Scheduling Divisible Workloads from Multiple Sources in Regular and Torus Mesh

Junwei Zhang

Stony Brook, New York

Abstract

This report is about multiple sources[1] workloads scheduling[2] in regular mesh and torus mesh.

Keywords: Divisible Load Theory, Processor Equivalence, Voronoi Diagram, Optimal Mass Transport, Monte Carlo Method, Manhattan Distance

1. Introduction

2. Processor Equivalence

In this section, we will discuss the power equivalence problem.
I will present the closed-form equal formula about the load from three different kind of injection position,load from corner,boundary and inner grid position in regular mesh.
In addition, considering the unit core with or without the front-end processor,the data transportation schema can be considered as two situation with front-end and without front-end transportation.

For a homogeneous regular mesh, which can be collapsed into an equivalent node, the notation is presented as follows.

- α_0: The load fraction assigned to the root processor.
- α_i: The load fraction assigned to the ith processor.
- ω_i: The inverse computing speed on the ith processor.
- ω_{eq}: The inverse computing speed on an equivalent node collapsed from a regular mesh.
- z_i: The inverse link speed on the ith link.
- T_{cp}: Computing intensity constant.The entire load can be transmitted in $z_i T_{cp}$ on the ith processor.
- T_{cm}: Communication intensity constant.The entire load can be transmitted in $z_i T_{cm}$ seconds over the ith link.
- $T_{f,m}$: The finish time of the whole regular network.Here $T_{f,m}$ is equal to $\omega_{eq} T_{cp}$.
- $T_{f,0}$: The finish time for the entire divisible load solved on the root processor.Here $T_{f,0}$ is equal to $1 \times \omega_0 T_{cp}$, that is $\omega_0 T_{cp}$.

2.1. With Front-end Schema

I investigate the simple case first and then deduce the more general closed form formula for the homogeneous regular mesh situation.

2.1.1. Load From Corner

- 2*2 regular mesh Fig.1
- 2*3 regular mesh Fig.2
- 2*n regular mesh,for example $n = 10$ Fig.3
- 3*n regular mesh,for example $n = 8$ Fig.4
- m*n regular mesh,for example $m = 5, n = 5$ Fig.5

According to Fig.1,data injection position comes from corner.
There are four unit cores to handle the whole workload in the regular mesh.

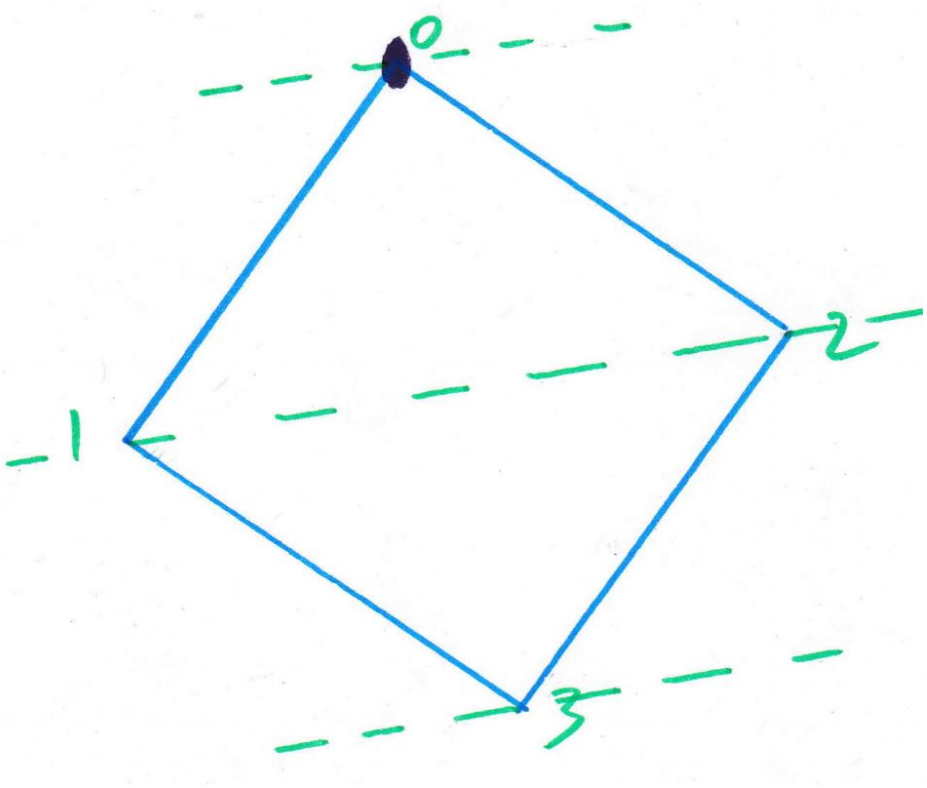

Figure 1: 2*2 regular mesh

$$\alpha_0 \omega T_{cp} = T_{f,m}$$
$$\alpha_1 \omega T_{cp} = T_{f,m}$$
$$\alpha_2 \omega T_{cp} = T_{f,m}$$
$$\alpha_1 z T_{cm} + \alpha_3 \omega T_{cp} = T_{f,m}$$

$$\sigma = \frac{zT_{cm}}{\omega T_{cp}}$$

$$\begin{bmatrix} 1 & 2 & 1 \\ 1 & -1 & 0 \\ 0 & \sigma - 1 & 1 \end{bmatrix} \times \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \tag{1}$$

According to Fig.2, the data load injection comes from corner and there are six cores to handle the whole workload.

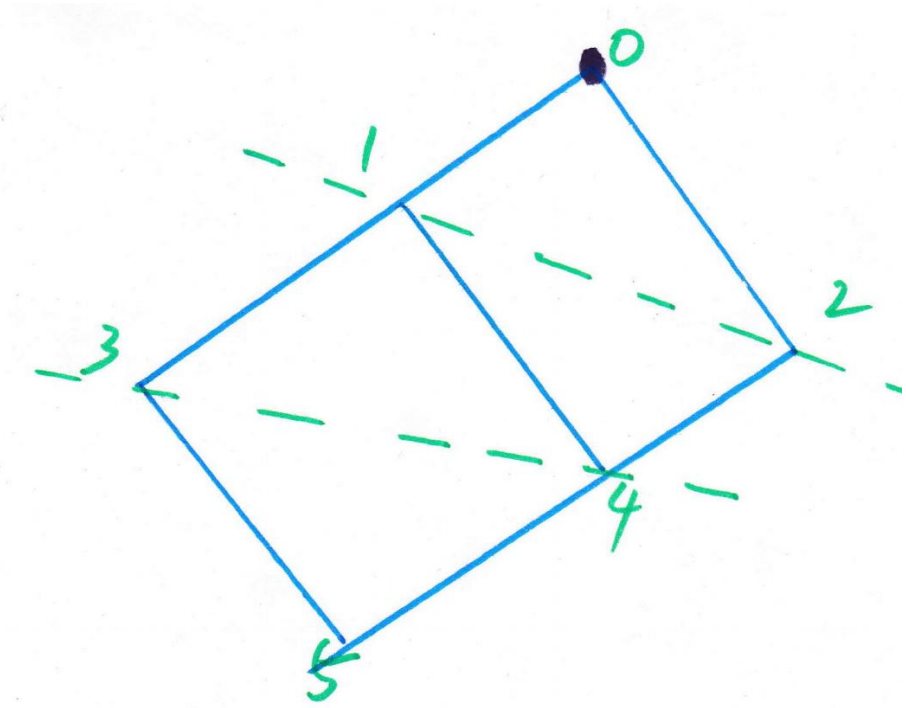

Figure 2: 2*3 regular mesh

$$\alpha_0 \omega T_{cp} = T_{f,m}$$
$$\alpha_1 \omega T_{cp} = T_{f,m}$$
$$\alpha_2 \omega T_{cp} = T_{f,m}$$
$$\alpha_1 z T_{cm} + \alpha_3 \omega T_{cp} = T_{f,m}$$
$$\alpha_2 z T_{cm} + \alpha_4 \omega T_{cp} = T_{f,m}$$
$$(\alpha_1 + \alpha_3) z T_{cm} + \alpha_5 \omega T_{cp} = T_{f,m}$$

$$\sigma = \frac{zT_{cm}}{\omega T_{cp}}$$

$$\begin{bmatrix} 1 & 2 & 2 & 1 \\ 1 & -1 & 0 & 0 \\ 0 & \sigma-1 & 1 & 0 \\ 0 & \sigma-1 & \sigma & 1 \end{bmatrix} \times \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_3 \\ \alpha_5 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{2}$$

Considering the $2 * n$ situation, the formula is:

$$\begin{gathered} \alpha_0 \omega T_{cp} = T_{f,m} \\ \alpha_1 \omega T_{cp} = T_{f,m} \\ \alpha_2 \omega T_{cp} = T_{f,m} \\ \alpha_1 z T_{cm} + \alpha_3 \omega T_{cp} = T_{f,m} \\ \alpha_2 z T_{cm} + \alpha_4 \omega T_{cp} = T_{f,m} \\ (\alpha_1 + \alpha_3) z T_{cm} + \alpha_5 \omega T_{cp} = T_{f,m} \\ \vdots \\ (\alpha_1 + \alpha_3 + \cdots + \alpha_{2\times n-1}) z T_{cm} + \alpha_{2\times n+1} \omega T_{cp} = T_{f,m} \\ \sigma = \frac{z T_{cm}}{\omega T_{cp}} \end{gathered}$$

For example, the $n = 10$ Fig. 3, the formula as follows:

$$\begin{bmatrix} 1 & 2 & 2 & \cdots & 2 & 2 & 1 \\ 1 & -1 & 0 & \cdots & 0 & 0 & 0 \\ 0 & \sigma-1 & 1 & \cdots & 0 & 0 & 0 \\ 0 & \sigma-1 & \sigma & 1 & 0 & \cdots & 0 \\ 0 & \sigma-1 & \sigma & \sigma & 1 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & \\ 0 & \sigma-1 & \sigma & \cdots & \sigma & \sigma & 1 \end{bmatrix} \times \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_3 \\ \alpha_5 \\ \vdots \\ \alpha_{2\times n-1} \\ \alpha_{2\times n+1} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{3}$$

So the Speedup is

$$\frac{1}{\alpha_0}$$

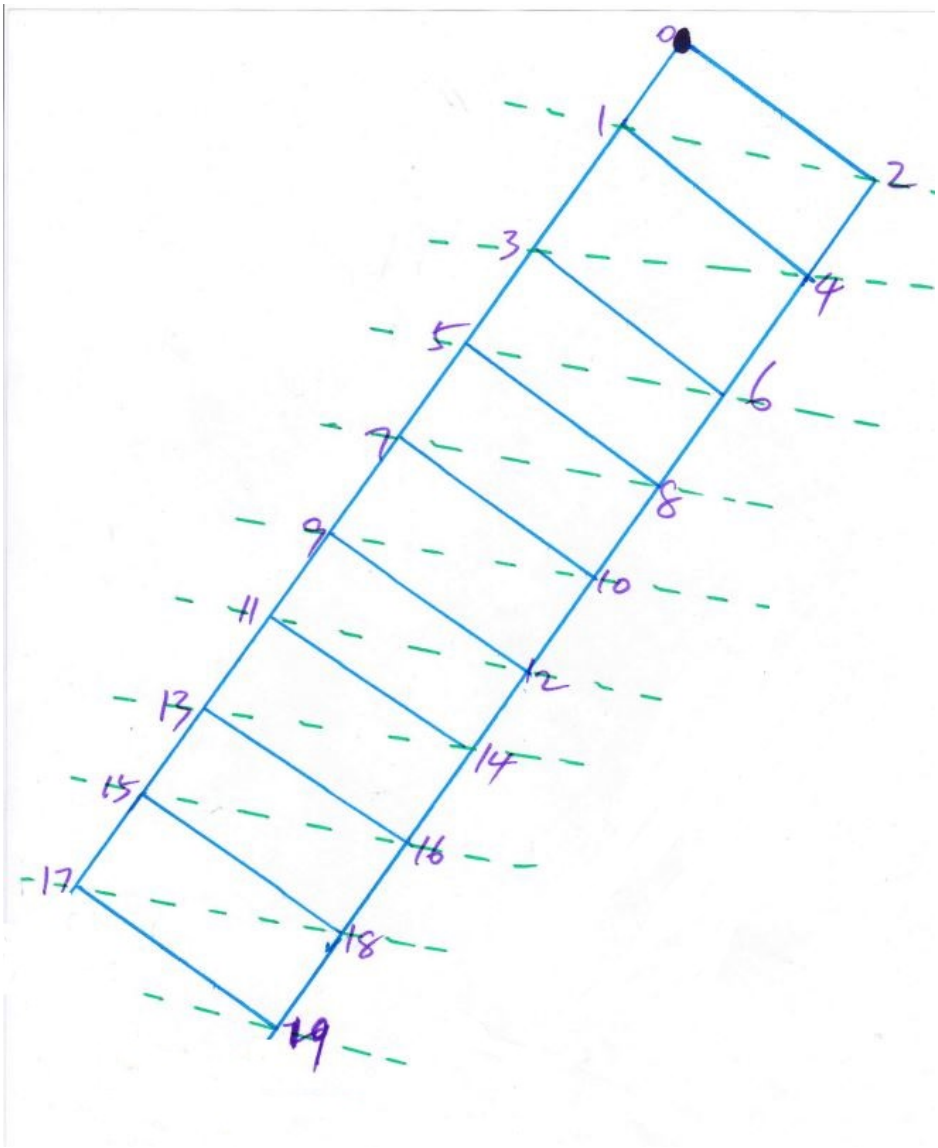

Figure 3: 2*n ($n = 10$) regular mesh

.

