Transactions of the Korean Nuclear Society Spring Meeting
Jeju, Korea, May 29-30, 2014

Enhancement of Pre- and Post-Processing Capability of the CUPID code

Jongtae Kim [a*], Ik Kyu Park [b], Han-Young Yoon [b]
[a]*Severe Accident & PHWR Safety Research Division, Korea Atomic energy Research Institute, Daejeon, Korea*
[a]*Thermal Hydraulics Safety Research Division, Korea Atomic energy Research Institute, Daejeon, Korea*
[*]*Corresponding author: ex-kjt@kaeri.re.kr*

1. Introduction

To simulate heat transfer and fluid flow in a field with a complicated geometry, an unstructured mesh is popularly used. Most commercial CFD (Computational Fluid Dynamics) solvers [1, 2, 3] are based on an unstructured mesh technology. An advantage of using unstructured meshes for a field simulation is reduced man-hours by automatic mesh generation compared to a traditional structured mesh generation, which requires a huge amount of man-hours to discretize a complex geometry. Initially, unstructured meshes that can be generated automatically are limited to regular cell elements such as tetrahedron, pyramid, prism, or hexahedron. However, it is commonly understood that polyhedral cells, which have more than six faces, are more efficient with a reduced degree of freedom (normally, this means the number of cells). In addition, hexahedral cells are more appropriate for resolving the boundary layers developed near walls.

The multi-dimensional multi-phase flow solver, CUPID [4], has been developed in the context of an unstructured mesh finite volume method (FVM). Its numerical formulation and programming structure is independent of the number of faces surrounding the computational cells. Thus, it can be easily extended into polyhedral unstructured meshes. In this study, new tools for enhancing the pre- and post-processing capabilities of CUPID are proposed. They are based on an open-source CFD tool box OpenFOAM [5]. A goal of this study is an extension of the applicability of the CUPID code by improving the mesh and solution treatment of the code.

2. Methods and Results

2.1 Analysis method and Conditions for Simulation

An unstructured mesh can be constructed with regular or irregular cells depending on the method of mesh generation. Fig. 1 shows irregular cells that belong to an unstructured mesh. Because the number of faces bounding the cells is arbitrary, they are called polyhedral cells. A polyhedral cell without a hanging point, which is a mesh point not fully connected by edges, can be made during the pre-processing of an automatic unstructured mesh generation. A good aspect of a polyhedral mesh is that it can reduce the number of computational cells in the mesh. Hexahedral cells are very attractive for a simulation of a thermal or viscous boundary layer near walls because they can efficiently resolve a gradient of a flow property normal to the walls. However, it is very difficult to generate a hexahedral mesh for a complicated geometry. Though a multi-block mesh technique can be applied for a complicated geometry, a topological problem that can occur in an interface of mesh blocks hinders its popular use in CFD. In other words, the multi-block mesh can be a very efficient way to generate a mesh with a good quality if the topological problem is overcome by introducing hanging points. The right figure in Fig. 1 is an example of a mesh with hanging points.

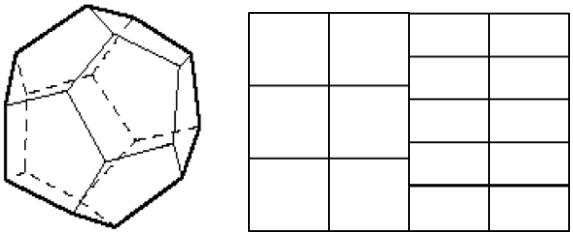

Fig. 1. Irregular cells in unstructured mesh, left: polyhedral cell, right: cells with hanging points.

The CUPID code is based on an unstructured mesh FVM. CUPID is formulated without regard to cell shapes or numbers of cell faces. Basically, CUPID can accept a polyhedral mesh with hanging points. Currently, makeGeo, a pre-processing tool for CUPID, is used to extract the connectivity and geometric data of an unstructured mesh. However, it is limited to regular cells such as a tetrahedron, pyramid, prism, and hexahedron.

In this study, new pre- and post-processing tools are developed to enhance the capability of the CUPID solver. Extracting the connectivity and geometric data needed by an unstructured mesh solver such as CUPID from a polyhedral mesh with or without hanging points requires a complicated data structure and searching algorithm. For easy development of the pre-processing tool, the open-source CFD tool box OpenFOAM is used. Fig. 2 shows a flow diagram for generating a CUPID mesh data (somaGrid.in). “foamMeshToCupid” is the pre-processing tool for CUPID. It converts an OpenFOAM mesh into a CUPID mesh. The OpenFOAM toolbox already has many utilities to convert meshes from many commercial mesh generators. Using the foamMeshToCupid tool with OpenFOAM utilities, many types of meshes from commercial mesh generators can be imported to CUPID. In addition,

open-source mesh generation tools included in the OpenFOAM toolbox can be used for CUPID.

