of machines (corresponding to the number of parts).

-machinefile ***filename*** **:** The files contain the name of network machines.

Necessary options (advthermal-s or advthermal-p or advthermal-h)

The options of 3 modes of ADVENTURE_Thermal will be described in Section 5.2.

2. Parallel Processing and Analysis Solver

ADVENTURE_Thermal can perform the steady and non-steady heat conduction analyses with dynamic load distribution between CPUs using parallel data processing techniques. These features will be described below.

2.1. Parallel Processing

ADVENTURE_Thermal uses the Hierarchical Domain Decomposition method to provide parallel processing of analysis data. An entire-type model is decomposed in two steps (*Figure 1*) by the ADVENTURE_Metis module prior to execution of ADVENTURE_Thermal. A large decomposed unit of the first hierarchy level refers as *Part*, and smaller units of the decomposed *Part* (2^{nd} hierarchy level) refer as *Subdomains*. The details are given in the User's Manual of the ADVENTURE_Metis module. ADVENTURE_Thermal supports several methods of job distribution to use the CPUs in the most efficient way. The Message Passing Interface (MPI) library is used for parallel data processing. The number of processes started at once depends on user-defined environment.

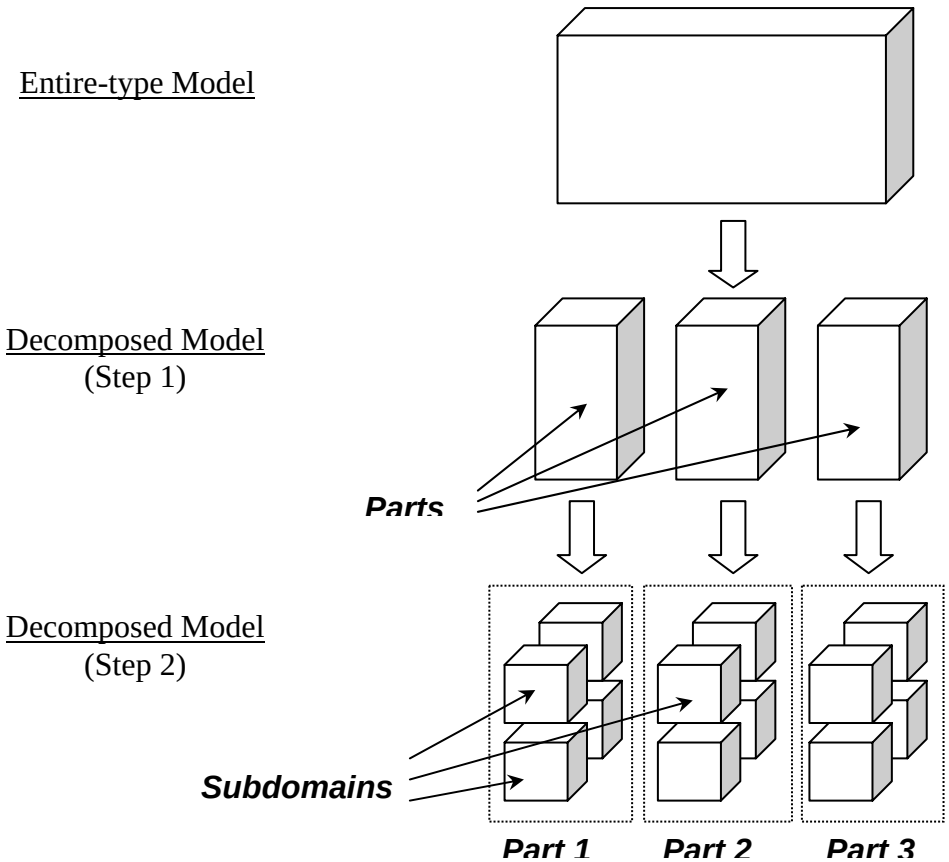


Figure 1. Hierarchical Domain Decomposition

The distributed package contains three versions of ADVENTURE_Thermal.

(1) **Single version (`advthermal-s`)**

A single CPU does all calculations without parallel data processing. The program can be compiled and executed without MPI. There are no limitations on number of "Domains" and "Parts". The model prepared for parallel computation can be used for the single processors without adjustment (*Figure 2*). In the single processor, the computational and data reprocessing procedure for each "Part" occur in the same order as it would be occurred in the parallel computing system. If the parallel computation is not performed well the single version of the program can be used as a checker.

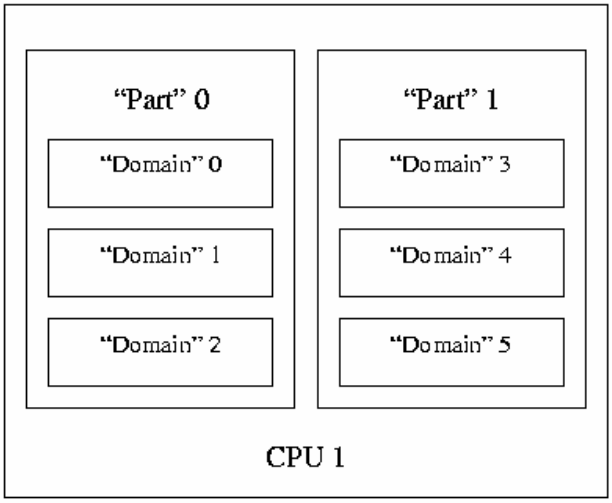


Figure 2. Adjustment of Domain to CPUs (Single version)

(2) **Static job distribution version (`advthermal-p`)**

One CPU treats one *Part* and the processes are statically distributed between CPUs as shown in *Figure 3*. The number of CPUs should correspond to the number of "Parts". This version works efficiently if all nodes have the same performance (uniform system).

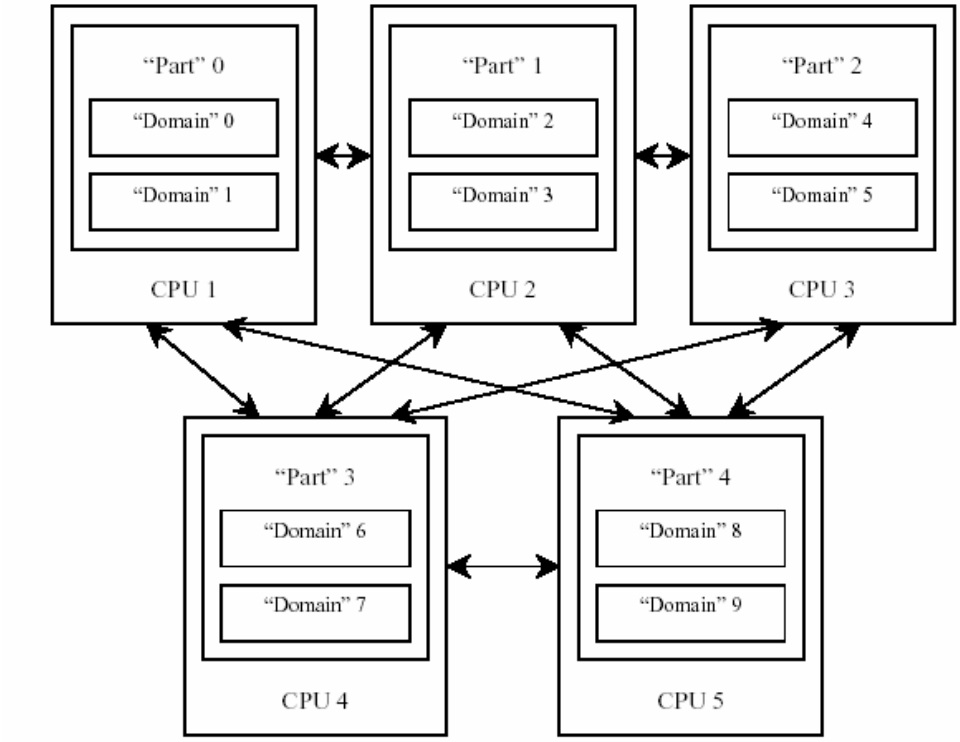


Figure 3. Adjustment of Domains to CPUs (Static load distribution version)

(3) **Dynamic job distribution version (`advthermal-h`)**

The processes are dynamically distributed between CPUs. All CPUs are subdivided into *Parent* CPUs and *Child* CPUs. The *Child* CPUs calculate "Domains" and the *Parent* CPUs collect the calculated information. The number of available CPUs should be more than the number of "Parts". Each "Part" will be assigned to one CPU, and the remained CPUs will be used for calculations of "Domains" (*Figure 4*).

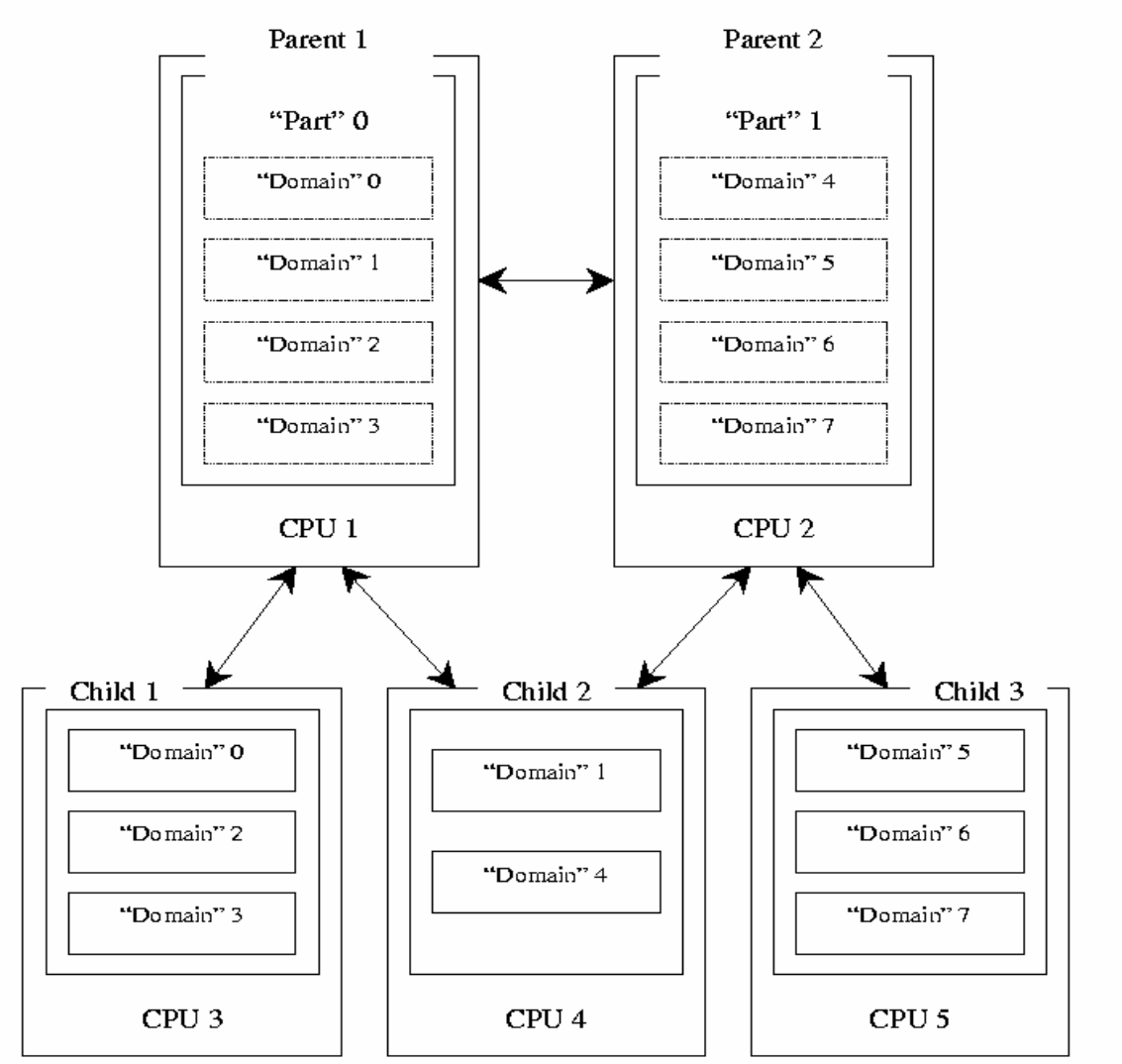


Figure 4. Adjustment of Domains to CPUs (Dynamic load distribution version)

2.2. Characteristics of solver

ADVENTURE_Thermal supports five types of solver to solve the linear systems arising from 3-dimensional heat transfer problems using the HDDM system. This system executes the algorithm of Domain Decomposition Method (DDM) in parallel computer. In DDM first of all, the domain (or perfectly the problem) is split into

subdomains (or subproblems), a problem is defined and solved on each subdomain in parallel and then the partial solutions are glued together to get the global solution. By this way, this method reduces the problem to another one of smaller dimension for the unknowns on interfaces. They are called Schur complement method. Usually the interface problem is solved by Conjugate Gradient method. A distinction between the solvers adopted in ADVENTURE_Thermal is the preconditioning technique which is used for solving the interface problem. A brief description of each solver is given as follows.