Consider more general case Fig. 4.
The closed formula groups are presented as following:

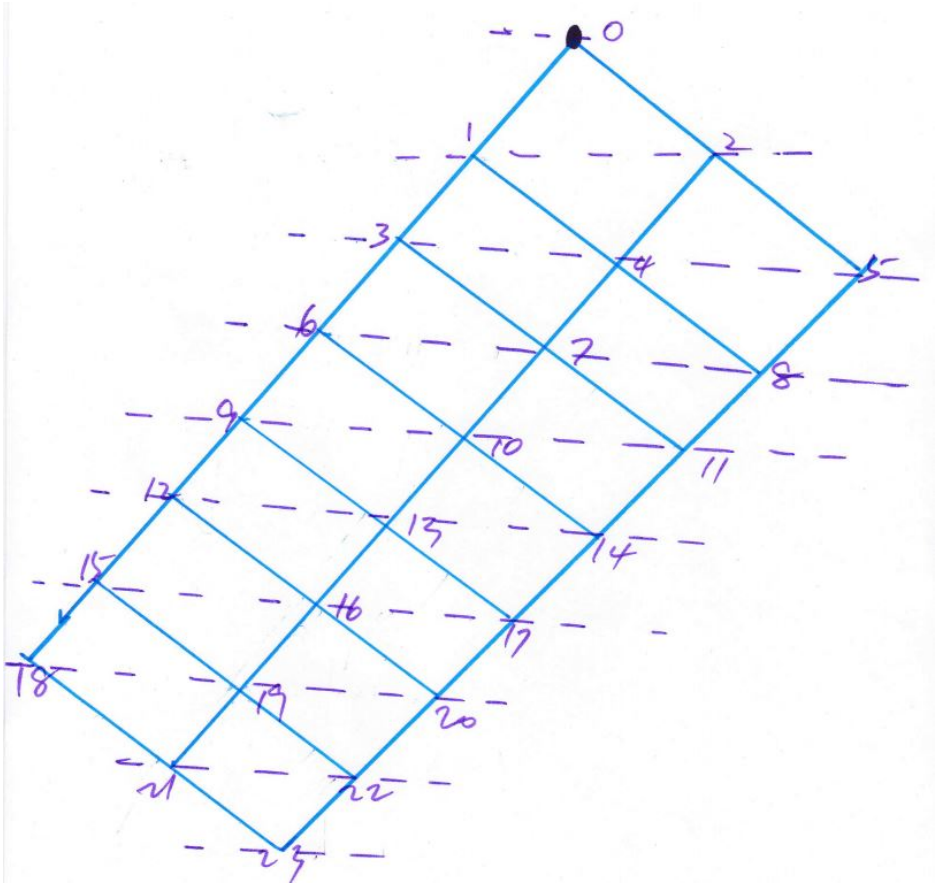

Figure 4: 3*n $n = 8$ regular mesh

$$\begin{bmatrix} 1 & 2 & 3 & 3 & 3 & 3 & 3 & 3 & 2 & 1 \\ 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \sigma-1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \sigma-1 & \sigma & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \sigma-1 & \sigma & \sigma & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & \sigma-1 & \sigma & \sigma & \sigma & 1 & 0 & 0 & 0 & 0 \\ 0 & \sigma-1 & \sigma & \sigma & \sigma & \sigma & 1 & 0 & 0 & 0 \\ 0 & \sigma-1 & \sigma & \sigma & \sigma & \sigma & \sigma & 1 & 0 & 0 \\ 0 & \sigma-1 & \sigma & \sigma & \sigma & \sigma & \sigma & \sigma & 1 & 0 \\ 0 & \sigma-1 & \sigma & \sigma & \sigma & \sigma & \sigma & \sigma & \sigma & 1 \end{bmatrix} \times \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_3 \\ \alpha_6 \\ \alpha_9 \\ \alpha_{12} \\ \alpha_{15} \\ \alpha_{18} \\ \alpha_{21} \\ \alpha_{23} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{4}$$

Consider more general case Fig.5 the closed formula groups are presented as following:

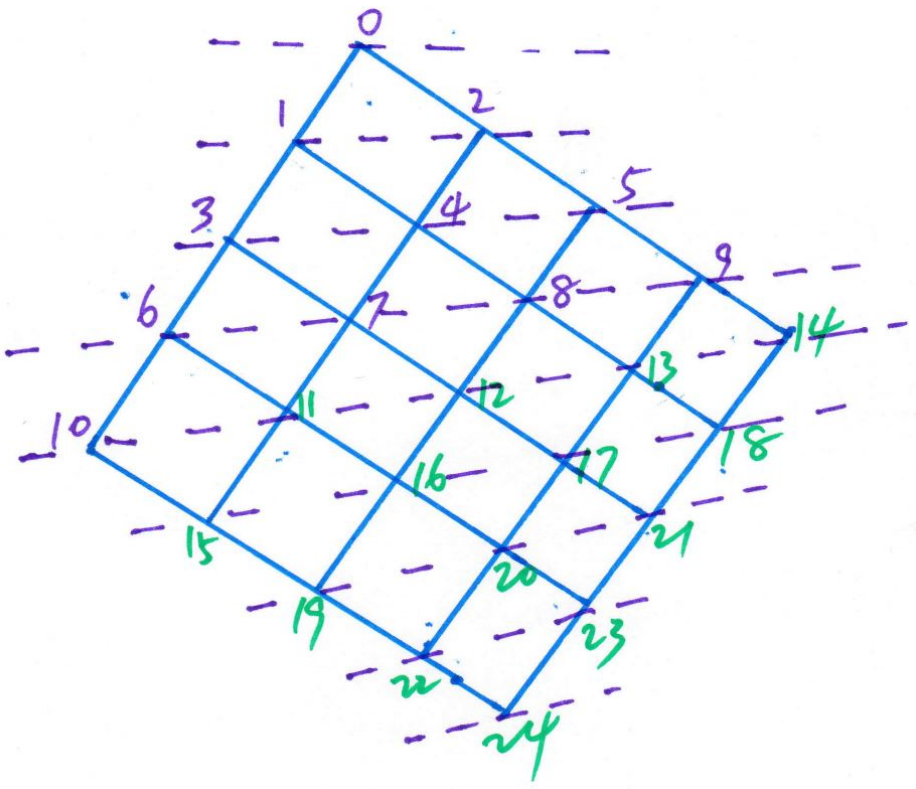

Figure 5: m*n $m = 5$, $n = 5$ regular mesh

$$\begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 4 & 3 & 2 & 1 \\ 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \sigma-1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \sigma-1 & \sigma & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & \sigma-1 & \sigma & \sigma & 1 & 0 & 0 & 0 & 0 \\ 0 & \sigma-1 & \sigma & \sigma & \sigma & 1 & 0 & 0 & 0 \\ 0 & \sigma-1 & \sigma & \sigma & \sigma & \sigma & 1 & 0 & 0 \\ 0 & \sigma-1 & \sigma & \sigma & \sigma & \sigma & \sigma & 1 & 0 \\ 0 & \sigma-1 & \sigma & \sigma & \sigma & \sigma & \sigma & \sigma & 1 \end{bmatrix} \times \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_3 \\ \alpha_6 \\ \alpha_{10} \\ \alpha_{15} \\ \alpha_{19} \\ \alpha_{22} \\ \alpha_{24} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{5}$$

2.1.2. Load From Boundary Grid Position

The data injection position lays on the boundary Fig.6 and the regular mesh is 3*3 situation. The equation for the boundary data injection condition as follows:

$$\alpha_0 \omega T_{cp} = T_{f,m}$$
$$\alpha_1 \omega T_{cp} = T_{f,m}$$
$$\alpha_2 \omega T_{cp} = T_{f,m}$$
$$\alpha_3 \omega T_{cp} = T_{f,m}$$

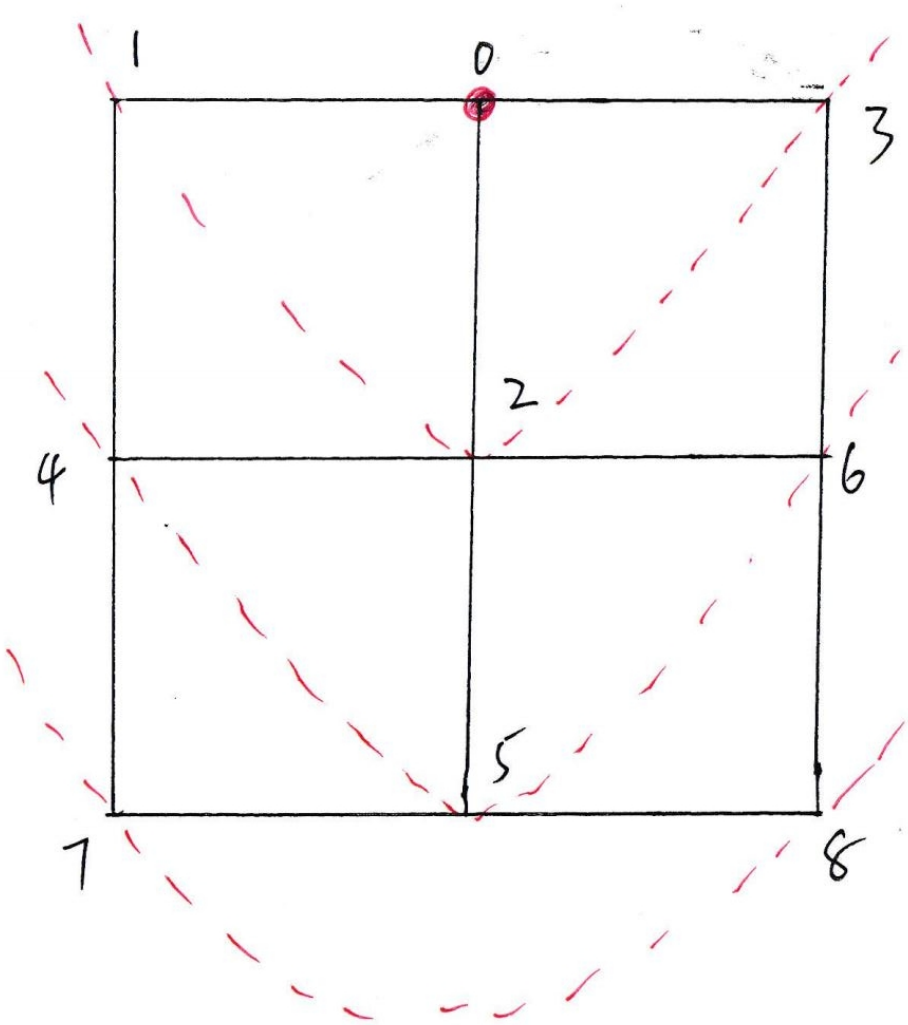

Figure 6: Data injection position on the boundary of 3*3 regular mesh

$$\alpha_1 z T_{cm} + \alpha_4 \omega T_{cp} = T_{f,m}$$
$$\alpha_2 z T_{cm} + \alpha_5 \omega T_{cp} = T_{f,m}$$
$$\alpha_3 z T_{cm} + \alpha_6 \omega T_{cp} = T_{f,m}$$
$$(\alpha_1 + \alpha_4) z T_{cm} + \alpha_7 \omega T_{cp} = T_{f,m}$$
$$(\alpha_2 + \alpha_5) z T_{cm} + \alpha_8 \omega T_{cp} = T_{f,m}$$

$$\sigma = \frac{z T_{cm}}{\omega T_{cp}}$$

$$\begin{bmatrix} 1 & 3 & 3 & 2 \\ 1 & -1 & 0 & 0 \\ 0 & \sigma-1 & 1 & 0 \\ 0 & \sigma-1 & \sigma & 1 \end{bmatrix} \times \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_4 \\ \alpha_7 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{6}$$

2.1.3. Load From Inner Grid

The data injection position lays on the inner grid position Fig.7 and the regular mesh is 3*3 situation.

$$\alpha_0 \omega T_{cp} = T_{f,m}$$
$$\alpha_1 \omega T_{cp} = T_{f,m}$$
$$\alpha_2 \omega T_{cp} = T_{f,m}$$
$$\alpha_3 \omega T_{cp} = T_{f,m}$$
$$\alpha_4 \omega T_{cp} = T_{f,m}$$
$$\alpha_1 z T_{cm} + \alpha_5 \omega T_{cp} = T_{f,m}$$
$$\alpha_1 z T_{cm} + \alpha_6 \omega T_{cp} = T_{f,m}$$
$$\alpha_1 z T_{cm} + \alpha_7 \omega T_{cp} = T_{f,m}$$
$$\alpha_1 z T_{cm} + \alpha_8 \omega T_{cp} = T_{f,m}$$

The equation for the boundary condition as follows:

$$\sigma = \frac{z T_{cm}}{\omega T_{cp}}$$

$$\begin{bmatrix} 1 & 4 & 4 \\ 1 & -1 & 0 \\ 0 & \sigma-1 & 1 \end{bmatrix} \times \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_5 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \tag{7}$$