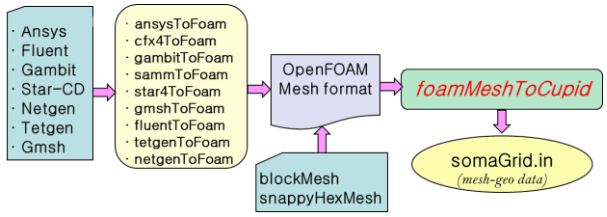

Fig. 2. A procedure of a pre-processing for CUPID

In parallel to the pre-processing tool, a post-processing tool was developed using the OpenFOAM library. Fig. 3 shows a flow diagram illustrating the procedure of the post-processing. After the solution data are obtained by running the CUPID solver, the "cupidToFoam" tool, which is a data conversion utility, converts the solution data into the OpenFOAM format. The converted solution is visualized using an open-source data visualizer, Paraview. Other post-processing tools such as sampling or averaging included in the OpenFOAM toolbox are also available on the converted solution data.

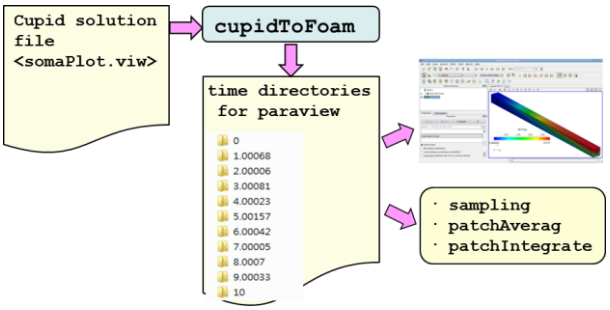

Fig. 3. A procedure of a post-processing for CUPID

2.2 Results

The CUPID system integrated with the newly developed pre- and post-processing tools is tested for some cases, i.e., dam-breaking, boiling of water in a circular cylinder, and flashing of water in a square channel.

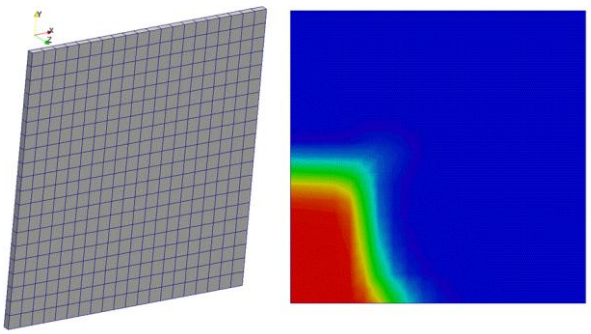

Fig. 4. Mass flow rates for hydrogen and steam-water released into S/G compartment for test case 1.

The first case used to verify the CUPID system is the well-known dam-breaking problem. A mesh for this test case is generated using a block mesh generator (blockMesh) included in the OpenFOAM tool box. Because blockMesh supports only a 3-dimensional geometry, a mesh is generated with a single cell in the z-coordinate direction. The two patches normal to the z-coordinate are a dummy, and the cell faces on the patches are skipped in flux integrations on boundary faces in the CUPID code. Fig. 4 shows a generated mesh imported to CUPID through the foamMeshToCupid tool, and a snapshot of the water-air interface, which is plotted through the cupidToFoam utility.

To test a polyhedral mesh, a 3-dimensional dam-break problem with a horizontal cylinder is chosen. The mesh shown in Fig. 5 is generated using the OpenFOAM utilities. The procedure from the mesh generation to visualization is as follows:

snappyHexMesh → polyDualMesh → foamMeshToCupid → Cupid → cupidToFoam

The commands above are sequentially executed by a computer script file. As shown in the figure, the cell shapes are arbitrary. The solution obtained from the polyhedral mesh is shown in the right figure of Fig. 5. It is seen that the free surface is interacting with the cylinder.

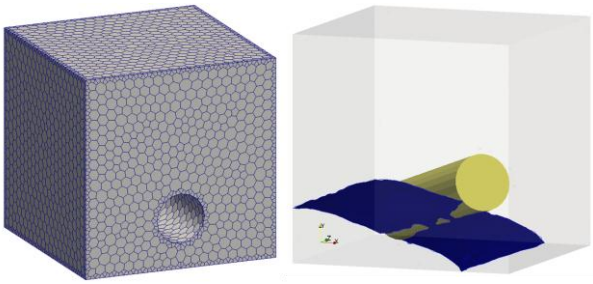

Fig. 5. 3-dimensional dam-break problem with a horizontal cylinder, left: polyhedral mesh, right: snap shot of a free surface