HDDM solver:

This solver uses a simplified diagonal scaling in the CG method to solve the interface problem. The user is recommended to use this solver for the problem with the degrees of freedom less than 10,000. This solver is also effective for the solution of a problem with the large number of temperature boundary conditions even the degrees of freedom is larger than 10,000. It requires less memory compared with other solvers. HDDM is the default solver of the current version of ADVENTURE_Thermal.

BDD solver:

The large scale and complicated thermal problem requires large number of iterations to converge. So it is absolutely necessary to reduce the number of iterations with a preconditioning technique especially for large problems. This solver uses a powerful CG preconditioner known as Balancing Domain Decomposition (BDD)[8] to meet this need. BDD is a variation of the Neumann-Neumann preconditioner. It solves a "coarse problem" with few degrees of freedom per subdomain in each CG iteration. For heat conductivity analysis [9], this solver uses one degree of freedom per subdomain to construct the coarse matrix. The coarse matrix is solved by parallel LU decomposition. By using this solver, the number of iteration as well as the computational time are reduced comparing with HDDM solver if the degrees of freedom is larger then 10,000 (investigated through the analysis of a sample problem).

In this solver a preconditioning matrix is made in the first CG loop. So a portion of computational time is consumed to make the preconditioner. Some times it is about 15-25% of total computational time. The time per iteration for the BDD solver is more than that of the HDDM solver. Though time per iteration becomes larger, BDD is an efficient solver as it reduces the total number of iterations. The BDD solver needs more memory then the HDDM solver. Users have high memory computational environment are suggested to use the BDD solver.

BDD must solve a Neumann-Neumann problem in each iteration. In the Neumann-Neumann problem the matrix that represents the subdomain or subproblem may be singular. To overcome these difficulties BDD solver uses a regularization parameter.

Only advthermal-p and advthermal-h supports the BDD solver in this module.

BDD-DIAG solver:

This solver is similar to BDD. But in this solver it does not need to solve the Neumann-Neumann problem in the BDD algorithm. This solver requires less memory than the BDD solver. Depending on the model, it may differ the computational time with that of BDD solver. The users who do not have computational environment with enough memory to use BDD solver are recommended to use BDD-DIAG solver. By testing some model it has been found that the ratio of memory required for BDD-DIAG solver to that of BDD solver is 7/10.

Only advthermal-p and advthermal-h support this solver.

IBDD solver:

This solver is an improved form of the BDD solver. It employs a new coarse problem solver[10] based on an incomplete parallel LU factorization. It reduces the computation time and improves parallel efficiency for large scale problems with large number of subdomains. There are few cases where this solver is used.

Only advthermal-p supports this solver.

IBDD-DIAG solver:

This solver employs a simplified diagonal scaling instead of the Neumann-Neumann method to solve the local problems[10] and a coarse problem solver as used in the IBDD solver. It requires the less computation time, the less memory compared with the BDD and IBDD solver. It may increase the number of iterations. The user may use this solver for the solutions of about 5,000,000 degrees of freedom. This solver is effective for the solutions of problems with large number of subdomains using large number of processors.

Only advthermal-p supports this solver.

2.3. ADVENTURE_Metis

The computational performance of ADVENTURE_Thermal module depends on the proper domain decomposition using the ADVENTURE_Metis. To execute the ADVENTURE_Metis the number of parts and number of subdomains should be determined before. Basically, the number of “Parts” should be decided based on the method used for parallel processing, the number of nodes used in network, and the computing environments. The number of “Domains” should be decided based on the memory used of computational processes. It has been found that as more detailed domain decomposition is done less memory is required. In case of static job distribution (advthermal-p), good performance can be achieved by using BDD or BDD_DIAG if the number of elements in one domain lies 180 to 370 while in case of dynamic job distribution (advthermal-h) the number of element in one domain lies 350 to 450. This range has been found by investigating some test models. For other models this rang may

be semi optimum. The total number of domains does not effect on the number of iterations for BDD and BDD-DIAG solver.

The number of elements in “Domain” that should be created by ADVENTURE_Metis module can be calculated using the following equation.

$$n = N_{element} / (N_{part} * N_{domain})$$

where: n is the number of elements in the considered “Domain”,
$N_{element}$ is the total number of elements,
N_{part} is the total number of “Parts”,
N_{domain} is the total number of “Domains” in the “Parts”.

Compared with the static job distribution method, much data transfer accomplished between the “Parent” and the “Child” in case of dynamic job distribution method. The static job distribution method results in better performance for uniform computer environments.

3. Analysis Algorithm

The algorithm of analysis using the ADVENTURE_Thermal module is shown in *Figure 5*.

(1) Creation of mesh data.
Mesh of the entire-type model data is prepared by ADVENTURE_TetMesh.

(2) Setting of boundary conditions.
Boundary conditions are set to mesh using the pre-processor module ADVENTURE_BCtool. The data of the extracted mesh surface groups are converted into GUI input binary format by using the **msh2pch** command. Then the boundary conditions are set up by the **bcGUI** command. For more details, see the manual of ADVENTURE_BCtool.

Since ADVENTURE_BCtool is based on the ADVENTURE_Solid system, some thermal tools should be used to prepare the data for ADVENTURE_Thermal.

(3) Creation of the entire-type FEA model file.
The boundary conditions and material properties attached to mesh can be saved in an entire-type FEA model of the ADVENTURE binary format. In order to perform this operation **makefem_thermal** tool is used. This tool can save the temperature and flux boundary conditions to the entire-type FEA model data file for ADVENTURE_Thermal. The **mkbc4th** tool is used to create the ADVENTURE format file of the convection and radiation boundary conditions. See *Appendix* for details of those tools. Both tools are distributed with the current version of ADVENTURE_Thermal.

The previous version of ADVENTURE_Thermal (0.5b) uses the **makefem tool of ADVENTURE_BCtool and then a conversion tool **so2th** is used to convert the data for ADVENTURE_Thermal. This tool supports the temperature boundary conditions only. There is an error in this tool to save the flux boundary conditions. So the users are recommended to use the **makefem_thermal** tool which can save both the flux and temperature boundary conditions. Details are given in *Appendix*.

(4) Domain decomposition.

Domain decomposition of the entire-type analysis model is done by ADVENTURE_Metis.

```
% mpirun [mpi_options] adventure_metis -difn 1 [options]
                                    model_filename directory_name div_num
```

The degree-of-freedom used for nodal displacements in static analyses of solids is 3. However, the degree-of-freedom used for temperature in heat conduction analyses should be 1. The necessary option `-difn 1` is used to set the degree-of-freedom for inner boundary nodes to 1.

(5) Heat conduction analysis (ADVENTURE_Thermal)

The HDDM-type model data are analyzed by finite element analysis solver ADVENTURE_Thermal.

(6) Visualization of analysis results (ADVENTURE_Visual or ADVENTURE_Auto)

The analysis results can be visualized using ADVENTURE_Visual. The **advauto_thermalview** of ADVENTURE_Auto can also be used to visualize the temperature distribution of the model.

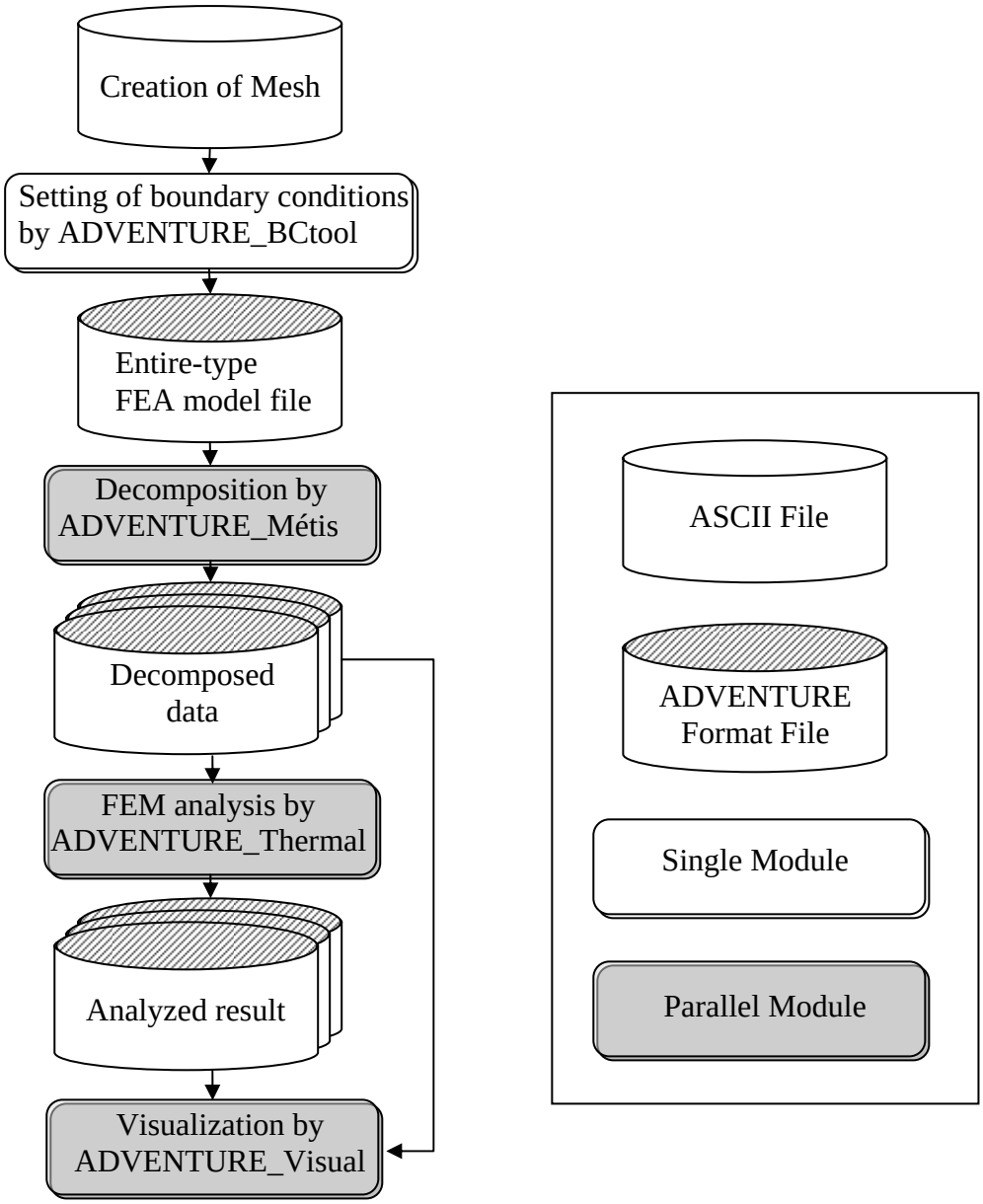


Figure 5. Algorithm of Analysis Using ADVENTURE_Thermal Module.

