

NUMERICAL SIMULATIONS OF AIR FLOW AND HEAT TRANSFER VIA PDES

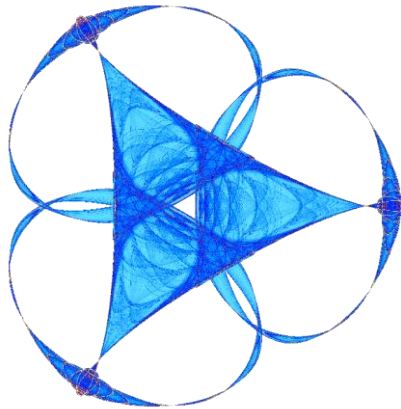
By

**Fei Cao, Luz Angelica Caudillo-Mata, Natalia Iwanski, Yong Li, Kamran Sadiq,
and Arturo Vargas**

Mentor: Vanessa López

IMA Preprint Series #2434

(August 2014)



INSTITUTE FOR MATHEMATICS AND ITS APPLICATIONS
UNIVERSITY OF MINNESOTA
400 Lind Hall
207 Church Street S.E.
Minneapolis, Minnesota 55455-0436
Phone: 612-624-6066 Fax: 612-626-7370
URL: <http://www.ima.umn.edu>

IMA Mathematical Modeling in Industry XVII Workshop:
Numerical Simulations of Air Flow and Heat Transfer via PDEs

Fei Cao, Luz Angelica Caudillo-Mata, Natalia Iwanski,
Yong Li, Kamran Sadiq, Arturo Vargas
Mentor: Vanessa López

August 7-16, 2013

Contents

1	Introduction	3
2	Mathematical Model	4
3	Modeling Approach	6
3.1	Define Model - BVP	6
3.2	Generating Mesh	7
3.3	System of Algebraic Equations	7
3.4	Numerical Solution/Visualizing results	7
4	Domain and Mesh	7
4.1	Constructive Solid Geometry	8
4.2	Algorithms	9
5	Numerical Solver	9
6	Visualizing Results	10
7	Case Studies and Data Visualization	10
7.1	Velocity	10
7.2	Temperature Distribution	10
7.3	Varying Temperature Neumann BC	11
8	Summary	12
9	Future Research	12
10	Acknowledgements	12
A	Algorithm 1	13
B	Algorithm 2	13
	References	15

1 Introduction

Partial differential equations (PDEs) describe a wide range of phenomena, for example fluid flow, heat transfer, and sound propagation, among others. As such, models comprised of sets of PDEs, coupled with suitable boundary and initial conditions, are frequently used to perform computer simulations in an effort to understand the behavior of solutions, as well as being essential components in various types of problems, like those dealing with control and optimal design. Thus, the numerical solution of such models, which is key in performing the computer simulations, is of fundamental importance.

In this workshop we considered the use of a simplified PDE model for simulating air flow and heat transfer, with the aim of gaining experience with the process of performing numerical simulations and the use of open source software to aid with the tasks of mesh generation, solution of the equations, and post-processing of the numerical results. In particular, the Netgen [1], OpenFOAM [2], and ParaView [3] software were used.¹ The PDE model studied in the workshop came from an application related to energy management in data centers [7]. In this report we discuss only aspects directly relevant to the workshop goals, though, but the interested reader is referred to [7] for a detailed description of the application and to [8] for a thorough treatment on the subject of fluid mechanics.

The physical environment which was used to define the domain for the PDE model, i.e., that of a data center, is depicted in Figure 1. Servers are placed in racks on a raised floor (which is referred to as the *plenum*). The side of a server that is required to be kept cool is referred to as its *inlet* side, while the side where (hot) air is blown out of the server is referred to as a server outlet. Cooled air is blown by the air conditioning units (ACUs) with large scale fans into the plenum thereby pressurizing it. This cooled air is then provided into the room through perforated tiles, which are placed in front of the server inlets. There are also air *intakes*, which in the layout from Figure 1 are located on top of the air conditioning units; these are meant to extract (hot) air out of the room. Thus, equipment like ACUs and servers, and other objects in the room, such as perforated tiles and air intakes, define sources or sinks of air and/or heat in the room, and therefore for the PDE model. How these sources and sinks are represented in the PDE model will be discussed in Section 3.1.