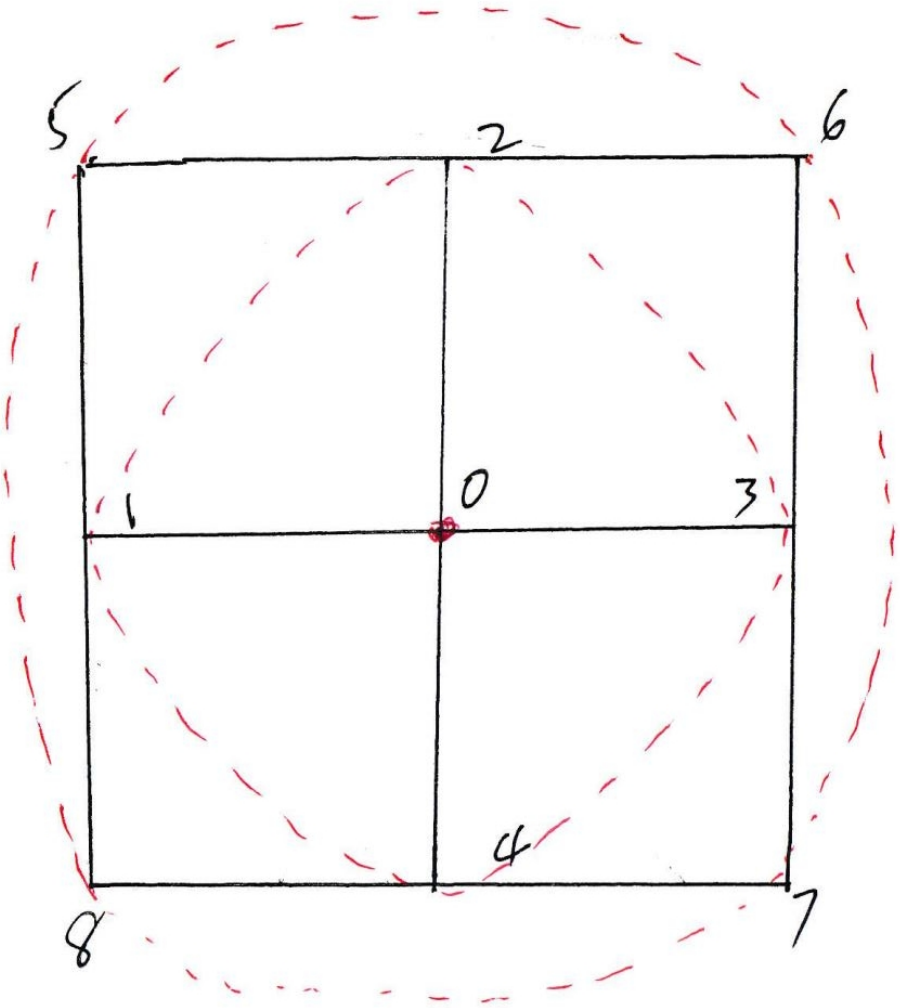

Figure 7: Data injection position on the inner grid position of 3*3 regular mesh

2.2. General Case

$$\begin{bmatrix} 1 & m_1 & m_2 & \cdots & m_{n-2} & m_{n-1} & m_n \\ 1 & -1 & 0 & \cdots & 0 & 0 & 0 \\ 0 & \sigma-1 & 1 & \cdots & 0 & 0 & 0 \\ 0 & \sigma-1 & \sigma & 1 & 0 & \cdots & 0 \\ 0 & \sigma-1 & \sigma & \sigma & 1 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & \\ 0 & \sigma-1 & \sigma & \cdots & \sigma & \sigma & 1 \end{bmatrix} \times \begin{bmatrix} \alpha_{l_0} \\ \alpha_{l_1} \\ \alpha_{l_2} \\ \alpha_{l_3} \\ \vdots \\ \alpha_{l_{n-1}} \\ \alpha_{l_n} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{8}$$

2.2.1. Workstation without front-end

We investigate the simple case first and then deduce the more general closed form formula for the regular mesh situation.

2.3. Load From Corner

- 2*2 regular mesh Fig.1
- 2*3 regular mesh Fig.2
- 2*n regular mesh,for example $n = 10$ Fig.3
- 3*n regular mesh,for example $n = 8$ Fig.4
- m*n regular mesh,for example $m = 5, n = 5$ Fig.5

According to Fig.1,data injection position comes from corner unit processor.
There are four cores to handle the whole workload in the regular mesh.

$$\alpha_0 \omega T_{cp} = T_{f,m}$$
$$\alpha_1 z T_{cm} + \alpha_1 \omega T_{cp} = T_{f,m}$$
$$\alpha_2 z T_{cm} + \alpha_2 \omega T_{cp} = T_{f,m}$$
$$(\alpha_1 + \alpha_3) z T_{cm} + \alpha_3 \omega T_{cp} = T_{f,m}$$
$$\sigma = \frac{z T_{cm}}{\omega T_{cp}}$$
$$\begin{bmatrix} 1 & 2 & 1 \\ 1 & -(\sigma+1) & 0 \\ 1 & -\sigma & -(\sigma+1) \end{bmatrix} \times \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \tag{9}$$

According to Fig.2, the formula groups are

$$\alpha_0 \omega T_{cp} = T_{f,m}$$
$$\alpha_1 z T_{cm} + \alpha_1 \omega T_{cp} = T_{f,m}$$
$$\alpha_2 z T_{cm} + \alpha_2 \omega T_{cp} = T_{f,m}$$
$$(\alpha_1 + \alpha_3) z T_{cm} + \alpha_3 \omega T_{cp} = T_{f,m}$$
$$(\alpha_1 + \alpha_4) z T_{cm} + \alpha_4 \omega T_{cp} = T_{f,m}$$
$$(\alpha_1 + \alpha_3 + \alpha_5) z T_{cm} + \alpha_5 \omega T_{cp} = T_{f,m}$$

$$\begin{bmatrix} 1 & 2 & 2 & 1 \\ 1 & -(\sigma+1) & 0 & 0 \\ 1 & -\sigma & -(\sigma+1) & 0 \\ 1 & -\sigma & -\sigma & -(\sigma+1) \end{bmatrix} \times \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_3 \\ \alpha_5 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{10}$$

$$\sigma = \frac{zT_{cm}}{\omega T_{cp}}$$

the speedup ratio is: $\frac{1}{\alpha_0}$

According to the 2*n regular mesh, the formula equation group as follows:

$$\alpha_1 zT_{cm} + \alpha_1 \omega T_{cp} = T_{f,m}$$
$$\alpha_2 zT_{cm} + \alpha_2 \omega T_{cp} = T_{f,m}$$
$$(\alpha_1 + \alpha_3) zT_{cm} + \alpha_3 \omega T_{cp} = T_{f,m}$$
$$(\alpha_1 + \alpha_4) zT_{cm} + \alpha_4 \omega T_{cp} = T_{f,m}$$

$$(\alpha_1 + \alpha_3 + \alpha_5) zT_{cm} + \alpha_5 \omega T_{cp} = T_{f,m}$$
$$\vdots$$
$$(\alpha_1 + \alpha_3 + \cdots + \alpha_{2\times n+1}) zT_{cm} + \alpha_{2\times n+1} \omega T_{cp} = T_{f,m}$$

$$\sigma = \frac{zT_{cm}}{\omega T_{cp}}$$

$$\begin{bmatrix} 1 & 2 & 2 & \cdots & 2 & 2 & 1 \\ 1 & -(\sigma+1) & 0 & \cdots & 0 & 0 & 0 \\ 1 & -\sigma & -(\sigma+1) & \cdots & 0 & 0 & 0 \\ 1 & -\sigma & -\sigma & -(\sigma+1) & 0 & \cdots & 0 \\ 1 & -\sigma & -\sigma & -\sigma & -(\sigma+1) & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & \\ 1 & -\sigma & -\sigma & \cdots & -\sigma & -\sigma & -(\sigma+1) \end{bmatrix} \times \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_3 \\ \alpha_5 \\ \vdots \\ \alpha_{2\times n-1} \\ \alpha_{2\times n+1} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{11}$$

So the Speedup is

$$\frac{1}{\alpha_0}$$

According to the Fig.4, the matrix is: We use $\sigma^\star$ to present the $-(\sigma+1)$

$$\begin{bmatrix} 1 & 2 & 3 & 3 & 3 & 3 & 3 & 3 & 2 & 1 \\ 1 & \sigma^\star & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & -\sigma & \sigma^\star & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & -\sigma & -\sigma & \sigma^\star & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & -\sigma & -\sigma & -\sigma & \sigma^\star & 0 & 0 & 0 & 0 & 0 \\ 1 & -\sigma & -\sigma & -\sigma & -\sigma & \sigma^\star & 0 & 0 & 0 & 0 \\ 1 & -\sigma & -\sigma & -\sigma & -\sigma & -\sigma & \sigma^\star & 0 & 0 & 0 \\ 1 & -\sigma & -\sigma & -\sigma & -\sigma & -\sigma & -\sigma & \sigma^\star & 0 & 0 \\ 1 & -\sigma & -\sigma & -\sigma & -\sigma & -\sigma & -\sigma & -\sigma & \sigma^\star & 0 \\ 1 & -\sigma & -\sigma & -\sigma & -\sigma & -\sigma & -\sigma & -\sigma & -\sigma & -\sigma^\star \end{bmatrix} \times \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_3 \\ \alpha_6 \\ \alpha_9 \\ \alpha_{12} \\ \alpha_{15} \\ \alpha_{18} \\ \alpha_{21} \\ \alpha_{23} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{12}$$

According to the Fig. 5, the formula groups as follows: We use $\sigma^\star$ to present the $-(\sigma+1)$

$$\begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 4 & 3 & 2 & 1 \\ 1 & \sigma^\star & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & -\sigma & \sigma^\star & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & -\sigma & -\sigma & \sigma^\star & 0 & 0 & 0 & 0 & 0 \\ 1 & -\sigma & -\sigma & -\sigma & \sigma^\star & 0 & 0 & 0 & 0 \\ 1 & -\sigma & -\sigma & -\sigma & -\sigma & \sigma^\star & 0 & 0 & 0 \\ 1 & -\sigma & -\sigma & -\sigma & -\sigma & -\sigma & \sigma^\star & 0 & 0 \\ 1 & -\sigma & -\sigma & -\sigma & -\sigma & -\sigma & -\sigma & \sigma^\star & 0 \\ 1 & -\sigma & -\sigma & -\sigma & -\sigma & -\sigma & -\sigma & -\sigma & \sigma^\star \end{bmatrix} \times \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_3 \\ \alpha_6 \\ \alpha_{10} \\ \alpha_{15} \\ \alpha_{19} \\ \alpha_{22} \\ \alpha_{24} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{13}$$

2.3.1. Load From Boundary Grid Position

The data injection position lays on the boundary Fig.6 and the regular mesh is 3*3 situation.

$$\alpha_0 \omega T_{cp} = T_{f,m}$$
$$\alpha_1 z T_{cm} + \alpha_1 \omega T_{cp} = T_{f,m}$$
$$\alpha_2 z T_{cm} + \alpha_2 \omega T_{cp} = T_{f,m}$$
$$\alpha_3 z T_{cm} + \alpha_3 \omega T_{cp} = T_{f,m}$$
$$(\alpha_1 + \alpha_4) z T_{cm} + \alpha_4 \omega T_{cp} = T_{f,m}$$
$$(\alpha_2 + \alpha_5) z T_{cm} + \alpha_5 \omega T_{cp} = T_{f,m}$$
$$(\alpha_3 + \alpha_6) z T_{cm} + \alpha_6 \omega T_{cp} = T_{f,m}$$
$$(\alpha_1 + \alpha_4 + \alpha_7) z T_{cm} + \alpha_7 \omega T_{cp} = T_{f,m}$$
$$(\alpha_1 + \alpha_4 + \alpha_8) z T_{cm} + \alpha_8 \omega T_{cp} = T_{f,m}$$

The equation for the boundary condition as follows:

$$\sigma = \frac{zT_{cm}}{\omega T_{cp}}$$

$$\begin{bmatrix} 1 & 3 & 3 & 2 \\ 1 & -(\sigma+1) & 0 & 0 \\ 1 & -\sigma & -(\sigma+1) & 0 \\ 1 & -\sigma & -\sigma & -(\sigma+1) \end{bmatrix} \times \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_4 \\ \alpha_7 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{14}$$

2.3.2. Load From Inner Grid

The data injection position lays on the inner grid position Fig.7 and the regular mesh is 3*3 situation.

$$\alpha_0 \omega T_{cp} = T_{f,m}$$
$$\alpha_1 z T_{cm} + \alpha_1 \omega T_{cp} = T_{f,m}$$
$$\alpha_2 z T_{cm} + \alpha_2 \omega T_{cp} = T_{f,m}$$
$$\alpha_3 z T_{cm} + \alpha_3 \omega T_{cp} = T_{f,m}$$
$$\alpha_4 z T_{cm} + \alpha_4 \omega T_{cp} = T_{f,m}$$
$$(\alpha_1 + \alpha_5) z T_{cm} + \alpha_5 \omega T_{cp} = T_{f,m}$$
$$(\alpha_2 + \alpha_6) z T_{cm} + \alpha_6 \omega T_{cp} = T_{f,m}$$
$$(\alpha_3 + \alpha_7) z T_{cm} + \alpha_7 \omega T_{cp} = T_{f,m}$$
$$(\alpha_4 + \alpha_8) z T_{cm} + \alpha_8 \omega T_{cp} = T_{f,m}$$

The equation for the boundary data injection condition as follows:

$$\sigma = \frac{zT_{cm}}{\omega T_{cp}}$$

$$\begin{bmatrix} 1 & 4 & 4 \\ 1 & -(\sigma+1) & 0 \\ 1 & -\sigma & -(\sigma+1) \end{bmatrix} \times \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_5 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \tag{15}$$