3.1. Transient Analysis

The backward finite difference approximation and the *Crank-Nicolson* method can be used in transient analyses. The algorithm is shown in *Figure 6*. It includes 2 loops. Time integration iterations are performed by the outer loop and iterative calculations by the CG method based on the hierarchical domain decomposition are performed by the inner loop.

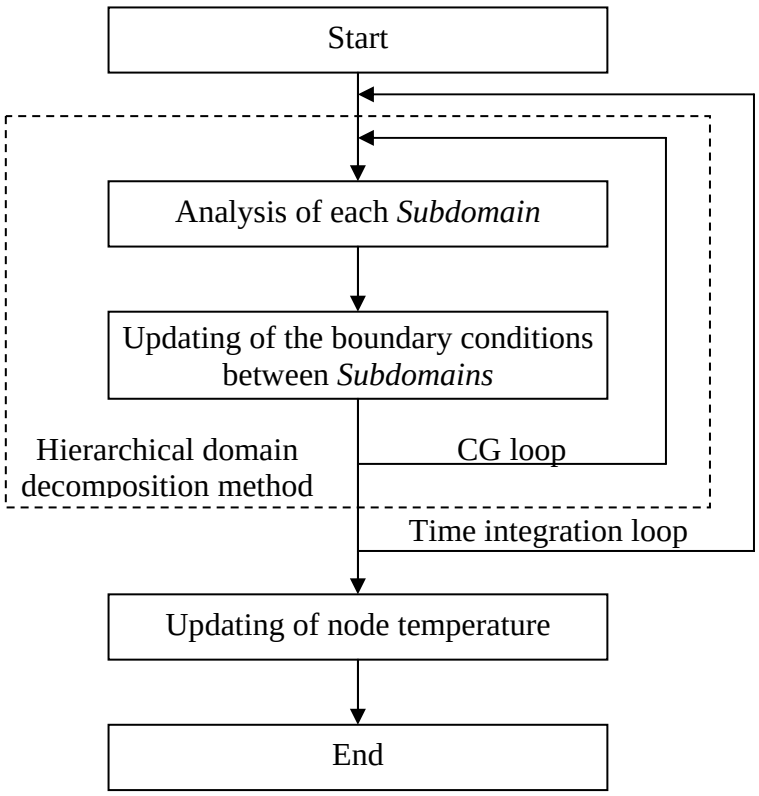


Figure 6. Algorithm of Transient Analysis

3.2. Input / Output Data

The files used by ADVENTURE_Thermal are shown in *Figure 7*. All files, except the job log file, have the binary ADVENTURE format. The data for one *Part* are stored in one file.

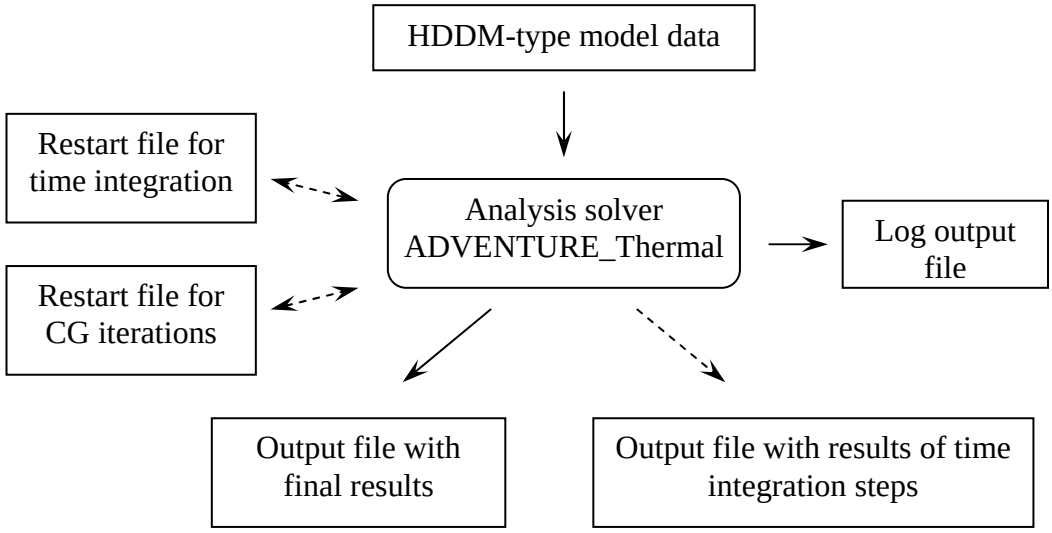


Figure 7. Input and Output Files

The ADVENTURE_Thermal module uses the input HDDM-type model data (hierarchically domain-decomposed data) files prepared by ADVENTURE_Metis. The calculated temperature is stored for all nodes in HDDM-type output data files. The output can be done for each step of time integration. Calculations can be terminated with saving of the data into temporary restart files and restarted using the restart data files. Two kinds of restart files can be used.

1). Restart file for CG loop.
Used for steady analyses.

2). Restart file for time integration loop.
Used for transient analyses.

If the user uses BDD solver then another restart file for coarse matrix (LU decomposition) can be used.

3.3. Standard of Temperature Units

The temperature data for ADVENTURE_Thermal must have the unit of degree *Celsius*. Other temperature units are not supported. The program converts all the temperature from *Celsius* to *Kelvin* automatically.

3.4. Boundary Conditions

The following boundary conditions can be set.

- Specified temperature (set for nodes)
- Specified heat flux (set for nodes)
- Specified heat convection (set for surface)
- Specified heat radiation (set for surface; only for transient analysis)

The boundary conditions for specified heat radiation can be used only for transient analyses.

***Only the unit of temperature is fixed as *Celsius* (automatically converted to *Kelvin* by the program) in the current version of AdvThermal. The user should define the other units (ex. convection coefficient, distance, time etc).

3.5. Material Properties

The following isotropic material properties can be specified.

- Thermal conductivity
- Specific heat (data necessary for transient analysis)
- Density (data necessary for transient analysis)
- Stefan-Boltzmann constant (data needed for the specified heat radiation boundary conditions)
- Calorific value

***Only the unit of temperature is fixed as *Celsius* (automatically converted to *Kelvin* by the program) in the current version of AdvThermal. The user should define the other units (ex. conductivity, distance, time etc).

3.6. Output Results

Only the nodal temperature and flux values can be saved to files. One file of ADVENTURE binary format contains information on one *Part*.

4. Program Compilation and Installation

4.1. Compile

To compile the ADVENUTRE_Thermal module, you need properly installed MPI environment and ADVENTURE_IO libraries on your computer.
The following procedure should be followed to compile the ADVENTURE_Thermal module:

1. ./configure
2. make

Both of the commands should be executed from the top directory of ADVENTURE_Thermal module. After execution of shell script **configure**, all necessary computing environment will be recorded into the **Makefile.**

The shell script **configure** uses the following options. The absolute path to the top directory should be mentioned.

--with-advio=*directory*
This option is used to define the top directory of ADVENTURE_IO. Default is "$HOME/ADVENTURE".

--with-mpicc=*command*
This option is used to define the C compiler for MPI. The default is **mpicc**.
Parallel versions of ADVENTURE_Thermal will not be compiled if the C compiler for MPI is not found.

--with-mpi-cflags=CFLAGS
The options for C compiler are specified by CFLAGS if the program is compiled for MPI environment. For example, the following statement can be used if it is necessary to specify the include files for MPI.
--with-mpi-cflags="-I/usr/local/include/mpi"
The options specified here by CFLAGS for MPI compiler can be used together with the options for the single version of the program (options for CC compiler)**.**

--with-mpi-libs =LIBS
This option is used to define the MPI links. For example, the following statement can be used to define the MPI libraries.
--with-mpi-libs="-L/usr/local/lib/mpi –lmpi"
The necessary options specified here for MPI link, can be used together with the necessary options for the single version of the program (options for CC compiler).

--enable-optimize
The optimization for compilation is performed. If any other options are required for optimization, the following option should be used.

--enable-optimize=CFLAGS
The optimization for compilation is performed using the options specified by CFLAGS.

--prefix=install_dir
This option is used to define the top directory specified by *install_dir* for program installation. Only the executable modules will be installed in the directory *install_dir*/**bin.** The default directory is /$HOME/ADVENTURE**.**

If the compilation using the supplied configure shell script is failed, the samples of **Makefile** prepared in each subdirectory should be used for compilation. **Makefile.sample** should be copied to **Makefile** in each directory contained **Makefile.sample**. The **Makefile.in.sample** should also be copied to **Makefile.in** in the top directory of ADVENTURE_Thermal module.

The following macros should be changed in the **Makefile.in** in accordance with the concrete computational environment.

ADVSYS_DIR	← Top directory of ADVENTURE system
ADVIO_CONFIG	← Full path to ADVENTURE_IO script *advsys-config*
MPI_CC	← *C* compiler for MPI
MPI_LINKER	← *C* linker for MPI
CC	← *C* compiler
LINKER	← *C* linker
CFLAGS	← Options for optimization

- After changing the Makefile.in, execute the command `make` in the top directory.

```
% make
```

The files in different directory can also be compiled separately by executing make command every time in each directory. In that case, the files located in the **libfem** should be compiled before the files located in the directory **solver**.

4.2. Installation of Executable Module

- Execute the command `make install`.

```
% make install
```

The default directory for installation is `$(HOME)/ADVENTURE/`. To change the directory for installation, execute the command

```
% make install prefix=<install_dir>
```

where the option <*install_dir*> should include a full path to the directory for installation.

The following files will be installed.

`bin/advthermal-s`	← Executable module
`bin/advthermal-p`	← Executable module
`bin/advthermal-h`	← Executable module
`bin/mkbc4th`	← Boundary condition setup tool
`bin/makefem_thermal`	← Tool for entire FEA model data
`bin/so2th`	← Data converter tool
`doc/AdvThermal/manual-jp.pdf`	← User's Manual in Japanese
`doc/AdvThermal/manual-eg.pdf`	← User's Manual in English
`doc/AdvThermal/README.eucJP`	← Brief information in Japanese
`doc/AdvThermal/README`	← Brief information in English
`doc/AdvThermal/copyright`	← Copyright agreement

5. Program Execution

The ADVENTURE_Thermal module can be executed in 3 modes. To execute ADVENTURE_Thermal with `mpirun`, use the following commands.