¹There are various other open source software packages for solving PDEs and/or for visualization, e.g., NGSolve [4], FEniCS [5], and Mayavi [6], which the reader may also find useful.

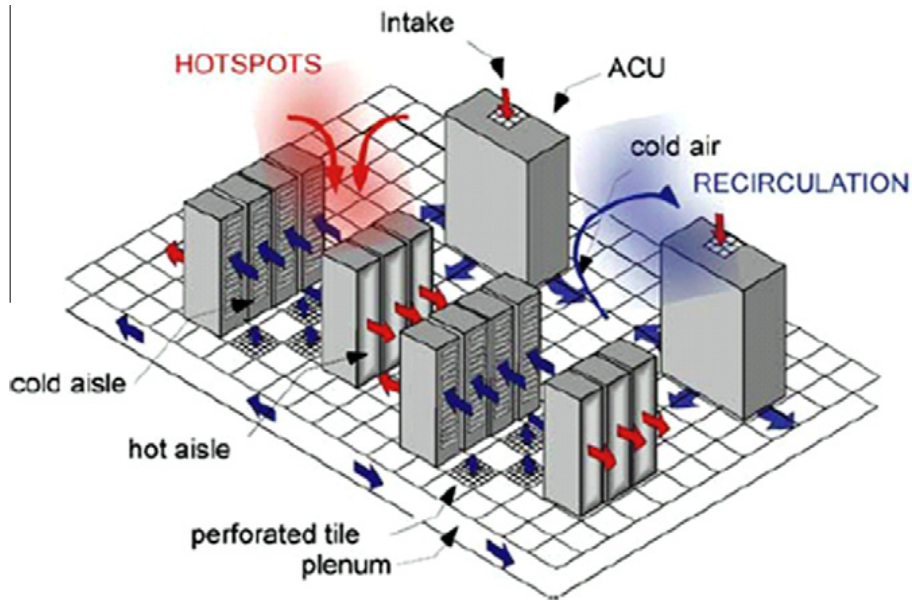


Figure 1: Typical data center layout. (Source of figure: [7])

2 Mathematical Model

Numerical simulations for temperature distribution are often modeled based on Navier-Stokes Equations (NSE) for air flow coupled with an equation for heat transfer. While this approach is highly accurate and has been used successfully, heavy computational effort is required due to the nonlinearity of NSE. Thus, it is often not suitable for real-time simulations.

Alternatively, in this report we consider a simplified model (linear and hence, faster to simulate) but still based on physical principles. More precisely, we employ potential flow theory [8]. Though a potential flow model cannot capture small-scale details, it may however capture global behavior. Since it is the global behavior of the system we are interested, it is reasonable to use a potential flow model.

To model air flow, we will assume flow is incompressible (constant air density ρ) and irrotational (Curl of the air velocity field is zero). We also neglect effects of air viscosity and turbulence.

Let v denote the air velocity field, incompressibility condition implies $\nabla \cdot v = 0$, where $\nabla = \langle \partial/\partial x, \partial/\partial y, \partial/\partial z \rangle$ denotes the gradient operator. The condition for irrotational flow ($\nabla \times v = 0$) is satisfied using the assumption that the air velocity field v is given by the gradient of a scalar function φ , called the potential function. That is

$$v = \nabla\varphi, \tag{1}$$

Substituting the expression 1 for velocity into the mass continuity equation

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho v) = 0, \quad (2)$$

and using the assumption that the air density ρ is constant yields the mass transport equation $\nabla \cdot (\rho \nabla \varphi) = 0$, or, after factoring our ρ , we get

$$\nabla^2 \varphi = \frac{\partial^2 \varphi}{\partial x^2} + \frac{\partial^2 \varphi}{\partial y^2} + \frac{\partial^2 \varphi}{\partial z^2} = 0, \quad (3)$$

which is the Laplace equation.