2.3.3. General Case

$$\begin{bmatrix} 1 & m_1 & m_2 & \cdots & m_{n-2} & m_{n-1} & m_n \\ 1 & -(\sigma+1) & 0 & \cdots & 0 & 0 & 0 \\ 1 & -\sigma & -(\sigma+1) & \cdots & 0 & 0 & 0 \\ 1 & -\sigma & -\sigma & -(\sigma+1) & 0 & \cdots & 0 \\ 1 & -\sigma & -\sigma & -\sigma & -(\sigma+1) & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & \\ 1 & -\sigma & -\sigma & \cdots & -\sigma & -\sigma & -(\sigma+1) \end{bmatrix} \times \begin{bmatrix} \alpha_{l_0} \\ \alpha_{l_1} \\ \alpha_{l_2} \\ \alpha_{l_3} \\ \vdots \\ \alpha_{l_{n-1}} \\ \alpha_{l_n} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \quad (16)$$

2.3.4. Speedup Result between front-end schema and without front-end schema

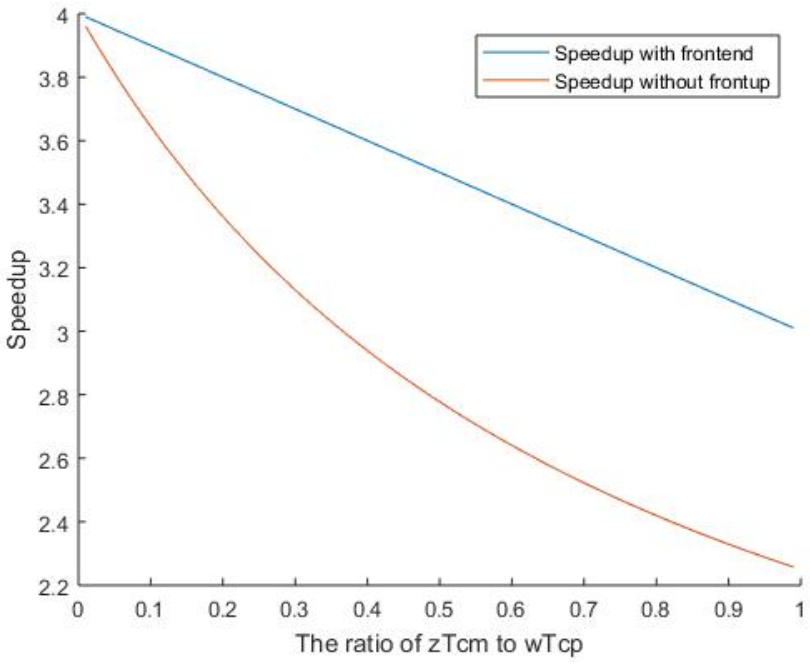

Figure 8: Speedup vs σ value in 2*2 regular mesh

3. Sensitivity Analysis

In this section, we investigate the sensitivity of regular mesh equal speedup analysis.

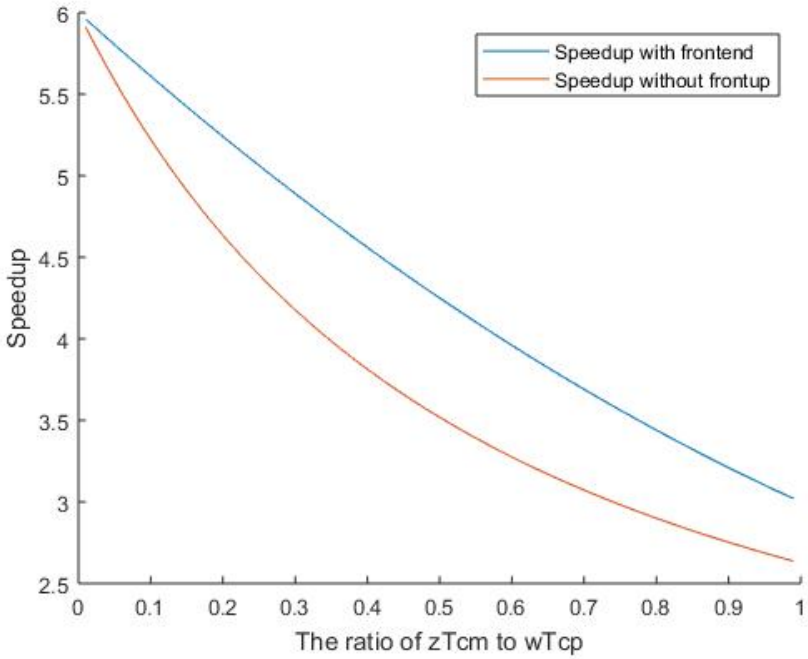

Figure 9: Speedup vs σ value in 2*3 regular mesh

3.1. Front End Schema

*3.1.1. 2*n regular mesh*

Considering the $2 * n$ regular mesh Fig .3 and the data injection position on the corner. The speedup vs the number of cores relationship Fig.13 and Fig.14 as follows:

- we can see as the number of cores grows and the equal computational grow as well.
- At the same, the σ value plays an import role, especially $\sigma > 0.25$. The speedup drops dramatically.

*3.1.2. 3*n regular mesh simulation result*

Considering the $3 * n$ regular mesh Fig .4 Fig.16,which represents the speedup vs the number of cores relationship as follows Fig.15:

We can see from Fig.16.

- If the $\sigma > 0.15$, the boundary data injection plan has positive speedup effect comparing with the corner injection plan.

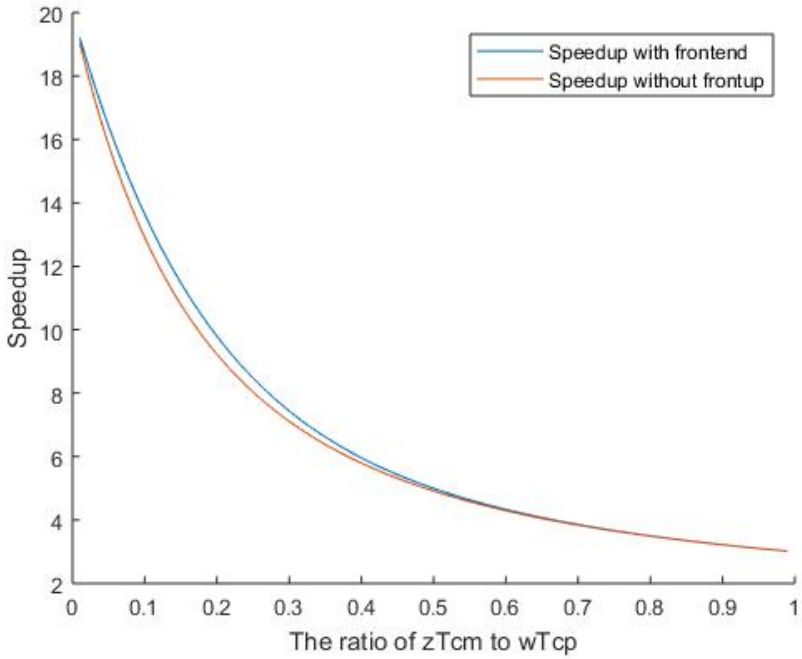

Figure 10: Speedup vs σ value in 2*10 regular mesh

- If the $\sigma <= 0.15$, the corner data injection plan will play a more helpful role.

*3.1.3. 5*5 regular mesh inner grid simulation result*

Considering the $5 * 5$ regular mesh Fig .5,which represents the speedup vs the number of cores relationship as follows:

The simulation result as follows Fig.17:

3.2. Without Front End Schema

*3.2.1. 2*n regular mesh*

Considering the $2 * n$ regular mesh Fig.3.The data injection position is on the corner.
The speedup vs the number of cores relationship as follows:

- we can see as the number of cores grows and the equal computational grow as well.
- At the same, the σ value plays an import role, especially $\sigma > 0.25$. The speedup drops dramatically.

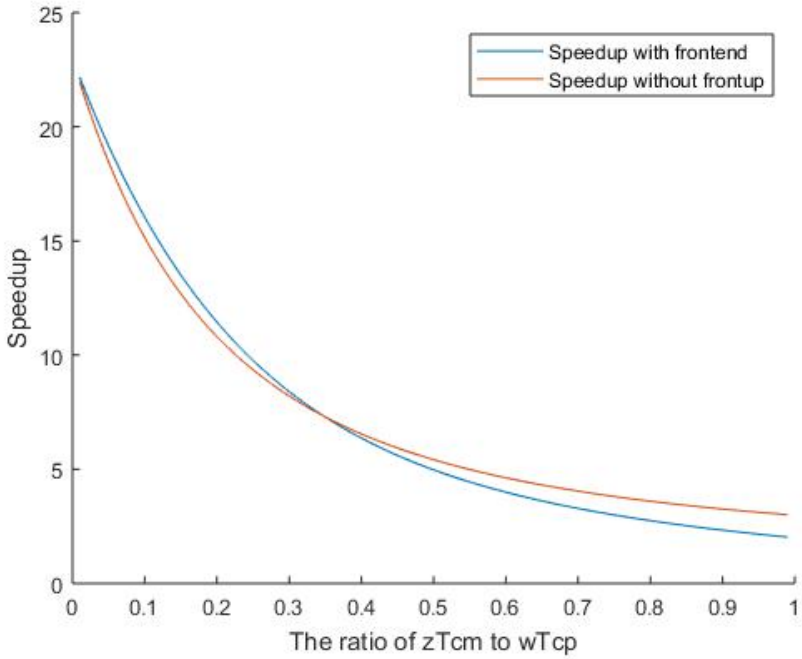

Figure 11: Speedup vs σ value in 3*8 regular mesh

*3.2.2. 3*n regular mesh simulation result*

Considering the $3 * n$ regular mesh Fig.4.

The speedup vs the number of cores relationship as follows:

- If the $\sigma > 0.2$, the boundary data injection plan has positive speedup effect comparing with the corner injection plan.
- If the $\sigma <= 0.2$, the corner data injection plan will play a more helpful role.

*3.2.3. 5*5 regular mesh inner grid simulation result*

Considering the $5*5$ regular mesh Fig.5 and the data injection position is on the $position = 12$. The simulation result as follows Fig.22:

4. Re-balance Voronoi Diagram Multi-source load injection

In this section, we introduce the Voronoi diagram technique to address the data injection position and data injection community division problem.
We will consider these problems from the following perspective:

- data load position dense or sparse
- data load injection fraction even or different
- how to choose data load injection position to achieve the high efficiency.

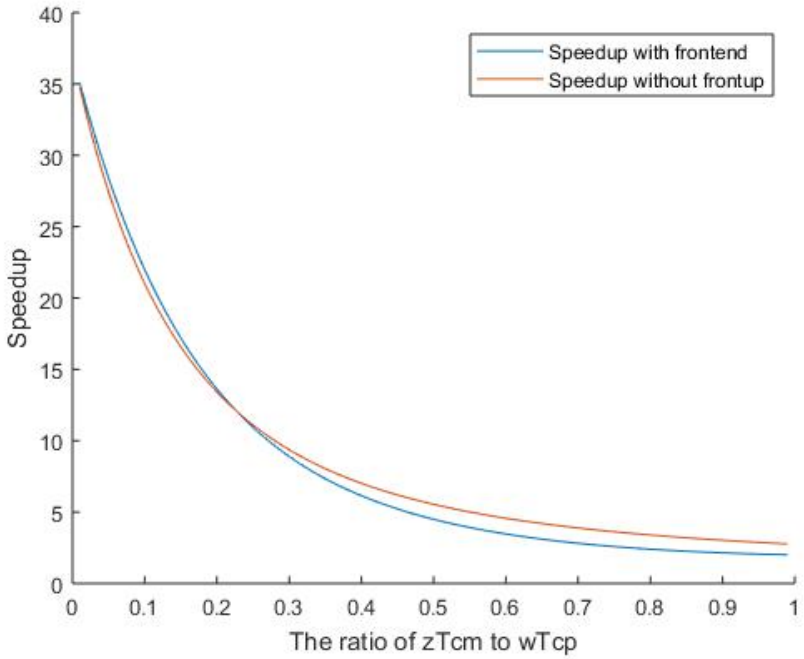

Figure 12: Speedup vs σ value in 4*10 regular mesh

4.1. General Case

We use Manhattan Distance to divide the regular mesh area to Voronoi Cell [3].
We can see the Fig.28 shows the simulation result.

4.2. Data Injection Position Dense

If the data injection positions are dense, which means the data injection position connect with each other.