- Single mode

  `% advthermal-s [`*options*`]` *data_dir*

- Parallel mode with static job distribution using MPI

  `% mpirun [`*options for* `mpirun] advthermal-p [`*options*`]` *data_dir*

- Parallel mode with dynamic job distribution using MPI

  `% mpirun [`*options for* `mpirun] advthermal-h [`*options*`]` *data_dir*

The options [*options for* `mpirun`] are specified for the `mpirun`. The options [*options*] are specified for the ADVENTURE_Thermal executable (see *Section 5.2* of the current manual for details). The option *data_dir* should contain a name of the top directory with data files for analysis (input/output directory).

5.1. Names of Input / Output Files

The default names of input and output files are presented below. The files are located under the top directory defined by *data_dir*. Here, *P* indicates the *Part* number and *S* indicates the step number of the time integration loop.

- HDDM-type analysis model file:
  data_dir`/model/advhddm_in_P.adv`

- Analysis results (steady analysis):
  data_dir`/result/advhddm_out_P.adv`

- Analysis results (transient analysis):
  data_dir`/result/advhddm_out_S_P.adv`

- Restart file for CG loop:
  data_dir`/cg-res/advhddm_in_P.adv`

- Restart file for time integration loop:
  data_dir`/result(right?)/advhddm_out_S_P.adv`

5.2. Command Options

The following command options can be used.

5.2.1. Options for the Transient Analysis

- `-ns` — The option is used to execute the transient analysis. It can be used with the following options.
- `--cn` — The option specifies that the time integration will be done by the *Crank-Nicolson* method. The default is backward finite difference scheme.
- `--step` *n* — The option specifies the maximum number of iterations for the time integration loop. The default number is 10.
- `--out-interval` *n* — The option specifies that the output results of each *n* step will be printed. There is no default value; only the results of the last step are printed.
- `--dt` *x* — The option specifies the range *x* of time interval. The default value is 1.0.
- `--init` *x* — The option specifies the initial temperature *x* for all nodes. The default value is 0 ^{o}C.
- `--use-resin` *n* — The option specifies the time integration step *n* from which the analysis will be restarted.

5.2.2. Options Related to Elements

- `-tet10-integ5` — The option is used to set 5 integral points for quadratic tetrahedral elements. The default integration is done with 4 integral points.

5.2.3. Options for the Iteration Control

ADVENTURE_Thermal uses the CG method to solve the linear equations of stiffness matrixes. The following options can be used to control the iterations by CG method.

- `-cg-tol` *x* — The option specifies the tolerance for convergence of iterations. The iterations stop when the relative error (ratio of the current CG residual to the initial CG residual) becomes smaller than the tolerance *x*. The default value is 1.0×10^{-6}.

- `-cgloop-max` *n* — The option specifies the maximum number of CG iterations. The default value is 1000.

- `-use-cg-resin` — The option specifies from which step the CG restart file will be read to restart the analysis. This option can be used only for steady analyses. No restart file will be read by default.

- `-resout-cglast` — The option specifies that the CG restart file will be created at the last step of CG loop. The file will be created whether the iterations have been converged or the limit number of iterations has been exceeded without convergence. No restart file is created by default.

5.2.4. Options for the BDD solver

ADVENTURE_Thermal uses the BDD solver to solve the linear equations of stiffness matrix. The following options can be used to control the BDD solver

- `-solver bdd` This option specifies that the BDD solver will be used.
- `-solver bdd-diag` This option specifies that the BDD solver will be used with diagonal scaling in the Neumann-Neumann problem inside the BDD algorithm.
- `-solver bdd -iLU` This option specifies that the IBDD solver will be used. Only advthermal-p supports this solver.
- `-solver bdd-diag -iLU` This option specifies that the IBDD-DIAG solver will be used. Only advthermal-p supports this solver.
- -resout-bdd-cmat This option specifies that the coarse matrix after LU decomposition will be saved in a file for restart. No file is created by default.
- `-use-bdd-cmat` This option specifies that coarse-matrix will be read from the file to reuse.
- `-bdd-dir` *`dir`* This option specifies name *dir* of directory for coarse-matrix input/output data. The default name is **bdd.** This option is used after using the –resout-bdd-cmat option.
- `-bdd-cmat-file` *`file`* This option specifies the name of coarse-matrix input/output files to restart. The default is **advhbdd_cmat_*** where '* 'is the processor number. This option is used after using the -resout-bdd-cmat option.
- `-ginv-alpha` *x* This option specifies the value of factor for alpha-regularization. The default value is 10^{-3}. You can use this option only if you use BDD or BDD-DIAG solver.

5.2.5. Options for Output Filename Specification

Usually, the user should set only the name of the top directory for analysis data. However, the filenames, other than the default filenames, can be specified adding the following options to the command line. Here, *S* is used for the step number of time integration and *P* is the *Part*'s number.

- `-model-file` *file* — The option specifies the name of input data files with analysis model. The characters *_P*`.adv` will be added to the filename set by the option *file*. The default filename is `advhddm_in`.

- `-model-dir` *dir* — The option specifies the name *dir* of directory with input data. The default name is `model`.

- `-result-file` *file* — The option specifies the name of output results files. The characters *_P*`.adv` (for steady analysis) or *_S_P*`.adv` (for transient analysis) will be added to the filename set by *file*. The default filename is `advhddm_out`.

- `-result-dir` *dir* — The option specifies the name *dir* of directory with output results. The default name is `result`.

- `-ns-resin-file` *file* — The option specifies the filename of input restart files for time integration steps. The characters *_S_P*`.adv` will be added to the filename set by *file*. The default filename is `advhddm_out`.

- `-ns-resin-dir` *dir* — The option specifies the name *dir* of directory with restart files for time integration steps. The default name is `result`.

- `-cg-resin-file` *file* — The option specifies the filename of input restart files for CG steps. The characters *_P*`.adv` will be added to the filename set by *file*. The default filename is `advhddm_cgres`.

- `-cg-resin-dir` *dir* — The option specifies the name *dir* of directory with restart files for CG steps. The default name is `cg-res`.

- `-cg-resout-file` *file* — The option specifies the filename of output restart files for CG steps. The characters `_P.adv` will be added to the filename set by *file*. The default filename is `advhddm_cgres`.

- `-cg-resout-dir` *dir* — The option specifies the name *dir* of directory with restart files for CG steps. The default name is `cg-res`.

5.2.6. Other Options

- `-file-para` — The option sets the parallel data processing mode. An exclusive data control is used for default mode.
- `-memlimit` *n* — The option specifies the upper limit of memory *n* [in *Mbytes*], which can be used for one process. If this limit is exceeded, the process will be terminated. The default value is 256 [*Mbytes*].
- `-help` or `-h` — These options are used to display the help information.
- `-version` or `-v` — These options are used to display the version of the code.
- `-help-ns` — This option is used to display the help information on possible control options for the transient analysis.
- `-help-cg` — This option is used to display the help information on possible control options for CG iterations.
- `-help-bdd` — This option is used to display the help information on possible control options for the BDD solver.

Appendix

A. Supported Elements

ADVENTURE_Thermal supports only linear and quadratic tetrahedral elements. However, to set boundary conditions for heat convection and heat radiation, the stiffness matrixes should be created for the boundary surfaces of model. In the case of linear tetrahedral elements, the integration is performed for the linear triangular elements formed from the surface nodes, and in the case of quadratic tetrahedral elements, the integration is performed for the quadratic triangular elements formed from the surface nodes.

A.1. Linear Tetrahedral Element

(1). Nodes. The element contains 4 nodes with connectivity and numbering shown in *Figure 8*.

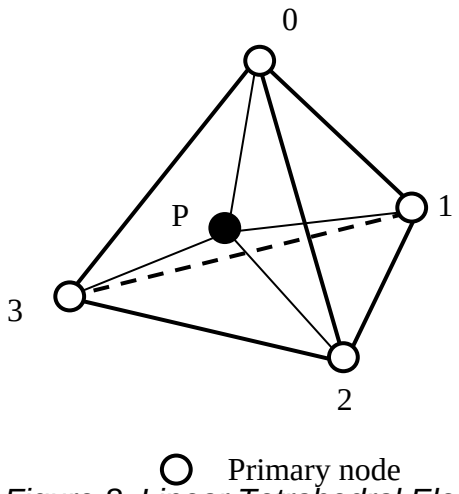


Figure 8. Linear Tetrahedral Element

(2). Integral points. The element has 1 integral point. The integral point *P* has the following volumetric coordinates (L_0, L_1, L_2, L_3).

L_0 = volume of tetrahedron *P123* / volume of tetrahedron *0123* (1)
L_1 = volume of tetrahedron *P023* / volume of tetrahedron *0123* (2)
L_2 = volume of tetrahedron *P013* / volume of tetrahedron *0123* (3)
L_3 = volume of tetrahedron *P012* / volume of tetrahedron *0123* (4)

Table 2. Integral Points of Linear Tetrahedral Element

Integral point number	L_0	L_1	L_2	L_3
0	1/4	1/4	1/4	1/4

A.2. Quadratic Tetrahedral Element

(1). Nodes. The element contains 10 nodes with connectivity and numbering shown in *Figure 9*.

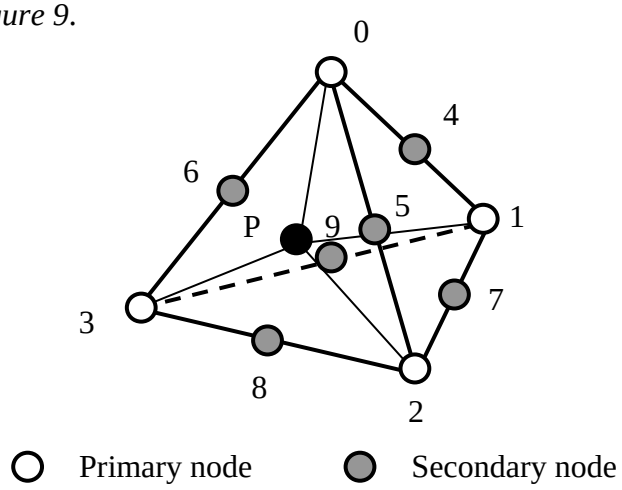


Figure 9. Quadratic Tetrahedral Element

(2). Integral points. The element has 4 integral points (default). It can be changed to 5 by command options. The integral point *P* has the following volumetric coordinates (L_0, L_1, L_2, L_3).

L_0 = volume of tetrahedron *P123* / volume of tetrahedron *0123* (5)
L_1 = volume of tetrahedron *P023* / volume of tetrahedron *0123* (6)
L_2 = volume of tetrahedron *P013* / volume of tetrahedron *0123* (7)
L_3 = volume of tetrahedron *P012* / volume of tetrahedron *0123* (8)

Table 3. Integral Points of Linear Tetrahedral Element (4 integral points)

Integral point number	L_0	L_1	L_2	L_3
0	β	α	β	β
1	β	β	α	β
2	β	β	β	α
3	α	β	β	β

α = 0:58541019662496845446
β = 0:13819660112501051518

Table 4. Integral Points of Linear Tetrahedral Element (5 integral points)

Integral point number	L_0	L_1	L_2	L_3
0	1/4	1/4	1/4	1/4
1	1/6	1/2	1/6	1/6
2	1/6	1/6	1/2	1/6
3	1/6	1/6	1/6	1/2
4	1/2	1/2	1/6	1/6

A.3. Linear Triangular Element

The linear triangular elements are used for integration when it is necessary to set the convection and radiation boundary conditions for linear tetrahedral elements.