As discussed in section 1, equipment like ACUs, servers, perforated tiles and air intakes, define sources or sinks of air and/or heat in the room. For all of the aforementioned types of surfaces, sources and sinks of air located on boundaries of the solution domain are represented with the use of a Neumann boundary condition,

$$n \cdot \nabla \varphi = v_N, \quad (4)$$

where n denotes the unit outward normal vector at a point on the boundary and v_N is a function giving the value of the air velocity in the direction normal to the boundary at the point. For other boundary surfaces, like the walls, a zero-flow boundary condition will apply (i.e., $v_N = 0$ in Eq. 4).

The solution of the boundary value problem 3, 4 is defined only up to a constant, so the model must be further coupled with a Dirichlet boundary condition

$$\varphi = \varphi_D, \quad (5)$$

where φ_D denotes the prescribed value of the potential function at given points on the boundary. It is sufficient to select one boundary point to impose the condition 5 when performing the numerical simulations and can set $\varphi_D = 0$ at such point.

The equation 3, coupled with the boundary conditions 4, 5 are referred to as the data center Laplacian model.

For a steady-state temperature distribution, the heat transfer equation is being represented in the model via the convection–diffusion equation

$$\rho c_p v \cdot \nabla T - \nabla \cdot (k \nabla T) = 0, \quad (6)$$

where T denotes temperature, ρ , c_p and k denote, respectively, density, specific heat and thermal conductivity of air.

In the data center each server inlet and outlet side contribute to the heat transfer process in the room and is thus represented via the boundary conditions in the model and the use of sensor measurements to prescribe the associated boundary values.

To complete the mathematical description of the model, Eq. 6 must be coupled with appropriate boundary conditions prescribing temperature and/or heat flow at the domain boundaries (server inlet/outlet side, ACU intake, and perforated tile). Specifically, a Dirichlet boundary condition

$$T = T_D, \quad (7)$$

where T_D represents the value of T at given boundary points.

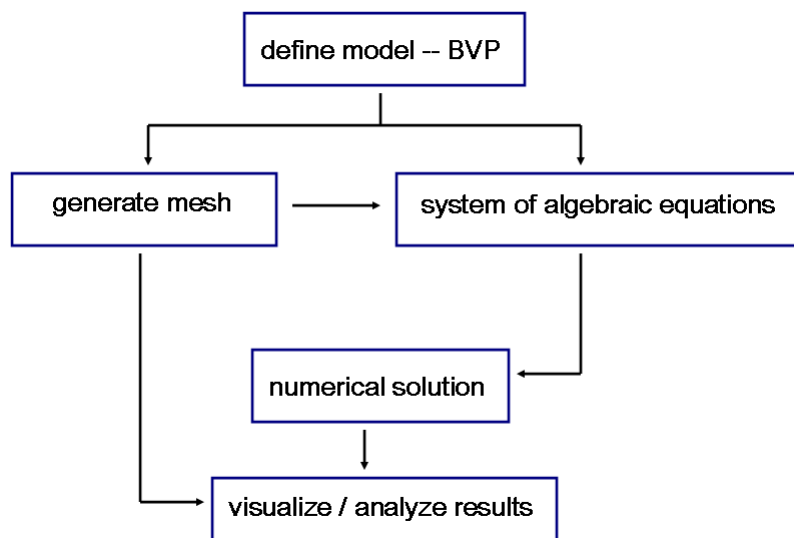
The flow of heat in the direction normal to the boundary (the inlet and outlet sides of servers) is being specified via the Neumann boundary condition

$$-n \cdot (k\nabla T) = q. \quad (8)$$

The equation 6, coupled with the boundary conditions 7 , 8 are referred as the data center temperature distribution model.

3 Modeling Approach

Finding the mathematical equation to model the physical behavior is just a part of the mathematical process. Below we outline the approach we employed and follow up with more in depth discussion on certain parts of the modeling process.



Outline of the employed modeling approach

3.1 Define Model - BVP

As described, any PDE requires boundary values and initial conditions in order to be well posed. In the context of a data center, the room and equipment will define our

boundary and various equipment will act as our sources and sinks for the model. For our study we will confine ourselves to a steady equation.