The sun-optimal solution is that

- we consider the whole cluster data load injection as one data injection
- execute the equal power algorithm in chapter one to process the divisible load theory principle.
- re-balance the load fraction between nodes in the cluster

Consider the Fig.24

The equal power matrix with front end as follows:

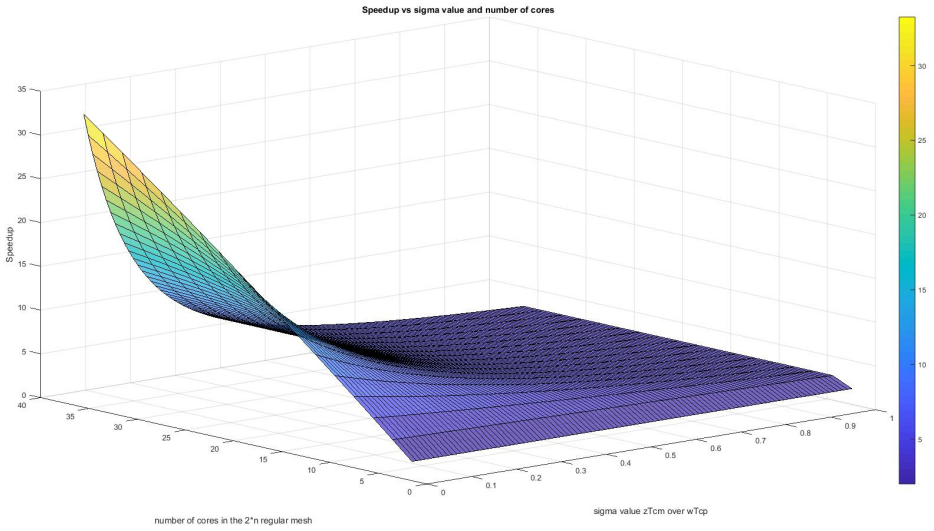

Figure 13: Speedup vs σ value and number of cores in 2*n regular mesh

$$\begin{bmatrix} 4 & 8 & 10 & 6 & 2 \\ 1 & -1 & 0 & 0 & 0 \\ 0 & \sigma-1 & 1 & 0 & 0 \\ 0 & \sigma-1 & \sigma & 1 & 0 \\ 0 & \sigma-1 & \sigma & \sigma & 1 \end{bmatrix} \tag{17}$$

The equal power matrix without front end as follows:

$$\begin{bmatrix} 4 & 8 & 10 & 6 & 2 \\ 1 & \sigma^\star & 0 & 0 & 0 \\ 1 & -\sigma & \sigma^\star & 0 & 0 \\ 1 & -\sigma & -\sigma & \sigma^\star & 0 \\ 1 & -\sigma & -\sigma & -\sigma & \sigma^\star \end{bmatrix} \tag{18}$$

Consider the Fig.25

The equal power matrix with front end as follows:

$$\begin{bmatrix} 7 & 14 & 15 & 10 & 6 & 3 & 1 \\ 1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & \sigma-1 & 1 & 0 & 0 & 0 & 0 \\ 0 & \sigma-1 & \sigma & 1 & 0 & 0 & 0 \\ 0 & \sigma-1 & \sigma & \sigma & 1 & 0 & 0 \\ 0 & \sigma-1 & \sigma & \sigma & \sigma & 1 & 0 \\ 0 & \sigma-1 & \sigma & \sigma & \sigma & \sigma & 1 \end{bmatrix} \tag{19}$$

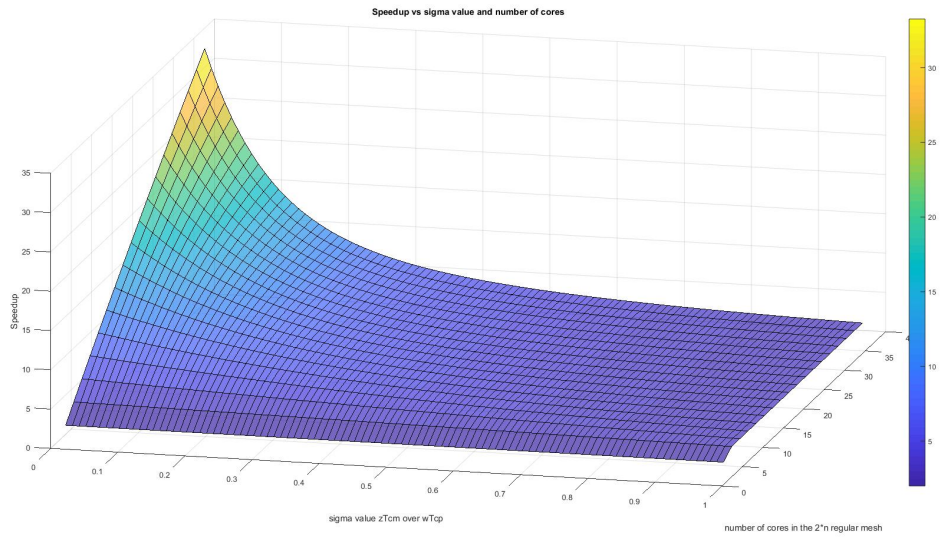

Figure 14: Speedup vs σ value and number of cores in 2*n regular mesh

The equal power matrix without front end as follows:

$$\begin{bmatrix} 7 & 14 & 15 & 10 & 6 & 3 & 1 \\ 1 & \sigma^\star & 0 & 0 & 0 & 0 & 0 \\ 1 & -\sigma & \sigma^\star & 0 & 0 & 0 & 0 \\ 1 & -\sigma & -\sigma & \sigma^\star & 0 & 0 & 0 \\ 1 & -\sigma & -\sigma & -\sigma & \sigma^\star & 0 & 0 \\ 1 & -\sigma & -\sigma & -\sigma & -\sigma & \sigma^\star & 0 \\ 1 & -\sigma & -\sigma & -\sigma & -\sigma & -\sigma & \sigma^\star \end{bmatrix} \tag{20}$$

So this kind of problem is transfered to finding the number of node on each level(contour line).

4.3. Choosing The Data Load Position

The data injection position are affected by two factors.

- the distance between each data load injection
- the number of unit cores in its Voronoi cell community.

So we choose the Manhattan distance centroidal voronoi tessellations[4] to choose the data injection position.
For example, we consider the 80*80 regular mesh, we have 10 data injection budget to choose 10 appropriate position.

We can see the simulation result from Fig.26 and Fig.27

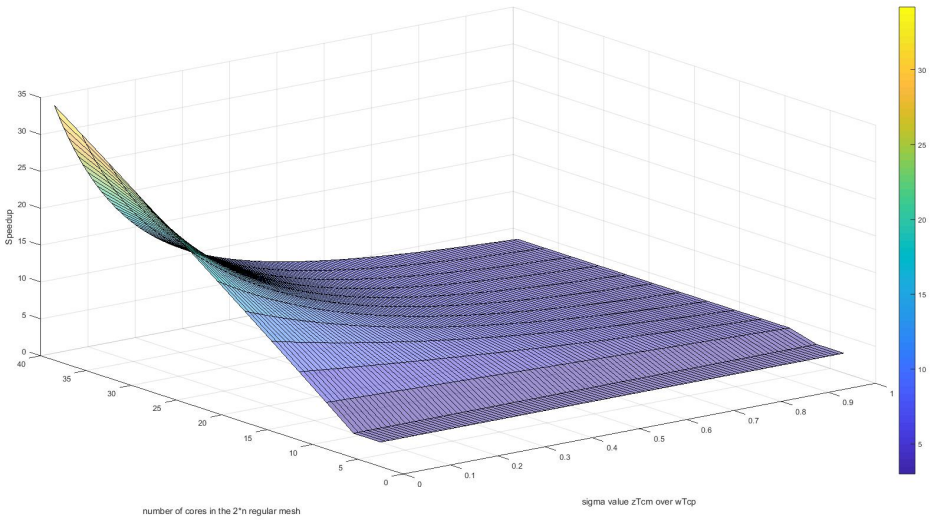

Figure 15: Speedup vs σ value and number of cores in 3*n regular mesh

4.4. Re-balance the community division

Considering the speedup ability depends on the number of short distance unit core. So after 1000 times random experiment, we find we can achieve the same computation ability yet save over 30% unit core.

The principle is we choose the minimum value of the largest depth of each community as the depth rule.

$$L = min(max(voronoi \quad cell \quad depth))$$

One example Fig.28 Fig.29 presents:

We also can find the speedup result from figures Fig.35 and Fig.31:

- $\sigma < 0.2$ the ratio is 5. Fig.35
- after re-balance the ratio is about 2.7 31

5. Optimal Mass Transport

In this section,we extend the optimal mass transport theory algorithm framework [5] [6] to address the data injection division problem.

The original method depends on the rigorous geometry structure,yet in the real application,the implementation is hard and not stable because of the numerical calculation errors.

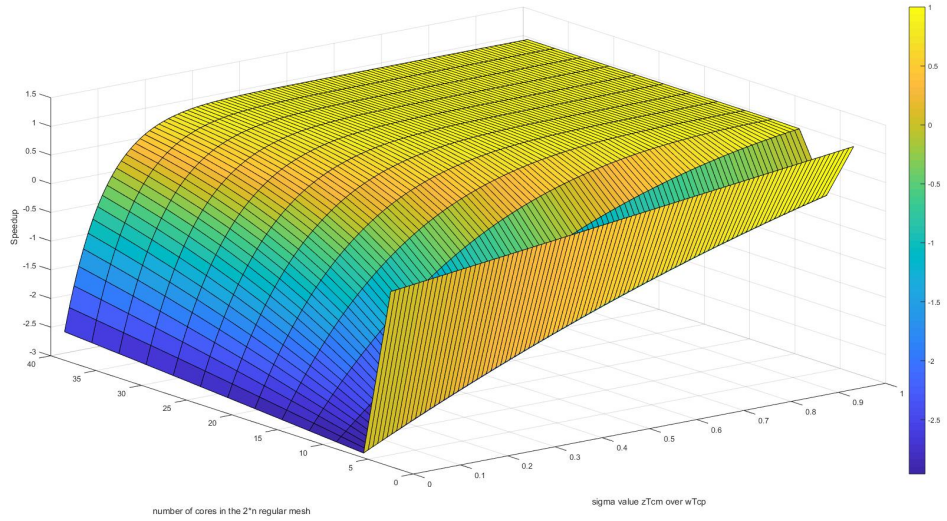

Figure 16: Speedup Difference between Corner and Boundary Data Injection in 3*n regular mesh

I re-implement this framework using Monte Carlo method, which can be extend to N dimension. In addition, the implement is simple and we get the appropriate accurate effect as adding more sampling points.

Let's consider two situations as example.

- Given an initial division,try to redivide the regular mesh to even division using the minimum cost. For example, the number of unit core ratio of 10 pieces is [0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4 0.4]
- Give an initial division,try to redivide the regular mesh to a target measure division using the minimum cost.For example,the number of unit core ratio of 10 pieces is [0.4 0.3 0.3 0.4 0.2 0.2 0.3 0.4 0.7 0.9]

5.1. Situation 1

The initial division as follows Fig.38:

The optimal mass transport even division as follows Fig.33:

After re-choose the data injection position as follows Fig.34:

The speedup ratio as follows:

The initial speedup figure. Fig.35

Re-balance optimize speedup figure Fig.36

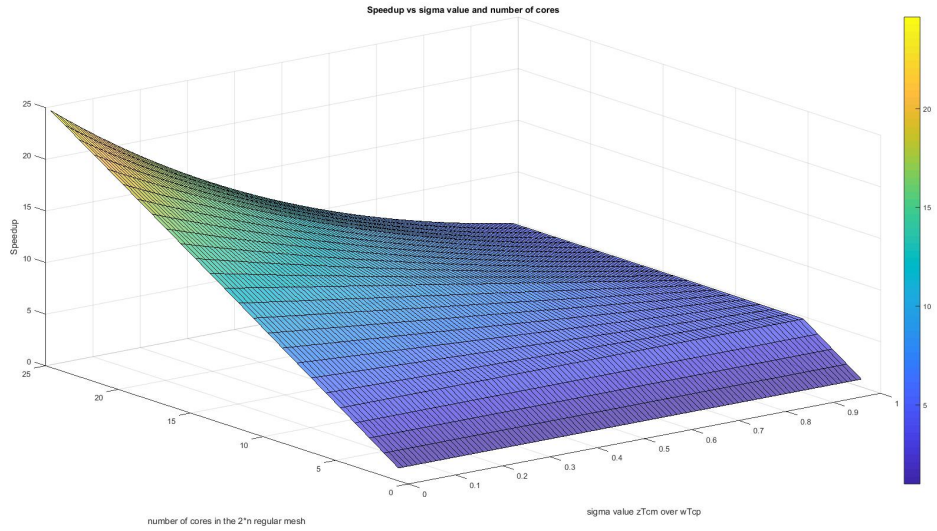

Figure 17: Speedup vs σ value and number of cores in 4*n regular mesh

The optimal mass transport speedup figure Fig.37

5.2. Situation 2

The optimal mass transport even division as follows:

After re-choose the data injection position as follows:

The speedup ratio as follows:
The initial speedup figure. Fig.41
Re-balance Voronoi optimization speedup figure Fig.42

The optimal mass transport speedup figure Fig.43

6. Experiment

7. Conclusion

[1] J. Jia, B. Veeravalli, J. Weissman, Scheduling multisource divisible loads on arbitrary networks, IEEE Transactions on Parallel and Distributed Systems 21 (2010) 520–531.

[2] T. G. Robertazzi, Processor equivalence for daisy chain load sharing processors, IEEE Transactions on Aerospace and Electronic Systems 29 (1993) 1216–1221.

[3] S. Fortune, A sweepline algorithm for voronoi diagrams, Algorithmica 2 (1987) 153.

[4] Q. Du, V. Faber, M. Gunzburger, Centroidal voronoi tessellations: Applications and algorithms, SIAM review 41 (1999) 637–676.

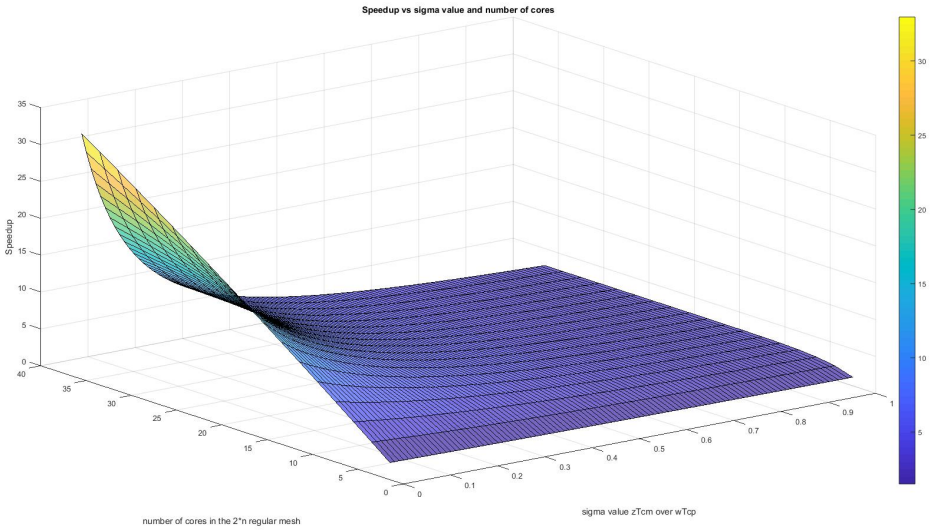

Figure 18: Speedup vs σ value and number of cores in 2*n regular mesh

[5] X. Gu, F. Luo, J. Sun, S.-T. Yau, Variational principles for minkowski type problems, discrete optimal transport, and discrete monge-ampere equations, arXiv preprint arXiv:1302.5472 (2013).