(1). Nodes. The element contains 3 nodes with connectivity and numbering shown in *Figure10.*

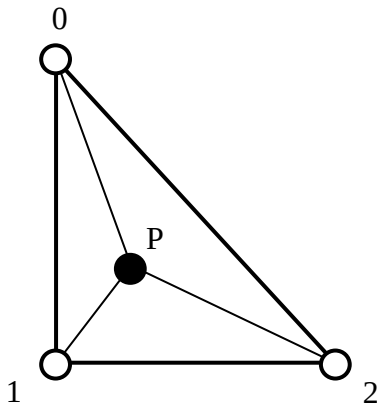


Figure 10. Linear Triangular Element

(2). Integral points. The element has 1 integral point. The integral point *P* has the following volumetric coordinates (L_0, L_1, L_2).

L_0 = area of triangular *P12* / area of triangular *012* (9)
L_1 = area of triangular *P02* / area of triangular *012* (10)
L_2 = area of triangular *P01* / area of triangular *012* (11)

Table 5. Integral Points of Linear Triangular Element

Integral point number	L_0	L_1	L_2
0	1/3	1/3	1/3

A.4. Quadratic Triangular Element

The quadratic triangular elements are used for integration when it is necessary to set the convection and radiation boundary conditions for quadratic tetrahedral elements.

(1). Nodes. The element contains 6 nodes with connectivity and numbering shown in *Figure 11*.

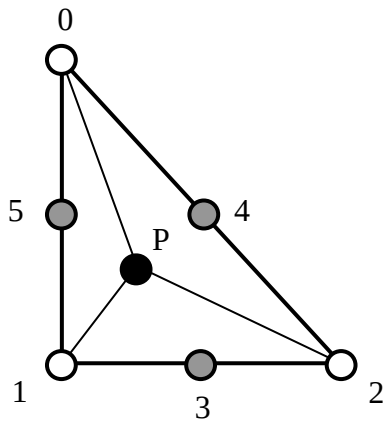


Figure 11. Quadratic Triangular Element

(2). Integral points. The element has 3 integral points. The integral point *P* has the following volumetric coordinates (L_0, L_1, L_2).

L_0 = area of triangular *P12* / area of triangular *012* (12)
L_1 = area of triangular *P02* / area of triangular *012* (13)
L_2 = area of triangular *P01* / area of triangular *012* (14)

Table 6. Integral Points of Quadratic Triangular Element

Integral point number	L_0	L_1	L_2
0	1/2	1/2	0
1	0	1/2	½
2	1/2	0	½

B. Setup of Boundary Conditions

The format of boundary condition data, which is used by ADVENTURE_Thermal will be presented below on examples. The **makefem_thermal** and **mkbc4th** is used to create the following boundary conditions data.

B.1. Boundary Conditions for Temperature

Example

```
----------------------------------------
[Properties]
1: content_type=FEGenericAttribute
2: num_items=81
3: fega_type=NodeVariable
4: label=Temperature
5: format=i4f8
6: index_byte=4
[Data]
0 0 1.000000e+02
1 0 1.000000e+02
3 0 1.000000e+02
58 0 1.000000e+02
59 0 1.000000e+02
60 0 1.000000e+02
.  .  .  .
.  .  .  .
.  .  .  .
```

The format of `[Data]` is (from left): the node number, the directional component, and the temperature. Since, the degree-of-freedom of nodes for heat conduction analysis is 1, setting of directional components, as it would be done for structure mechanics analysis, is unnecessary. All directional components are set to 0.

B.2. Boundary Conditions for Heat Flux

Example

```
----------------------------------------
[Properties]
1: content_type=FEGenericAttribute
2: num_items=81
3: fega_type=NodeVariable
4: label=HeatFlux
5: format=i4f8
6: index_byte=4
[Data]
0 0 0.000000e+00
1 0 3.333333e+01
3 0 0.000000e+00
58 0 6.666666e+01
59 0 6.666666e+01
60 0 6.666666e+01
.  .  .  .
.  .  .  .
.  .  .  .
```

The format of `[Data]` is (from left): the node number, the directional component, and the heat flux. The heat flux shown here was converted from the surface heat flux to the node-concentrated heat. If q is the heat flux per unit area S, the node-concentrated heat for quadratic tetrahedral element can be presented as

- P0
  $q_0 = 0$
- P1
  $q_1 = 0$
- P2
  $q_2 = 0$
- P3
  $q_3 = q \times S/3$
- P4
  $q_4 = q \times S/3$
- P5
  $q_5 = q \times S/3$

B.3. Boundary Conditions for Heat Convection

Example

```
-----------------------------------------
[Properties]
1: content_type=FEGenericAttribute
2: num_items=8
3: fega_type=ElementVariable
4: label=HeatConvection
5: format=i4f8f8
6: index_byte=4
[Data]
0 1 1.000000e+02 1.231002e+02
5 3 1.000000e+02 1.231002e+02
.  .  .  .
.  .  .  .
.  .  .  .
```

The format of `[Data]` is (from left): the element number, the surface number, the outer contact temperature, and the heat convection coefficient. The numbering of surfaces is done in a way that the surface numbers of each element been equal to the number of the node opposite to the surface. For example, the surface number 0 is opposite to the node number 0. **mkbc4th** can be used to create the above data format.

B.4. Boundary Conditions for Heat Radiation

Example

```
----------------------------------------
[Properties]
1: content_type=FEGenericAttribute
2: num_items=8
3: fega_type=ElementVariable
4: label=HeatRadiation
5: format=i4f8f8f8
6: index_byte=4
[Data]
0 1 1.000000e+02 1.000000e+00 1.000000e+00
5 3 1.000000e+02 1.000000e+00 1.000000e+00
.  .  .  .
.  .  .  .
.  .  .  .
```

The format of `[Data]` is (from left): the element number, the surface number, the temperature of emitter, the emissivity, and the geometrical viewfactor. **mkbc4th** can be used to create the above data format.

C. Tool Program

C.1. Converter `so2th` for Analysis Model File

To perform heat conduction analyses by ADVENTURE_Thermal, the input data should be converted into a special format. The data conversion program **`so2th`** is distributed along with the ADVENTURE_Thermal module. **`so2th`** converts the following boundary conditions data without changing the material properties data.

- Displacement boundary conditions are converted to temperature boundary conditions.

Note
`so2th` does not support the heat convection and heat radiation boundary conditions. For setting the flux boundary conditions, it is recommended to use **makefem_thermal** tool.

Program Execution

To execute **`so2th`**, use the command:

```
% so2th solid_file
```

where, *solid_file* is the entire-type FEA model data created by ADVENTURE_BCtool. The characters `_thermal.adv` will be added by `so2th` to the filename defined by *solid_file*.

To use the command `makefem` of ADVENTURE_BCtool, which creates the entire-type FEA model file for heat conductivity analysis, the file with material properties should be created in advance. Examples of material properties data are given below.

(1). An example of a one-material model

```
---------------------------------
HeatConductivity 200
Density 10.0
SpecificHeat 100.0
StefanBoltzmanConstant 5.667e-8
InternalHeatGeneration 0.0
```

From the top: the heat conductivity coefficient, the material density, the specific heat, the *Stefan-Boltzmann* constant, and the internal heat generation.

(2). An example of a multi-material model

```
---------------------------------
#materialInfo
materialN 2
propertyN 5
HeatConductivity 100
Density 5000
SpecificHeat 41.78
StefanBoltzmanConstant 5.667e-8
InternalHeatGeneration 0.0
HeatConductivity 50
Density 2500
SpecificHeat 20.0
StefanBoltzmanConstant 5.667e-8
InternalHeatGeneration 0.0
#volumeInfo
volumeN 2
1
0
```

C.2. makefem_thermal

The boundary conditions and material properties attached to mesh can be saved in an entire-type FEA model file of ADVENTURE binary format by using the **makefem_thermal** tool. This tool supports temperature and flux boundary conditions and uses the following input and output files.

Input:

Mesh data file (extension is msh)
Mesh surface data file (extension is fgr)
File with boundary conditions (extension is cnd)
Material properties data file (extension is dat)

Output:

Entire-type FEA model file (extension is adv)

The following argument should be specified with **makefem_thermal** in the command line.

% makefem_thermal mshFile fgrFile cndFile matFile advFile

mshFile : the name of the mesh data file
fgrFile : the name of the mesh surface data file
cndFile : the name of the boundary conditions data file
matFile: the name of the material properties data file
advFile: the name of entire-type FEA model file

File format of boundary conditions data file(cndFile)

```
gravity 0 0 0                  ← Dummy for thermal problem
boundary 2                     ← Number of boundary conditions
dispOnFaceGroup 0 0 0 10       ← Temperature on the surface group 0 is 10[°C]
dispOnVertex    123 0 20       ← Temperature of node 123 is 20 [°C]
loadOnFaceGroup 5 0 0 100      ← Flux on the surface group 5 is 100.
```

dispOnFaceGroup specifies the temperature on a surface group.
dispOnVertex specifies the temperature on a node.
loadOnFaceGroup specifies the flux on a surface group.

The above cndFile is created by **bcGUI** command of ADVENTURE_BCtool. After startup of **bcGUI,** a default window will appear on the screen. Then the boundary conditions are attached to a node or a surface group as follows.

1) Select the surface group (or node).
2) Select the menu **Add Load (**in case of flux boundary conditions) or **Add Displacement** (in case of temperature boundary conditions) from the main menu **BC.**
3) Check the [X] box and set the values of flux or temperature into the test box on the right.

** Do not use the [normal] box to add the flux or temperature boundary conditions.

C.3. mkbc4th

mkbc4th tool is used to make the ADVENTURE format binary file of convection(B.3) and radiation(B.4) boundary conditions. This binary file should be added to the entire-type FEA ADVENTURE format binary file by using **advcat** tool. This tool uses surface data file (fgrFile) which is the output of **msh2pch.**

Input :

Mesh surface data file (extension is fgr)
File with boundary conditions (extension is cnd)

Output :

ADVENTURE format binary file (extension is .adv)

This file contains only convection or radiation boundary conditions part.

Command

% mkbc4th fgrFile cndFile advFile

fgrFile : the name of the mesh surface data file
cndFile : the name of the boundary conditions data file
advFile: the name of boundary condition data file

cnd file for convection

```
boundary 1                      ← Number of boundary conditions
convOnFaceGroup 1 100 123.1     ← The fluid flows through surface 1 has outer
                                  contact temperature 100[°C] and heat
                                  transfer coefficient 123.1.
```

cnd file for radiation

```
boundary 1                          ← Number of boundary conditions
radiOnFaceGroup 0 100 1.00 1.00     ← The surface 0 has temperature of emitter
                                      100[°C], the emissivity 1.00,
                                      and the geometrical viewfactor 1.00
```

A cndFile comes from **bcGUI** of ADVENTURE_BCtool can be edited to make the above two files. Since the ADVENTURE binary format file from the execution of **mkbc4th** contains only the information of convection or radiation boundary conditions, so this file should be added to the entire-type FEA model by using the **advcat** command of ADVENTURE_IO .

C.4. Visualization of Temperature Distribution (advauto_thermalview)

advauto_thermalview is a tool distributed with ADVENTURE_Auto-0.1b can be used to visualized the analyzed results. A data processing by **hddmmrg** is necessary for visualization of the analyzed results obtained by the advauto_thermalview. The **hddmmrg** distributed with ADVENTURE_Solid is designed to merge the domain-decomposed ADVENTURE binary format files containing the analyzed results data to the text data file.