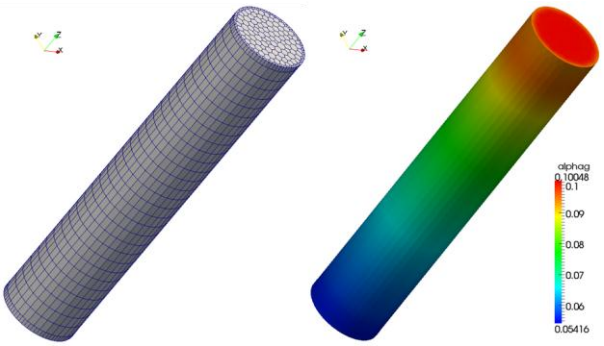

Fig. 6. Boiling water flow in a circular pipe, left: the polyhedral mesh, right: void distribution in the pipe

For a pipe flow with a volumetric heat generation, a mesh is generated by stacking a 2-dimesional mesh. In

this case, the stacked 3-dimensional mesh is converted into a polyhedral mesh through the following procedure:

get 2D mesh for a circle ➔ 2dTo3D ➔ polyDualMesh ➔ foamMeshToCupid ➔ Cupid ➔ cupidToFoam

which can also be seamlessly executed by a script file.

As described above, hanging points in a mesh are very attractive features for a high-quality mesh. They can be generated during a mesh refinement or merging meshes. Fig. 7 shows an example of mesh adaptation near walls. A mesh is generated and refined near walls for an analysis of a flashing water flow in a square channel. The gravitational force is applied normal to the flow direction. In the right figure of Fig. 7, it can be seen that the vapor concentration is shifted to the upper part of the channel because of gravity.

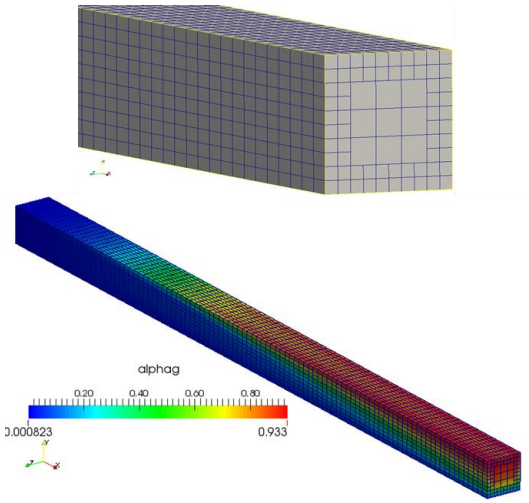

Fig. 7. Flashing water flow in a square channel, top: the mesh refined near channel walls, bottom: void distribution in the channel.

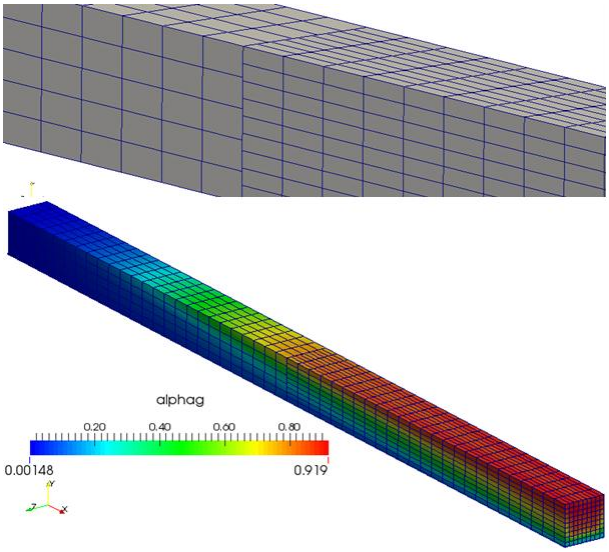

Fig. 8. Flashing water flow in a square channel, top: the mesh obtained by merging two meshes blocks, bottom: void distribution in the channel.

A mesh for the flashing water flow is generated by merging two mesh blocks, as shown in Fig. 8, which is a test case for the capability of a multi-block mesh generation with hanging points. It was found that a solution is obtained by the CUPID code without any problem. It is thought that the pre-processing tool for the polyhedral mesh with hanging points is working as expected.

3. Summary

A mesh generation for an interesting flow field is the first step of a numerical simulation of the field. It still requires a large amount of user effort. In this study, new pre- and post-processing tools are developed to enhance the capability of the CUPID solver. Many utilities such as mesh conversion, refinement, and merging the mesh blocks included in the OpenFOAM tool box are available for the pre-processing of a simulation by CUPID.

REFERENCES

[1] ANSYS FLUENT Theory Guide, ANSYS, Inc., 2011
[2] ANSYS CFX-Solver Theory Guide, ANSYS, Inc., 2009
[3] Star-CCD+ http://www.cd-adapco.com
[4] CUPID 1.7 Code Manual Volume 1: Mathematical Models and Solution Methods, KAERI, 2013
[5] OpenFOAM, http://www.openfoam.org