3.2 Generating Mesh

For our particular model, we will be employing a Finite Volume scheme. Finite volume is one the most popular numerical schemes in computational fluid dynamics due to the fact that the solution conserves mass quantities. In order to use the finite volume scheme, one must create a mesh. Mesh generation is a study in its own context and various groups have spent time developing software to simplify mesh generation, eg. Gmsh, NetGen. Prior to generating a mesh a domain must be established. Similar to mesh generation there are various tools available to create the geometry, such as constructive solid geometry or boundary representation.

3.3 System of Algebraic Equations

Once we establish a mesh we will be left with numerically solving a discretized PDE. There are various methods to numerically solve a PDE, e.g. finite difference, finite element, finite volume, etc... regardless of method we will end up with solving a system of algebraic equations. Depending on the structure of the system, certain solvers will be more suited for the system.

3.4 Numerical Solution/Visualizing results

Solving our PDE will result in large arrays of values, much like each component of the modeling process, data visualization is in its own right its own field and many have spent time creating tools to simplify the process.

4 Domain and Mesh

Specific to our model there were various challenges that we had to overcome. One of our goals was to be able to solve our PDE in a variety of domains. We strived for a minimum amount of information for a room layout to describe the domain. In our model we restrict to ourselves to cuboid shaped rooms with a constant difference between the ceiling and the floor. To hasten the process we chose to use open source software to mesh our domain.

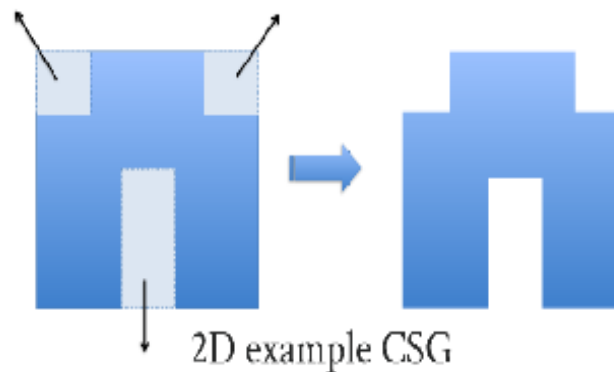
By looking at a room's floor plans we were able to devise algorithms that utilize the two dimensional cartesian coordinates for the room's corner points. In addition we confined

ourselves to domains that were simply connected and assumed our points were given in a counter clockwise manner.

4.1 Constructive Solid Geometry

There are various methods to define the boundary of a domain but due to the assumption that our domain would be cuboid shaped we used Constructive Solid Geometry (CSG) to define the boundary of our domain. CSG allows us to define intricate domains by using simple shapes (eg. cuboids, spheres) and the logical operators 'and', 'not' and 'or'.

As an example we can consider the following figure.



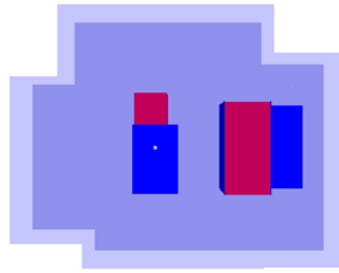
An example of Constructive Solid Geometry. We define the outbox and not the two corner boxes and the center cut.

With the aid of NetGen [1], an open source mesh generator, we were able to convert a series of points to a mesh for our domain.

points:

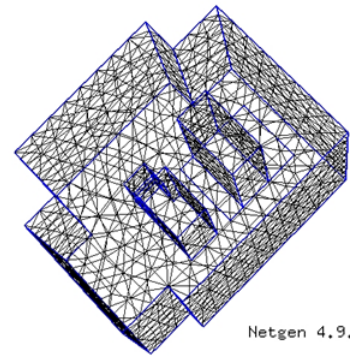
```
12.0 0.0
12.0 9.0
9.0 9.0
9.0 11.0
0.0 11.0
0.0 8.0
-2.0 8.0
-2.0 1.0
1.0 1.0
1.0 0.0
z min: 0.0
z max: 9.0
...
```

CSG
→



Netgen 4.9.13

mesh generation
algorithm
↘



Netgen 4.9.13

4.2 Algorithms

The first algorithm that we devised is able to handle single cuts, while the second algorithm is able to handle nested cuts. Both algorithms can be found in the appendix of this report. As examples of such cuts we provide the following images.