The following command is used to execute the **hddmmrg**:

```
% hddmmrg    Temperature    data_dir
```

Here, the data_dir is the top directory where the analyzed results of the domain-decomposed model are located. The output file is Temperature.dat.

And then the following command is used to execute the **advauto_thermalview**:

```
% advauto_thermalview mshFile fgrFile
```

The mshFile and fgrFile are the same that are used in C.2. This program should be executed on the directory containing the Temperature.dat file.

D. Examples of Analysis

Here simple examples of applications of the ADVENTURE_Thermal are described. In the interest of simplicity, the examples are limited to heat transfer problems on some test models and use of quadrilateral tetrahedral elements.

D.1. Examples With Temperature Boundary Conditions

D.1.1. Analysis Model

Consider the conduction heat transfer in a block of dimension 1 m by 1 m by 1 m and thermal conductivity $k=100[W/m\ K]$. The two surfaces Γ_1 and Γ_2 of the model are maintained at a temperature of $T_1 = 100(^oC)$ and $T_2 = 0(^oC)$ (see *Figure 12*). The rest of the surfaces are maintained with natural boundary conditions. The total number of elements and nodes are 6511 and 10187 respectively. We wish to determine the steady temperature distribution through the model using ADVENTURE_Thermal.

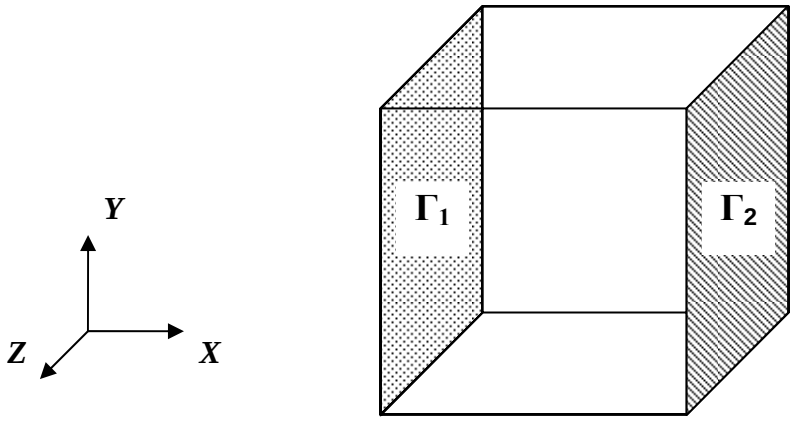


Figure 12. Analysis Model With Temperature Boundary Conditions

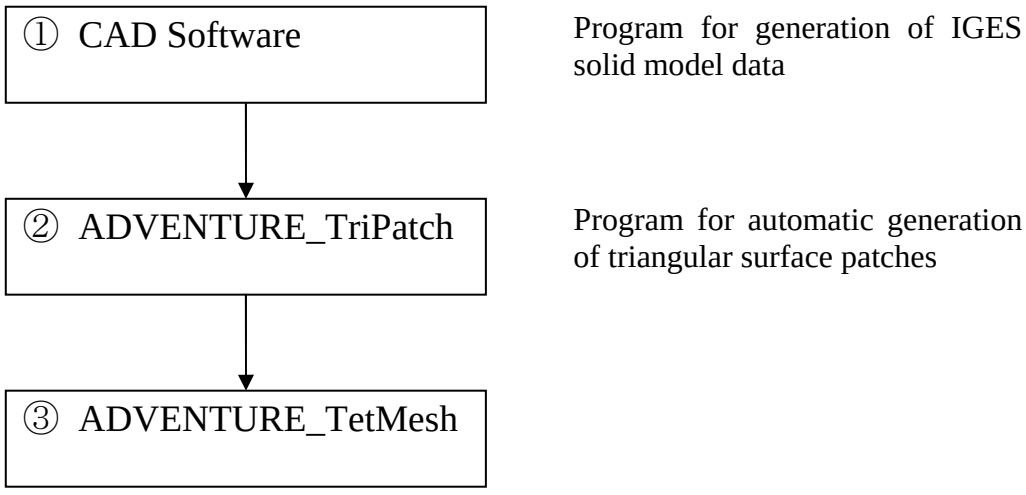
D.1.2. Example of Input Data Preparation

The following files should be prepared in the following sequence to proceed with analysis using ADVENTURE_Thermal:

1). Create the one-domain input data file.
2). Create the boundary conditions data file by using **bcGUI** tool (ADVENTURE_BCtool)
3). Create the material properties data file.
4). Create the entire-type FEA model data file using **makefem_thermal** command.
5). Create the HDDM-type model data file by using the ADVENTURE_Metis.

The above mention procedure will be explained below.

1). One-domain input data file (mesh file)
Preparations of one-domain input data file should follow the following sequence.



① Preparation of IGES solid model data file (*sample_1.igs*).

② Creation of node density file (*sample_1.ptn*) as follows. and execution of ADVENTURE_TriPatch.

sample_1.ptn

```
BaseDistance
1.0
```

Input:

Solid model data file (*sample_1.igs*)
Node density file (*sample_1.ptn*)

Command:

%**ADVENTURE_TriPatch** *sample_1 sample_1 --out_pch_form*

Output:

Created data of triangular surface patches are stored in the following files.

Surface patch data file (*sample_1.pch*).
Surface patch VRML file (*sample_1_c_mpc.wrl*).
Surface patch group data file (*sample_1.pcg*).

③ Execution of ADVENTURE_TestMesh

Execution of TestMesh_P

Input:

Surface patch data file (*sample_1.pch)*
Node density file (*sample_1.ptn*)

Command:

advtmesh9p *sample_1* **–d**

Output**:**

sample_1c.pch
sample_1c.ptn
sample_1_c.wrl

Execution of TestMesh_M

Input:

Surface patch data file (*sample_1c.pch)*
Node density file (*sample_1c.ptn*)

Command:
advtmesh9m *sample_1c* **–s** ***-p***

Output**:**
sample_1c.msh
sample_1c_n.wrl
sample_1c_e.wrl

2) Creation of boundary conditions file and execution of ADVENTURE_BCtool

Creation of one-domain input data file with suitable boundary conditions follows the following steps sequentially.

Step 1. Extraction of Mesh Surface

Input:
Mesh data file (*sample_1c.msh*)

Command:
%**msh2pch** *sample_1c.msh* 3

Output:
Mesh surface data file (*sample_1c_3.fgr*)
Extracted surface mesh data file (*sample_1c_3.pch*)
Patch group data file (*sample_1c_3.pcg*)
Global index file (*sample_1c_3.trn*)

Step 2. Setting Boundary Conditions

Input:
Extracted surface mesh data file (*sample_1c_3.pch*)
Patch group data file (*sample_1c_3.pcg*)

Command :
%bcGUI *sample_1c_3.pch sample_1c_3.pcg*

** Here you should input the boundary condition data file according to the ADVENTURE_BCtool manual.

Output :
Boundary conditions file (*sample_1c.cnd*)
sample_1c.cnd

```
gravity 0 0 0
boundary 2
dispOnFaceGroup 0 0 0 100
dispOnFaceGroup 5 0 0 0
```

3). Preparation of material properties data file

The material properties data file is created according to ***C.1.*** as follows.

material.dat

```
HeatConductivity 100
```

4). Creation of entire-type FEA model file

Input:
Mesh data file (*sample_1c.msh*)
Mesh surface data file (*sample_1c_3.fgr*)
Boundary conditions data file (*sample_1c.cnd*)
Material properties data file (*material.dat*)

Command:
%**makefem_thermal** *sample_1c.msh sample_1c_3.fgr sample_1c.cnd material.dat sample_1.adv*

Output:
Entire-type FEA model data file (*sample_1.adv*)

5). Create HDDM-type model file

The HDDM-type model file is prepared by ADVENTURE_Metis using the predetermined number of subdomains. The number of subdomains is determined following the method described in section 2.3.
If the number of elements per subdomain is n, number of total elements is $N_{element}$ and number of part is $N_{part,}$ then the number of subdomains is

$$N_{subdomain} = \frac{N_{element}}{n \times N_{part}} = \frac{6511}{2 \times 200} = 16.2775 \cong 16$$

Input :
Entire-type FEA model file for ADVENTURE_Thermal (*sample_1.adv*)

Command:
%**mpirun –np 2 adventure_metis –difn 1** *sample_1.adv sample_1 8*

Output:
HDDM-type model file

./sample_1/model/advhddm_in_0.adv
./sample_1/model/advhddm_in_1.adv
** File for each part is stored in *sample_1/model* directory.

D.1.3. Execution of ADVENTURE_Thermal

Input:
HDDM-type model file (*sample_1/model*)

Command:
%**mpirun –np 2 advthermal-p** *sample_1*

Output :
Result file of each part (*sample_1/result*)
./sample_1/result/advhddm_out_0.adv
./sample_1/result/advhddm_out_1.adv
** File for each part is stored in *sample/result* directory.

D.1.4. Calculated Results

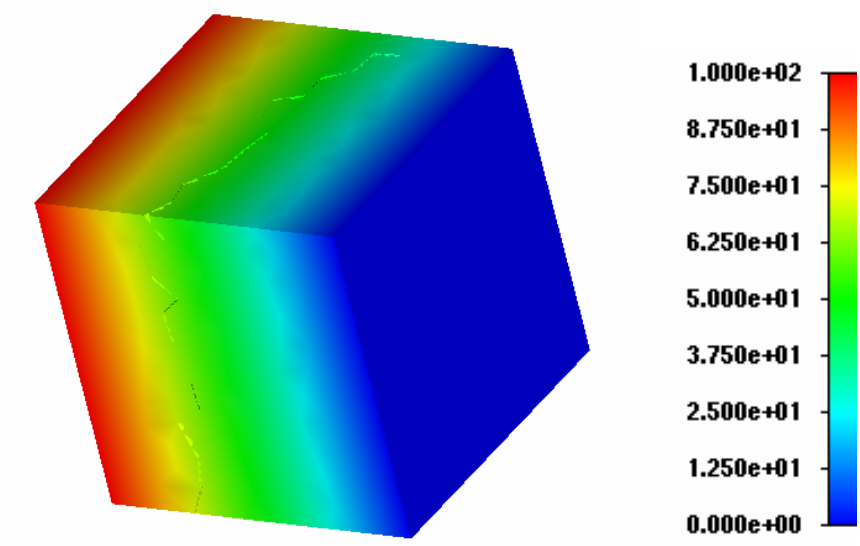


Figure 13. Temperature Distribution Visualized by ADVENTURE_Visual

D.2 Examples With Internal Heat Generations

D.2.1. Analysis Model

Consider the conduction heat transfer in a cross section of a sphere shown in *Figure 14*, thermal conductivity k = 8.854185Xe^{-12}[W/m K] and uniform internal heat generation of f = 1.0Xe^{-05}[W/m^3]. The outer surface Γ_1 is maintained at a temperature of T_1 = 1.0Xe^{04}[^{o}C]. The rest of the surfaces are maintained with natural boundary conditions. The total number of elements and nodes are 937 and 1378 respectively. We wish to determine the steady temperature distribution through the model using ADVENTURE_Thermal.