[6] K. Su, W. Chen, N. Lei, J. Zhang, K. Qian, X. Gu, Volume preserving mesh parameterization based on optimal mass transportation, Computer-Aided Design 82 (2017) 42–56.

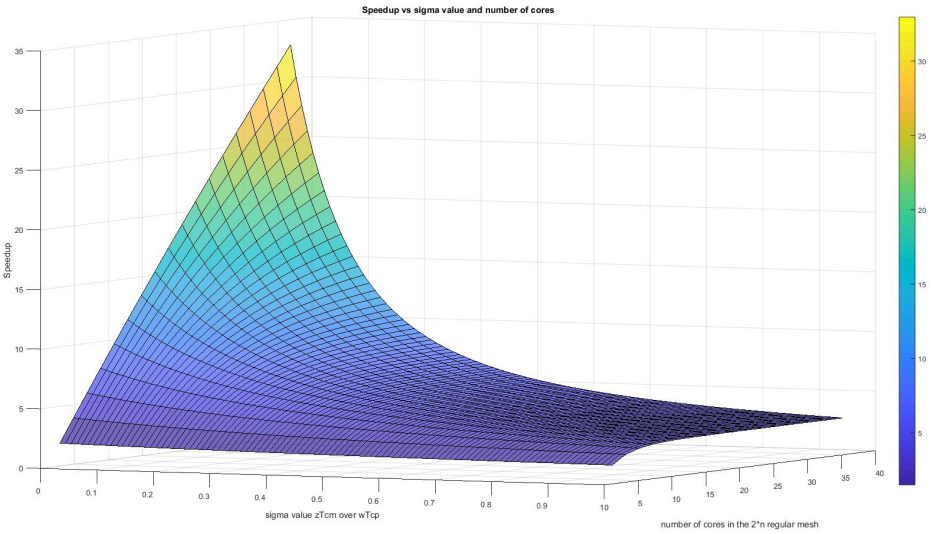

Figure 19: Speedup vs σ value and number of cores in 2*n regular mesh

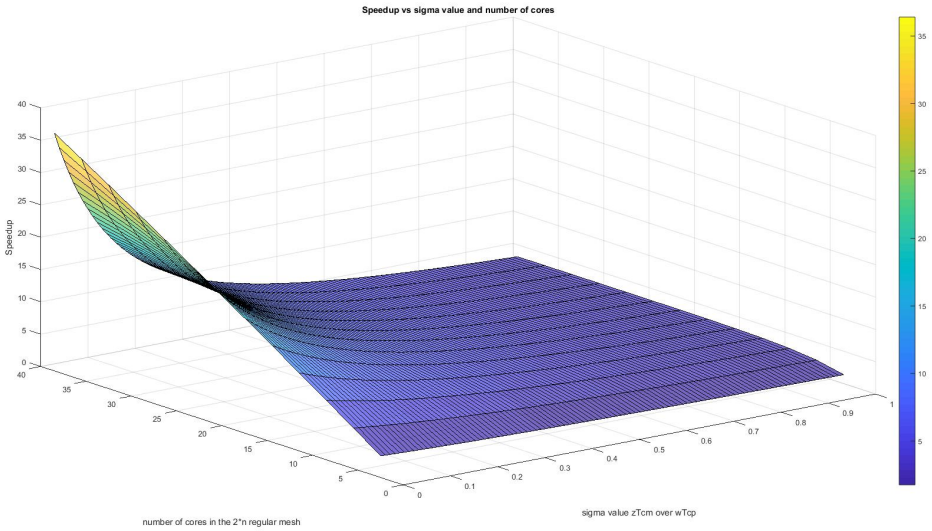

Figure 20: Speedup vs σ value and number of cores in 3*n regular mesh

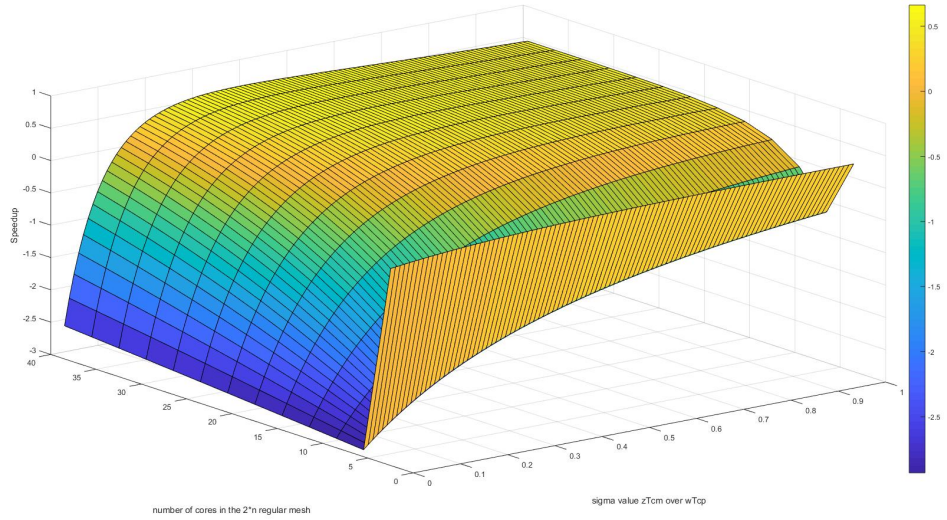

Figure 21: Speedup Difference between Corner and Boundary Data Injection in 3*n regular mesh

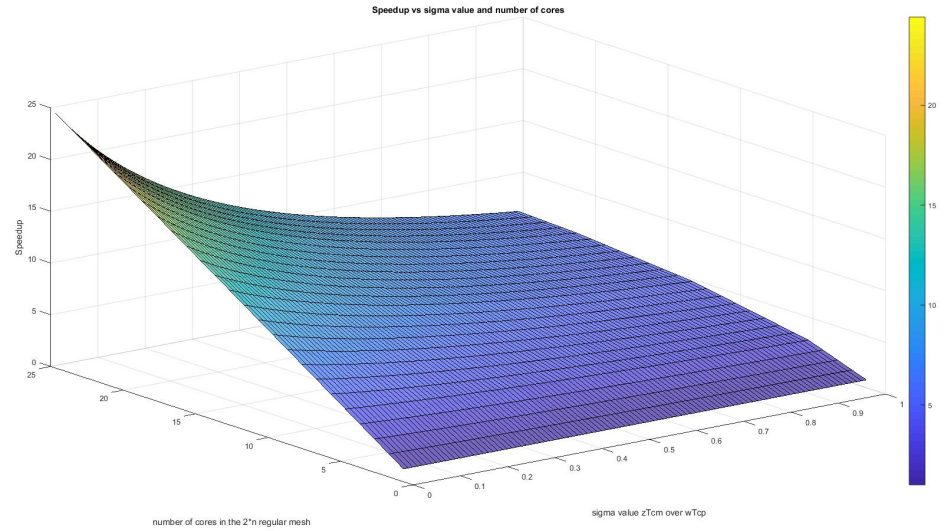

Figure 22: Speedup vs σ value and number of cores in 4*n regular mesh

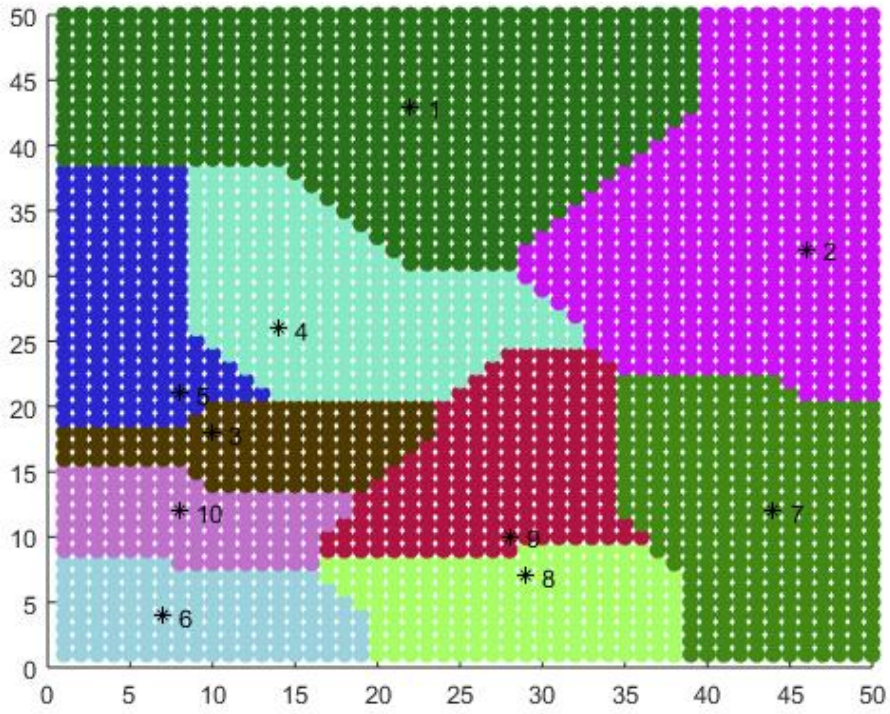

Figure 23: 50*50 regular mesh with 10 data injection community division

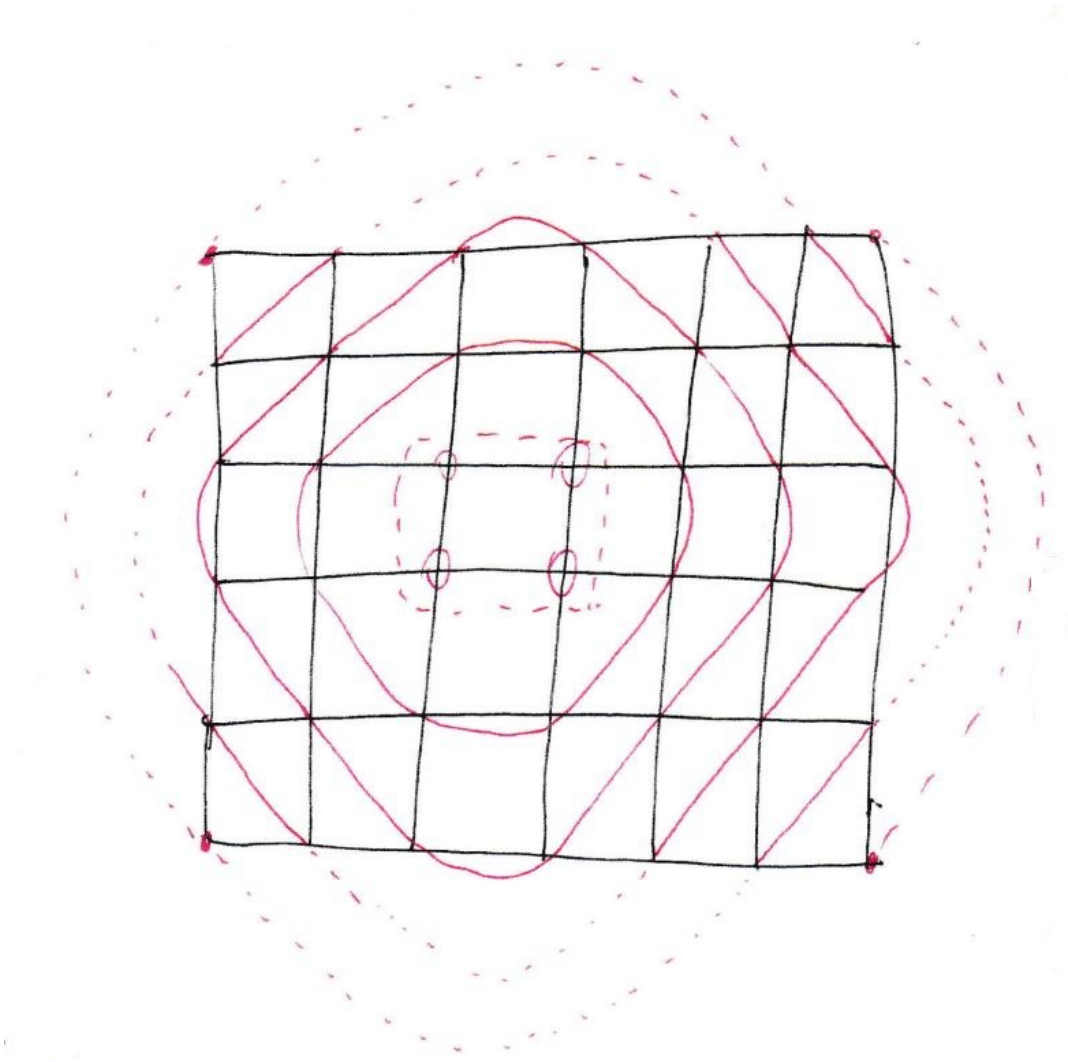

Figure 24: 4 data injection connecting with each other consists of a whole cluster