Figure 2: Simple Cuts



Figure 3: Nested Cuts

5 Numerical Solver

As with mesh generation we use the aid of open source software. OpenFOAM [2] is a popular Computational Fluid Dynamics toolbox written in C++ and our choice for solving our PDE. By using Matlab we were able to write a script that would allow us

to convert a mesh from Netgen into OpenFOAM format; and use the built in solver to solve our PDE.

6 Visualizing Results

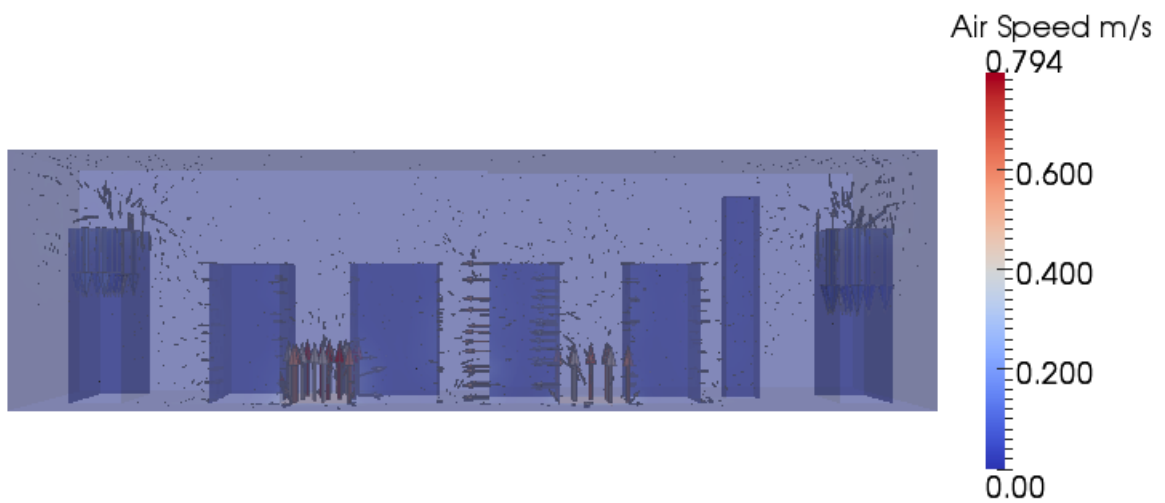
As important as the other processes is being able to interpret data. Once OpenFoam numerically solves our PDE we are able able to utilize ParaView [3] to visualize our results. Examples of the data visualization are provided in the following section.

7 Case Studies and Data Visualization

We present several case studies and examples of data visualization.

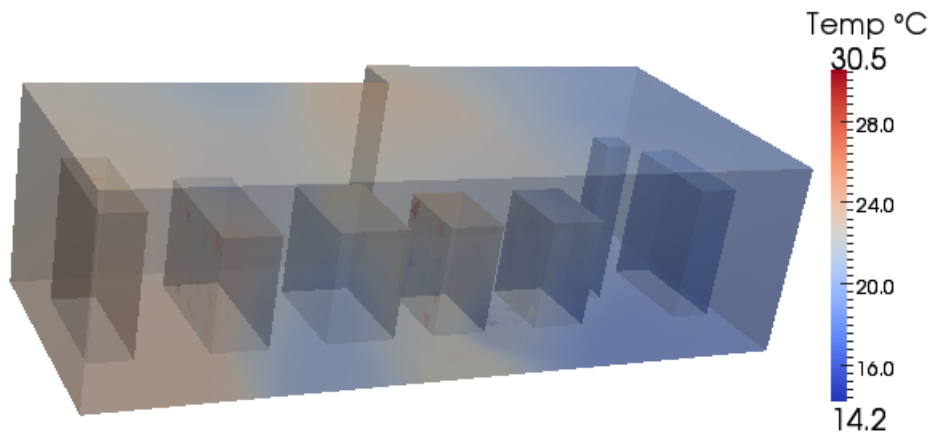
7.1 Velocity

Our couple of visualizations is a demonstration of ParaView's visualization capabilities. The figure below demonstrates the air velocity of the domain's sources and sinks.