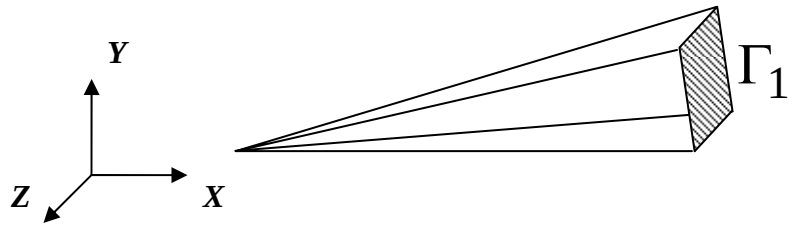


Figure 14. Analysis Model (cross section of a sphere)

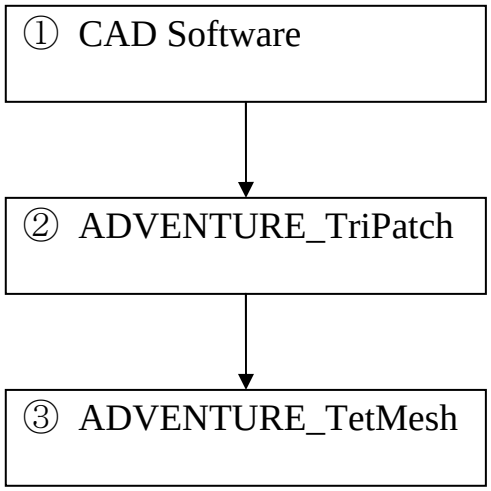
D.2.2. Input Data Preparation

The following files should be prepared in the following sequence to proceed with analysis using ADVENTURE_Thermal.

1). Create the one-domain input data file.
2). Create the boundary conditions data file by using **bcGUI** tool (ADVENTURE_BCtool)
3). Create the material properties data file.
4). Create the entire-type FEA model file using **makefem_thermal** command.
5). Create the HDDM-type model data file by using the ADVENTURE_Metis.

1). One-domain Input Data (mesh file)

Preparation of one-domain input data file should follow the following sequence.



Program for generation of IGES solid model data

Program for automatic generation of triangular surface patches

① Preparation of IGES solid model data file (*sample_2.igs*).
② Creation of node density file (*sample_2.ptn*) as follows. and execution of ADVENTURE_TriPatch.

Input:

IGES solid model data file (*sample_2.igs*).
Node density file (*sample_2.ptn*) as following.

sample_2.ptn

```
BaseDistance
0.1
```

Command:

ADVENTURE_TriPatch *sample_2 sample_2 --out_pch_form*

Output:

Created data of triangular surface patches are stored in the following files.
Surface patch data file (*sample_2.pch*).
Surface patch VRML file (*sample_2_c_mcp.wrl*).
Surface patch group data file (*sample_2.pcg*).

③ Execution of ADVENTURE_TetMesh

Execution of TetMesh_P

Input:

Surface patch data file (*sample_2.pch)*
Node density file (*sample_2.ptn*)

Command:

advtmesh9p *sample_ 2* **–d**

Output **:**
sample_2c.pch
sample_2c.ptn
sample_2c.wrl

Execution of TetMesh_M

Input :
Surface patch data file (*sample_2c.pch*
Node density file (*sample_2c.ptn*)

Command :
advtmesh9m *sample_2c* **–s** ***-p***

Output**:**
sample_2c.msh
sample_2c_n.wrl
sample_2c_e.wrl

2). Creation of boundary conditions file and execution of ADVENTURE_BCtool

Creation of one-domain input data file with suitable boundary conditions follows the following steps sequentially.

Step 1. Extraction of Mesh Surface

Input:
Mesh data file (*sample_2c.msh*)

Command:
%**msh2pch** *sample_2c.msh* 4

Output:
Mesh surface data file (*sample_2c_4.fgr*)
Extracted surface mesh data file (*sample_2c_4.pch*)
Patch group data file (*sample_2c_4.pcg*)
Global index file (*sample_2c_4.trn*)

Step 2. Setting Boundary Conditions

Input:
Extracted surface mesh data file (*sample_2c_4.pch*)
Patch group data file (*sample_2c_4.pcg*)

Command:
% bcGUI *sample_2c_4.pch sample_2c_4.pcg*

** Here you should input the boundary condition data file according to the ADVENTURE_BCtool manual.

Output:

Boundary conditions file (*sample_2c.cnd*)

sample_2c.cnd

```
gravity 0 0 0
boundary 1
dispOnFaceGroup 4 0 0 10000
```

3). Preparation of material properties data file

The material properties data file is created according to ***C.1.*** as follows.

material.dat

```
HeatConductivity 8.854185e-12
InternalHeatGeneration 1.000000e-05
```

The procedure 4) and 5) follows the same way as used in ***D.1***.

D.2.3. Example of Output Data (After solution by ADVENTURE_Thermal)

Input:

HDDM-type model file (*sample_2/model*)

Command:

%**mpirun –np 2 advthermal-p** *sample_2*

Output:

Result file of each part (*sample_2/result*)

./sample_2/resultl/advhddm_out_0.adv

./sample_2/result/advhddm_out_1.adv

** File for each part is stored in *sample_2/result* directory.

D.2.4. Calculated Results

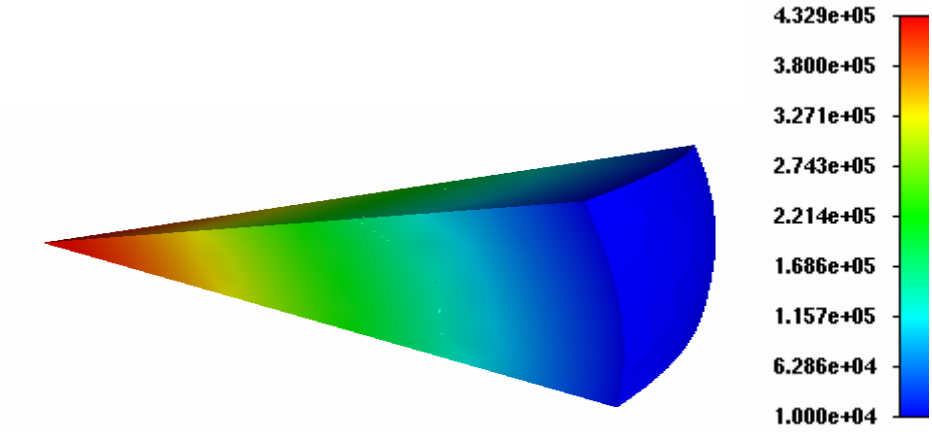


Figure 15. Temperature Distribution Visualized by ADVENTURE_Visual

D.3 Examples With Flux Boundary Conditions

D.3.1. Analysis Model

Consider the conduction heat transfer in a cross section of a cylinder shown in *Figure 16*, thermal conductivity $k = 50$ *[W/mm K]*, inner radius in 125 mm and outer radius 250 mm. The outer surface Γ_2 is maintained at a temperature of $T_1 = 10$ *[^oC]* and heat flows through the inner surface Γ_1 at the rate of 100 *[W/mm^2]*. The rest of the surfaces are maintained with natural boundary conditions. The total number of elements and nodes are 17,150 and 243,563 respectively. We wish to determine the steady temperature distribution through the model using ADVENTURE_Thermal.

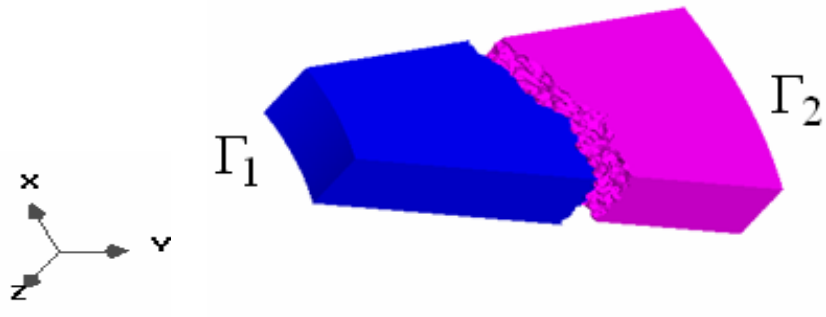


Figure 16. Analysis Model (Cross section of a cylinder)

D.3.2. Input Data Preparation

The following files should be prepared in the following sequence to precede with analysis using ADVENTURE_Thermal.

1). Create the one-domain input data file.
2). Create the boundary conditions data file by using **bcGUI** tool (ADVENTURE_BCtool)
3). Create the material properties data file.
4). Create the entire-type FEA model data file using **makefem_thermal** command.
5). Create the HDDM-type model data file by using the ADVENTURE_Metis.

1). Create the one-domain input data file.

Preparation of one-domain input data file follows the same procedure described in ***D.1*** and ***D.2.***

2). Create of boundary conditions data file

Creation of boundary conditions data file follows the following steps sequentially.

Step 1. Extraction of Mesh Surface

Input:
Mesh data file (*sample_3c.msh*)

Command:
%**msh2pch** *sample_3c.msh* 4
Output:
Mesh surface data file (*sample_3c_4.fgr*)
Extracted surface mesh data file (*sample_3c_4.pch*)
Patch group data file (*sample_3c_4.pcg*)
Global index file (*sample_3c_4.trn*)

Step 2. Setting Boundary Conditions

Input:
Extracted surface mesh data file (*sample_3c_4.pch*)
Patch group data file (*sample_3c_4.pcg*)

Command:
%bcGUI *sample_3c_4.pch sample_3c_4.pcg*

Output:
Boundary conditions file (*sample_3.cnd*)

sample_3.cnd

```
gravity 0 0 0
boundary 2
dispOnFaceGroup 4 0 0 10
loadOnFaceGroup 5 0 0 -100
```

3). Preparation of material properties data file

The material properties data file is created according to ***C.1.*** as follows.

material.dat

```
HeatConductivity 50
```

4). Creation of entire-type FEA model file

Input:
Mesh data file (*sample_3c.msh*)
Mesh surface data file (*sample_3c_4.fgr*)
Boundary conditions data file (*sample_3.cnd*)
Material properties data file (*material.dat*)

Command:
% **makefem_thermal** *sample_3c.msh sample_3c_4.fgr sample_3.cnd material.dat sample_3.adv*

Output:
Entire-type FEA model (*sample_3.adv*)

Procedure 5) follows the same way as in ***D.1*** and ***D.2.***

D.3.3. Example of Output Data (After solution by ADVENTURE_Thermal)

Input:
HDDM-type model file *sample_3/model*)

Command:
%**mpirun –np 2 advthermal-p** *sample_3*

Output:
Result file of each part (*sample_3/result*)
./sample_3/resultl/advhddm_out_0.adv
./sample_3/result/advhddm_out_1.adv
** File for each part is stored in *sample_3/result* directory.