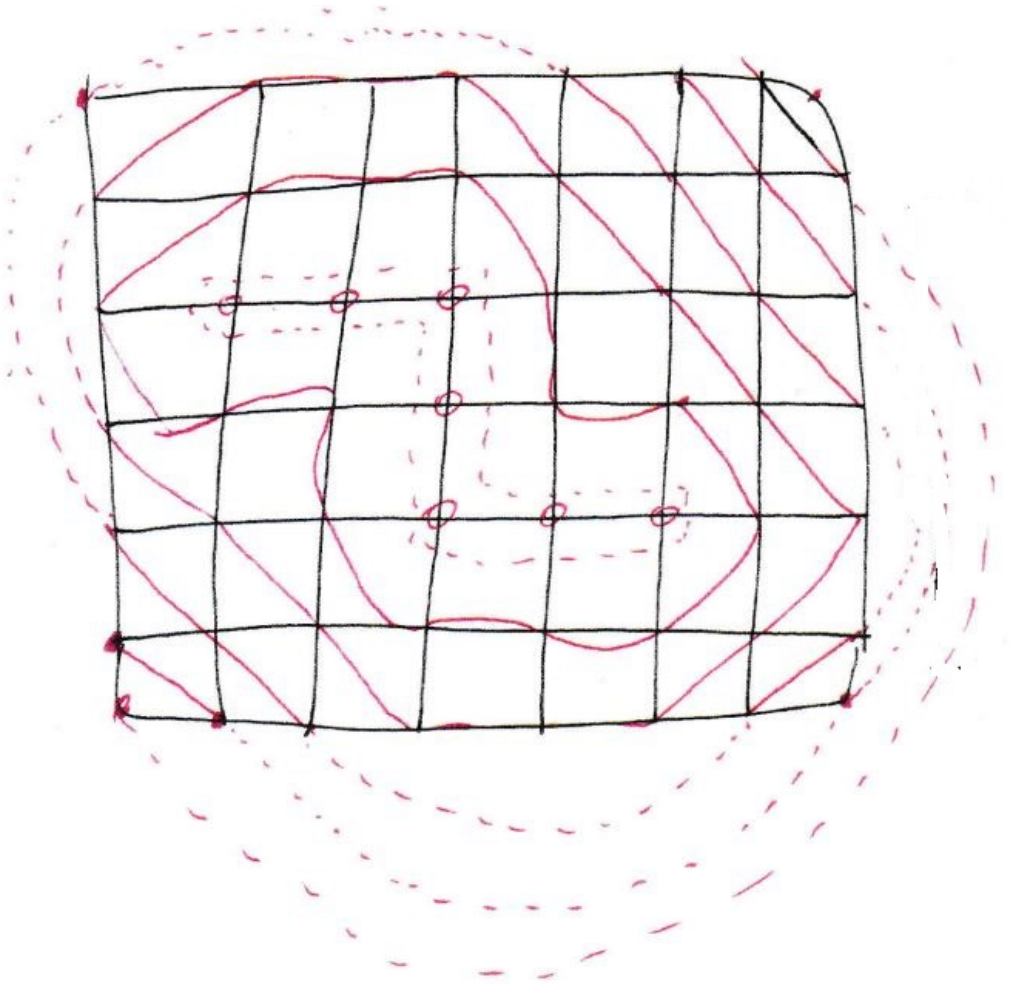

Figure 25: 7 data injection connecting with each other consist of a whole cluster