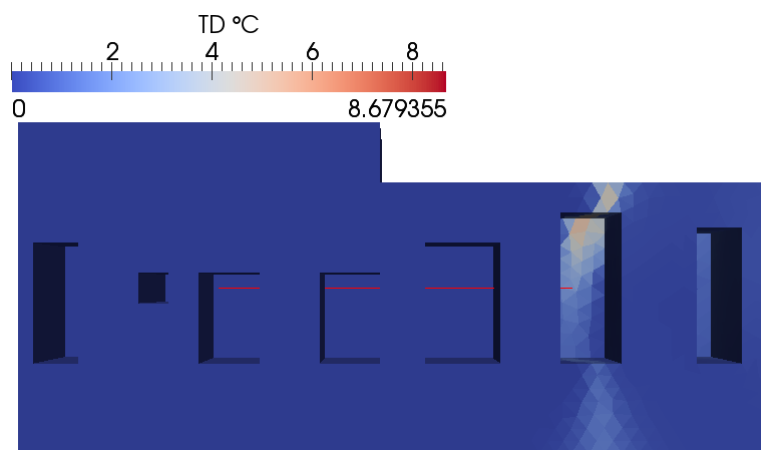
7.2 Temperature Distribution

Our second visualization demonstrates the temperature distribution at steady state.



7.3 Varying Temperature Neumann BC

The following figures are a result of a case study. In this case study we examined the effects of varying temperature boundary conditions. The figure below demonstrates the temperature difference.



The figure below on the left contains zero valued Neumann boundary value conditions on a particular set of servers (the hollowed out box on the right). The bottom right figure non-zero Neumann boundary values. Such numerical experiments allow us to develop insight on the behavior of the PDE.

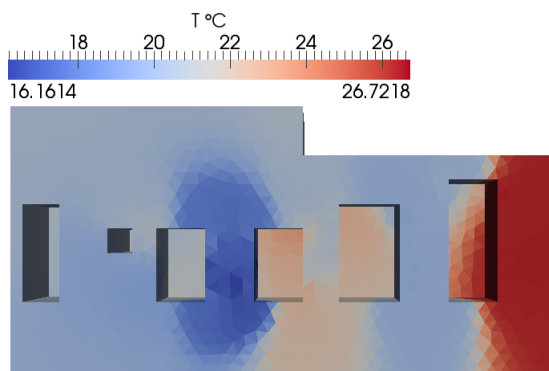


Figure 4: Zero Neumann boundary condition

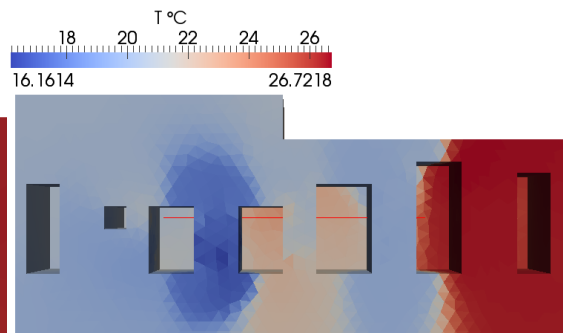


Figure 5: non-zero Neumann boundary condition

8 Summary

In summary we provided a walkthrough for modeling heat and air flow PDE's. We utilized various open source software in order to hasten the process. We provided a few examples of the visualization and demonstrated a particular case study. The bulk of the workshop was spent in getting to the point in which we could visualize our results. As a next step we would like to work on developing different case studies.

9 Future Research

In regards to the PDE model we employed, there are various things we would like to explore. In particular there was a limited amount of data available to prescribe boundary conditions. We would like to explore modeling with more such data. We also did not consider different mesh sizes. To test the robustness of our simulations we would like to vary the mesh size and compare the results. We would also like to introduce a transient model that incorporates time-dependent data for the boundary conditions.