D.3.4. Calculated Results

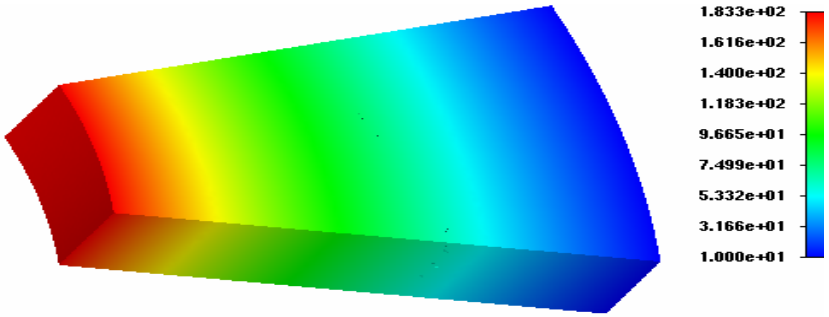


Figure 17. Temperature Distribution Visualized by ADVENTURE_Visual

D.4 Examples With Convection Boundary Conditions

D.4.1. Analysis Model

Consider the conduction heat transfer in a cross section of a cylinder shown in *Figure 18* , thermal conductivity, k=8.647500Xe^{-02}[W/mm K], inner radius is 125 mm and outer radius is 250 mm. A fluid with high temperature of 37.77 [^{o}C] flows through the inner surface Γ_1 and with a low temperature of -17.77 [^{o}C] flows through the outer surface Γ_2. The rest of the surfaces are maintained with natural boundary conditions. The total number of elements and nodes are 937 and 1378 respectively. We wish to determine the steady temperature distribution through the model using ADVENTURE_Thermal.

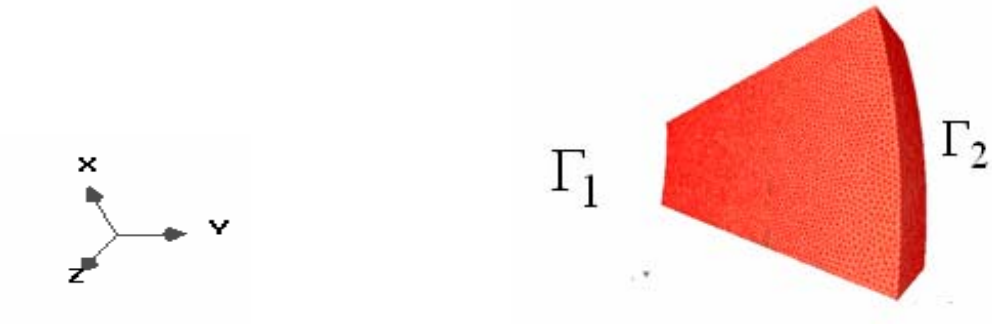


Figure 18. Analysis Model (Cross section of a cylinder)

D.4.2. Input Data Preparation

The following files should be prepared in the following sequence to proceed with analysis using ADVENTURE_Thermal.

1). Create the one-domain input data file.
2). Create the cnd file.
3). Create the material properties data file.
4). Create the entire-type FEA model data file using **makefem_thermal** command.
5). Create the boundary conditions data file using **bcGUI** tool (ADVENTURE_BCtool), edit the file suitable for **mkbc4th** and execution of **mkbc4th**.
6). Save the convection boundary conditions to entire-type FEA model using **advcat** command.
7). Create the HDDM-type model data file using the ADVENTURE_Metis.

1). One-domain Input Data (mesh file)
Preparation of one-domain input data file should follows the procedure described in **D.1** or **D.2** or **D.3.**

2). Creation of boundary conditions file and execution of ADVENTURE_BCTool

Creation of one-domain input data file with suitable boundary conditions follows the following steps sequentially.

Step 1. Extraction of Mesh Surface

Input:
Mesh data file (*sample_4c.msh*)

Command:
% **msh2pch** *sample_4c.msh* 4

Output:
Mesh surface data file (*sample_4c_4.fgr*)
Extracted surface mesh data file (*sample_4c_4.pch*)
Patch group data file (*sample_4c_4.pcg*)
Global index file (*sample_4c_4.trn*)

Step 2. Create the Boundary Conditions Data File

In this model temperature or flux boundary conditions are not used. So the cnd file for **makefem_thermal** tool is used as follows. Later we add convection boundary conditions with **mkbc4th** tool

sample_4.cnd

```
gravity 0 0 0
boundary 0
```

3). Prepare material properties data file

The material properties data file is created according to ***C.1.*** as follows.

material.dat

```
HeatConductivity 50
```

4). Create the entire-type FEA model file using **makefem_thermal** command

Input:
Mesh data file (*sample_4c.msh*)
Mesh surface data file (*sample_4c_4.fgr*)
Boundary conditions data file (*sample_4.cnd*)
Material properties data file (*material.dat*)

Command:

%**makefem_thermal** *sample_4c.msh sample_4c_4.fgr sample_4.cnd mateiral.dat sample_4e.adv*

Output :
Entire-type FEA model (*sample_4e.adv*)

5). Create the ADVENTURE binary format file for convection boundary conditions.

Input:
Mesh surface data file (*sample_4c_4.fgr*)
Boundary conditions data file (*convection.cnd*)

convection.cnd

```
boundary 2
convOnFaceGroup 5    37.77    2.837233e-02
convOnFaceGroup 4    -17.77    2.837233e-02
```

Command:
% **mkbc4th** *sample_4c_4.fgr convection.cnd convection.adv*

Output:
ADVENTURE binary format convection boundary conditions data file (*convection.adv*)

6). Save the convection boundary condition in entire-type FEA model data file

Input:
Entire-type FEA model data file (*sample_4e.adv*)
Convection boundary conditions data file (*convection.adv*)

Command:
% **advcat** *sample_4e.adv convection.adv sample_4.adv*

Output:
Entire-type FEA model data file with convection boundary conditions (*sample_4.adv*)

The procedure 7) follows the same way as used in ***D.1.*** and ***D.2***.

D.4.3. Example of Output Data (After solution by ADVENTURE_Thermal)

Input:
HDDM-type model file *sample_4/model*)

Command:
%**mpirun –np 2 advthermal-p** *sample_4*

Output:
Result file of each part (*sample_4/result*)
./sample_4/resultl/advhddm_out_0.adv
./sample_4/result/advhddm_out_1.adv
** File for each part is stored in *flux/result* directory.

D.4.4. Calculated Results

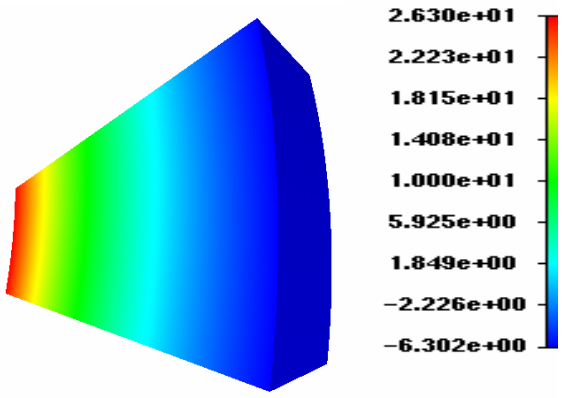


Figure 19. Temperature Distribution Visualized by ADVENTURE_Visual

D.5 Examples of Large Scale Analysis

A large scale HTTR (High Temperature Test Reactor) model (*Figure 20*) with about 2 millions degrees of freedom (dof) is analyzed by ADVENTURE_Thermal module. As the boundary conditions for this model, some high temperature is set on the lower plan and some low temperature is set on the upper plan. *Figure 21*. shows the temperature distribution after solution by ADVENTURE_Thermal.

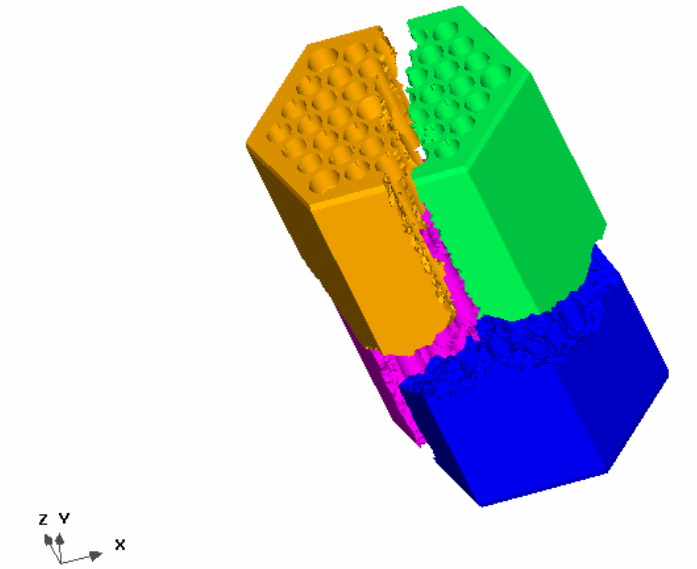


Figure 20. Domain Decomposition of HTTR Model

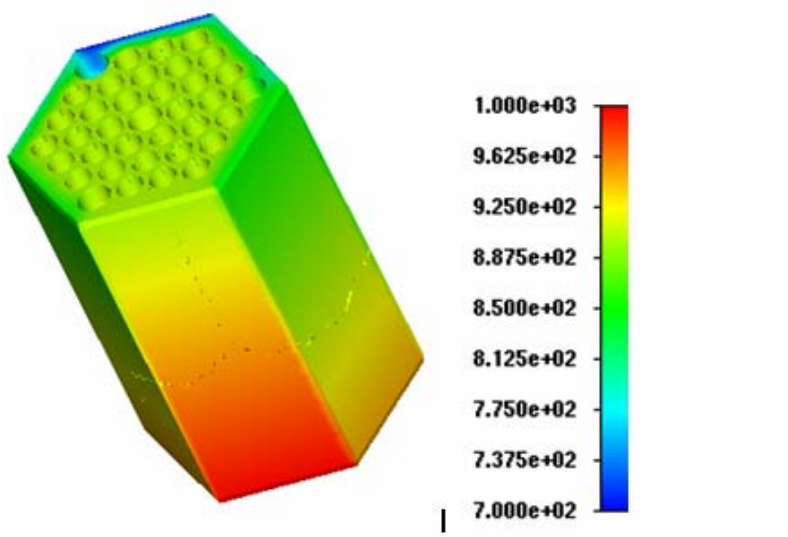


Figure 21. Temperature Distribution Visualized by ADVENTURE_Thermal

References

[1]. ADVENTURE Project: http://adventure.q.t.u-tokyo.ac.jp

[2]. 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), pp. 495-503.

[3]. G.Yagawa and R.Shioya: Massively Parallel Finite Element Analysis, Asakura-Shoten, (1998) (in Japanese).

[4]. T.Miyamura, H.Noguchi, R.Shioya, S.Yoshimaura and G.Yagawa: Massively Parallel Elastic-Plastic Finite Element Analysis Using the Hierarchical Domain Decomposition Method, *Transactions of Japan Society of Mechanical Engineers (JSME)*, 65-A, No.634(1999), pp. 1201-1208 (in Japanese).

[5]. R.Shioya, H.Kanayama, D.Tagami and E.Imamura: A Domain Decomposition Approach for Non-steady Heat Conductive Analysis, *Advances in Computational Engineering & Science*, 189.pdf, pp. 1-6, 2001.

[6]. MPI: http://www-unix.mcs.anl.gov/mpi/

[7]. MPICH: http://www-unix.mcs.anl.gov/mpi/mpich/

[8] Jan Mandel: Balancing Domain Decomposition, *Communications on Numerical Methods in Engineering*, 9(1993), 233-241

[9] R.Shioya, H. Kanayama, A.M.M.Mukaddes and M. Ogino: Heat Conductive Analysis with Balancing Domain Decomposition Method, *Journal of Theoretical and Applied Mechanics*, 52(2003), 43-53.

[10] M. Ogino, R. Shioya, H. Kanayama and A.M.M.Mukaddes: Incomplete Balancing Domain Decomposition for Large Scale Thermal-Solid Coupling Problems, *WCCM VI in Conjunction with APCOM*(2004).