10 Acknowledgements

We would like to thank our mentor, Dr. Vanessa Lopez, for taking the time to introduce us to her work. She provided the guidance and mentorship that were essential in this project. We would also like to thank the IMA and PIMS for the opportunity and funding to participate in the workshop. Richard Braun for his insights in the design of the case studies. And a great thanks to the various people who wrote on our behalf when we were applying to the program.

A Algorithm 1

Algorithm 1 Pseudocode:

```
1: function IDENTIFYCUTS( $P$ ) ▷  $P$ : Set of Domain Coordinates
2:   1. Find outer domain
3:   2. Classify points  $P_i \in P$  into two sets:
4:       a) Interior ( $\Omega$ )
5:       b) Boundary ( $\partial\Omega$ )
6:   3. Find cut locations:
7:   for each interior point,  $P_i$  do
8:     Look for  $P_{i-1}$  and  $P_{i+1}$ 
9:     if  $P_{i-1} \in \Omega$  then
10:       $\Omega \leftarrow \Omega \setminus \{P_{i-1}\}$ 
11:     end if
12:     if  $P_{i+1} \in \Omega$  then
13:       $\Omega \leftarrow \Omega \setminus \{P_{i+1}\}$ 
14:     end if
15:     Save cut information:
16:     Find dimensions of rectangular cut from  $P_{i-1}$  and  $P_{i+1}$ 
17:   end for
18:   4. Post-process information in required output format
19:       a) Number of cuts
20:       b) Position of cuts
21:       c) Dimension of cuts
22: end function
```

B Algorithm 2

For this algorithm the pseudo boundary refers to the rectangle that would be if no cuts were made.

Algorithm 2 Pseudocode:

```
1: function IDENTIFYCUTS( $P$ ) ▷  $P$ : Set of Domain Coordinates
2:   1. Find pseudo boundary (PB)
3:   2. Find boxes to remove out:
4:   for each  $P_i \in PB$  do
5:     if  $P_{i+1} \in PB$  then
6:        $P_i \leftarrow P_{i+1}$ , stop until the index has run a full loop
7:     else
8:       if  $P_{i+1} \notin PB$  then
9:         find  $P_j$  that first returns to PB since  $P_i$ 
10:        if  $P_i \& P_j$  on the same edge then
11:          Divide all edges from  $P_i$  to  $P_j$  into vertical(V) and horizontal(H)
12:          two groups
13:          for each  $u \in V$  do
14:            Do a rigid move for  $u$  along the exterior direction until it touches
15:             $v \in V$ , remove the box with  $|u|$  as the length and the distance between  $u$  and  $v$  as
16:            the width
17:          end for
18:          for each  $u \in H$  do
19:            Do a rigid move for  $u$  along the exterior direction until it touches
20:             $v \in H$ , remove the box with  $|u|$  as the length and the distance between  $u$  and  $v$  as
21:            the width
22:          end for
23:          else
24:            if  $P_i \& P_j$  not on the same edge then
25:              find the most outer corner point  $P$  on the PB between  $P_i, P_j$ .
26:              say  $\tilde{P}_j$  (where  $\tilde{P}_j$  is a point in the pseudo boundary). Then repeat the same process
27:              as in the previous if sentence.
28:            end if
29:          end if
30:        end if
31:      end if
32:    end for
33:  end function
```

References

- [1] sourceforge.net/projects/netgen-mesher.
- [2] openfoam.com.
- [3] paraview.org.
- [4] sourceforge.net/projects/ngsolve.
- [5] fenicsproject.org.
- [6] code.enthought.com/projects/mayavi.
- [7] V. López and H. F. Hamann. Heat transfer modeling in data centers. *Int. J. Heat Mass Transfer*, 54, 2011.
- [8] L. D. Landau and E. M. Lifshitz. *Fluid Mechanics*, volume 6 of *Course of Theoretical Physics*. Pergamon Press, 1959.