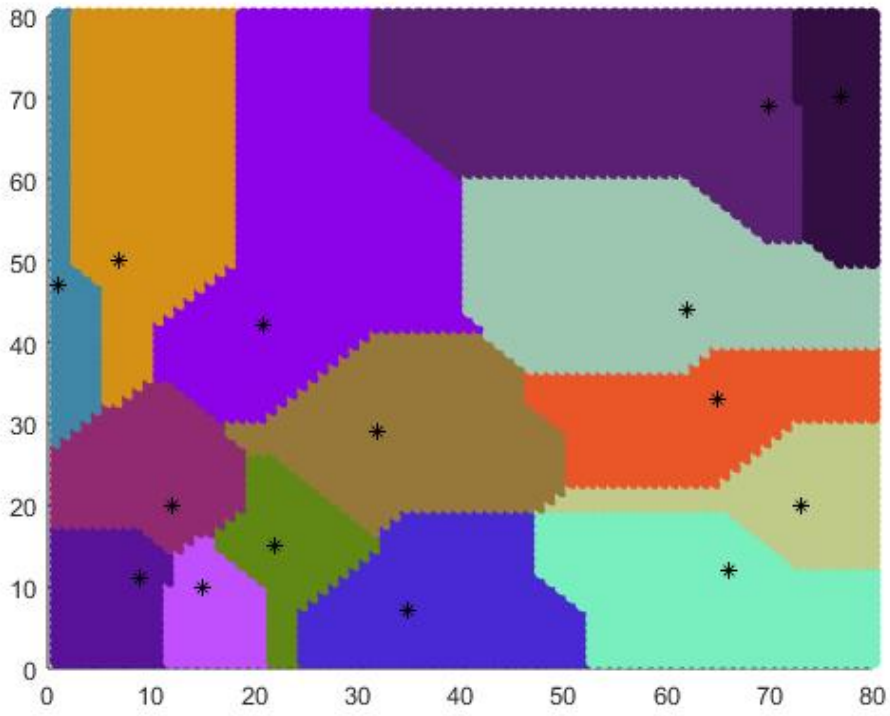

Figure 26: Random choose 10 potential position to do the community division

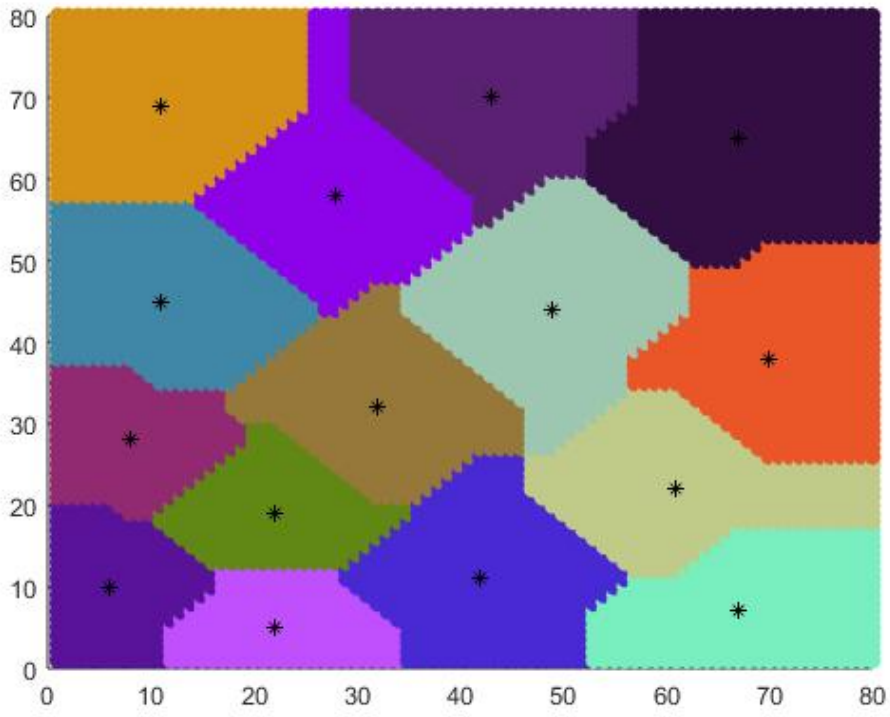

Figure 27: After 20 rounds the division achieve stable

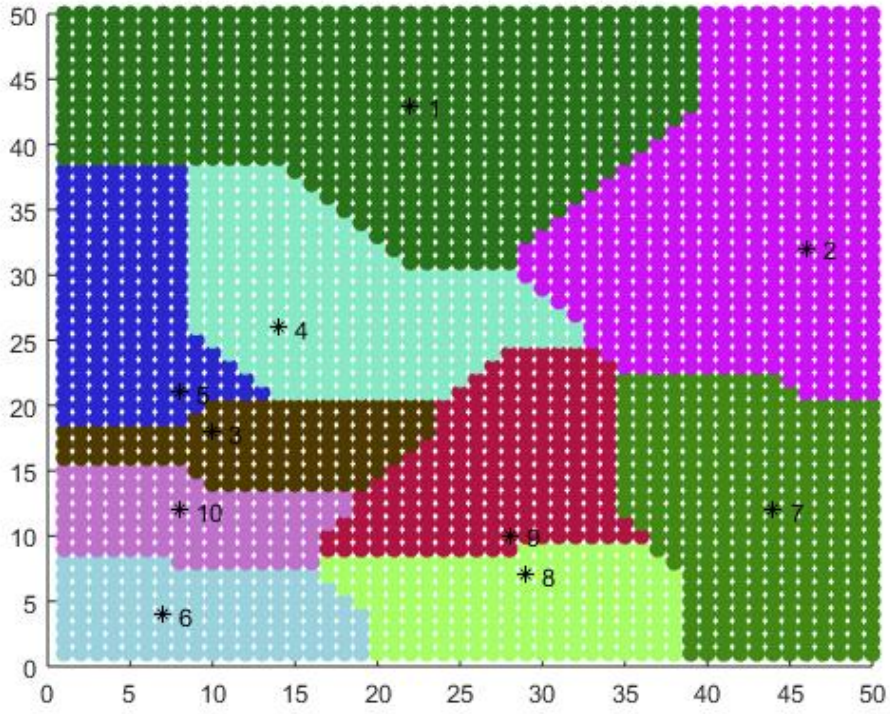

Figure 28: 50*50 regular mesh with 10 division cell

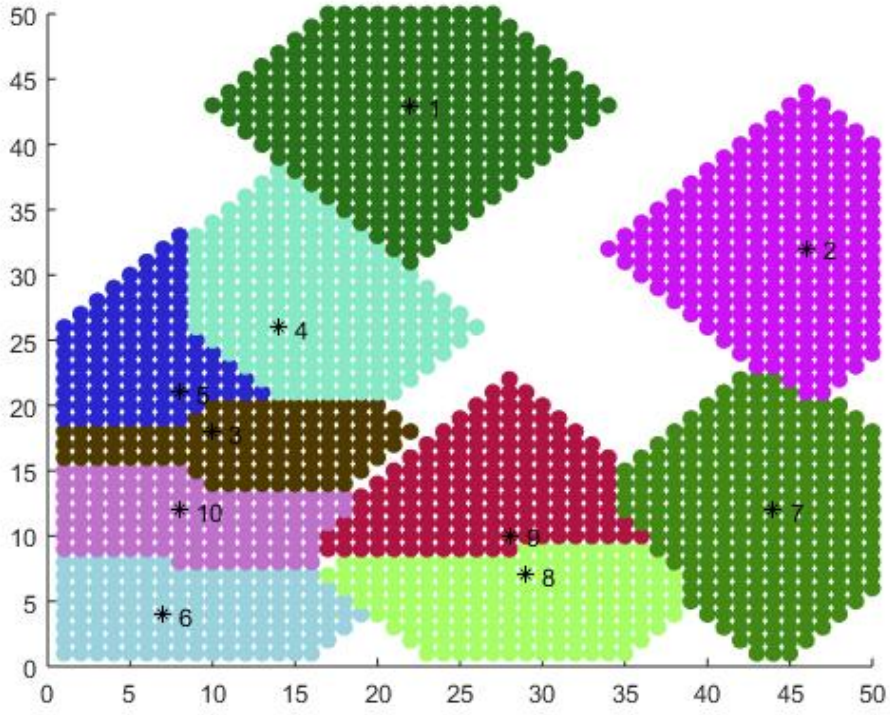

Figure 29: 50*50 regular mesh with 10 division cell and save over 30% unit core

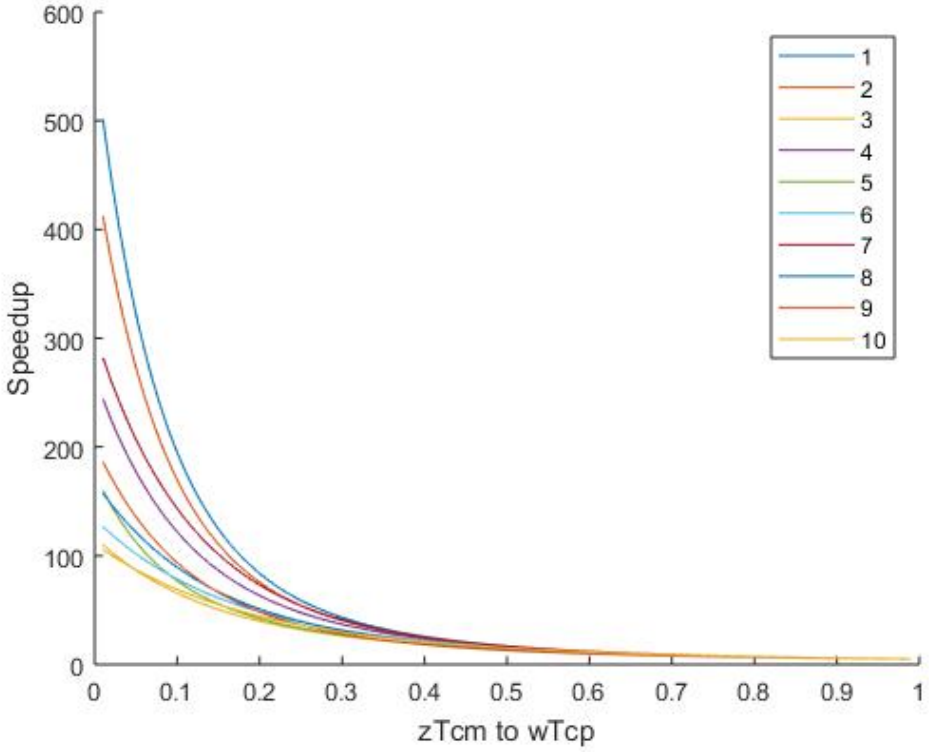

Figure 30: 50*50 regular mesh with 10 division cell

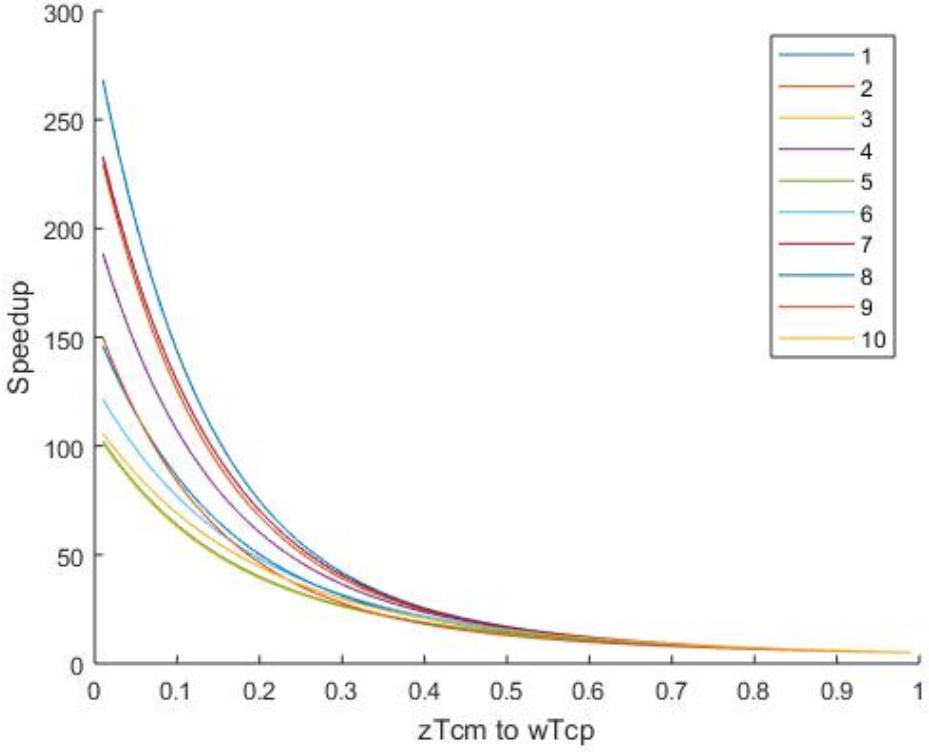

Figure 31: 50*50 regular mesh with 10 division cell

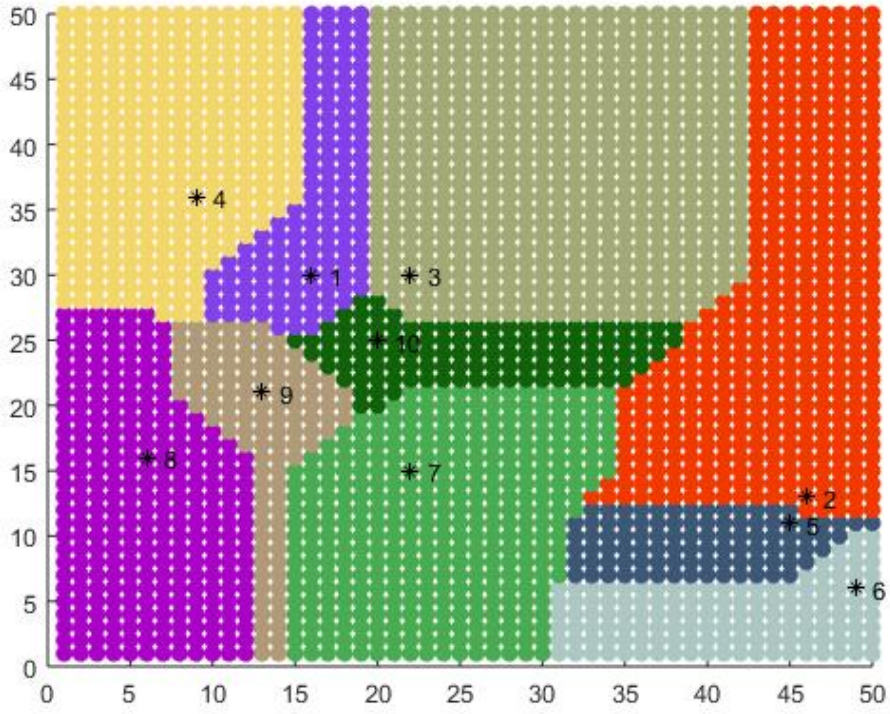

Figure 32: 50*50 regular mesh with 10 data injection community division

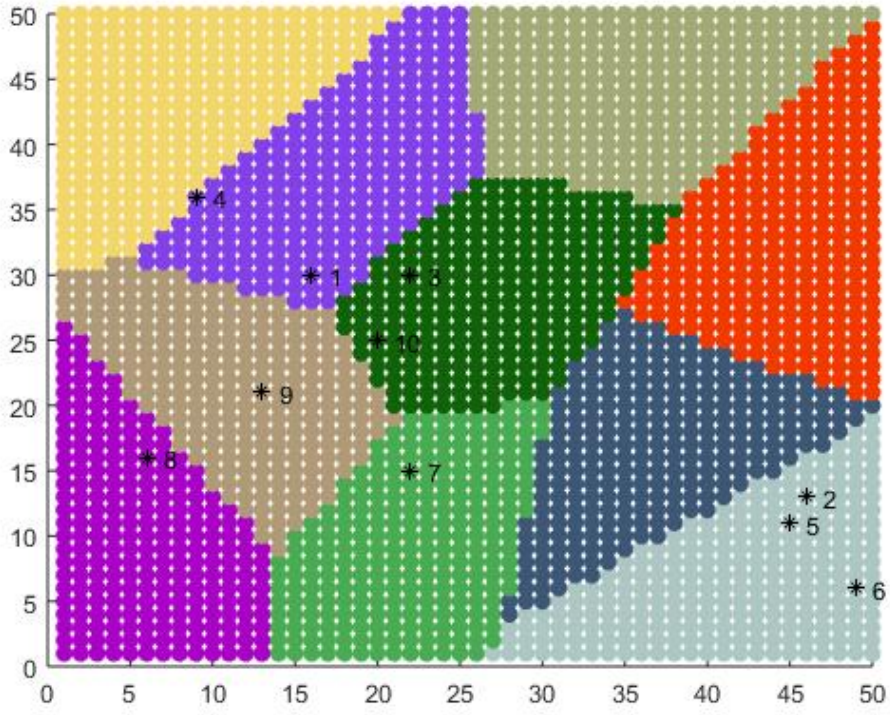

Figure 33: 50*50 regular mesh with 10 data injection community division

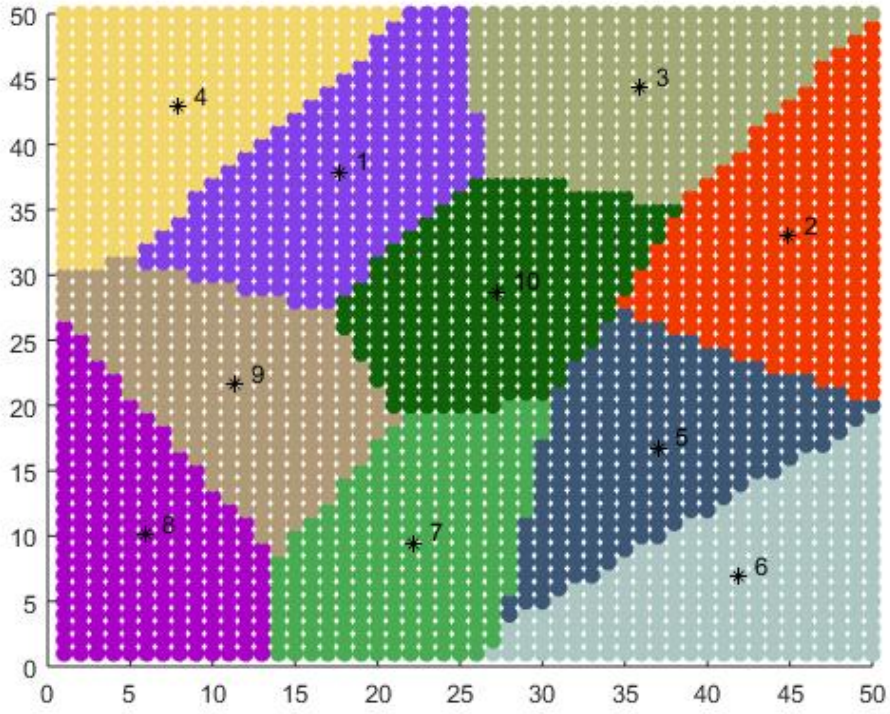

Figure 34: 50*50 regular mesh with 10 data injection community division

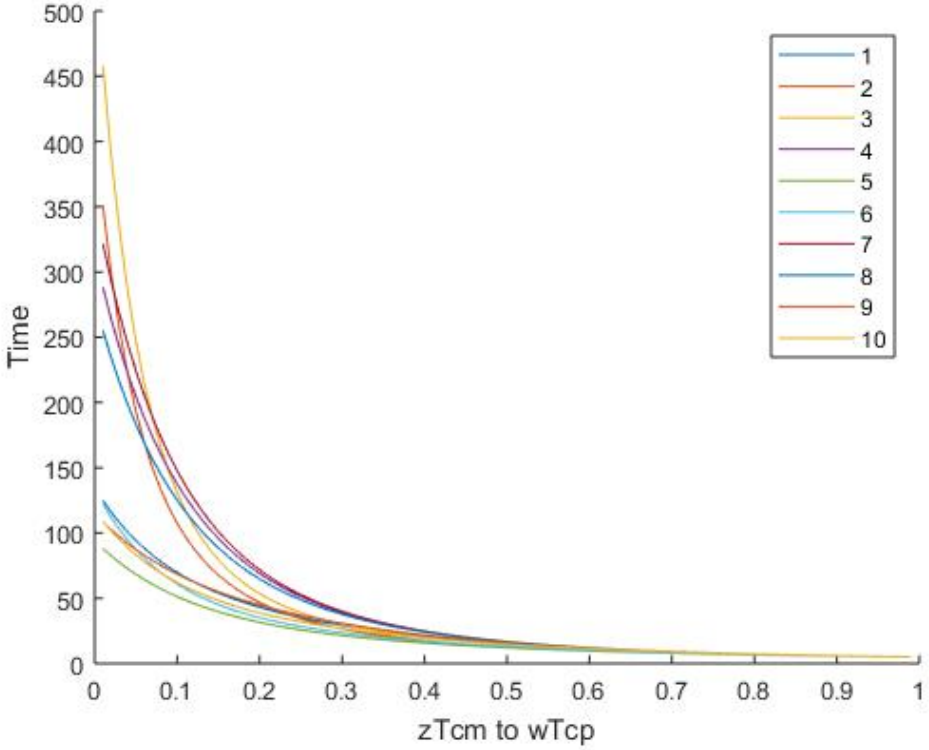

Figure 35: Speedup vs σ in different community division

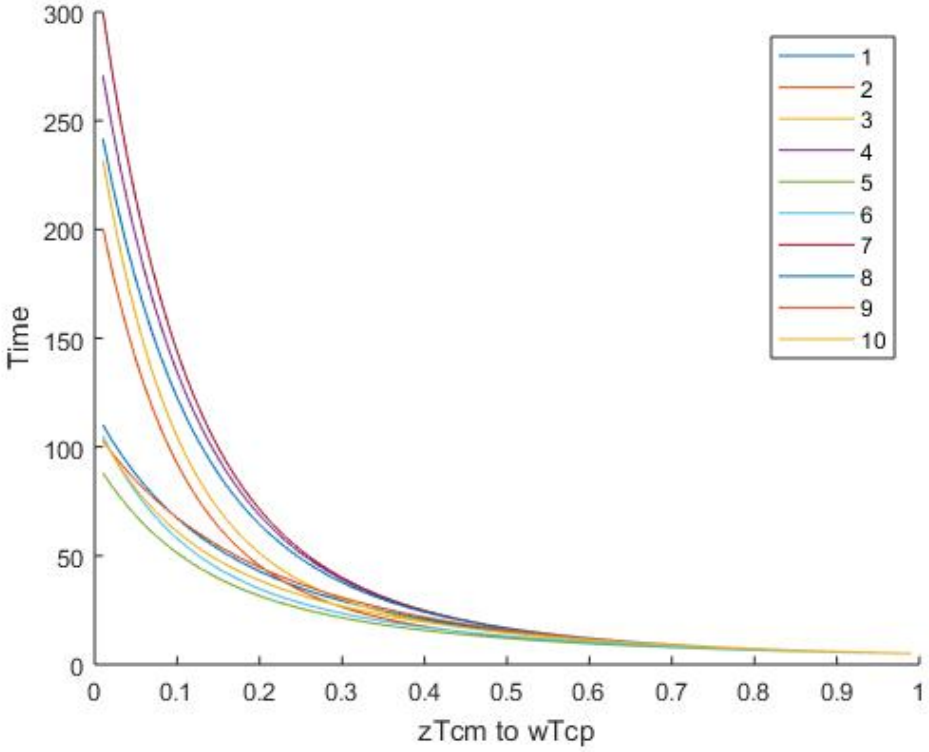

Figure 36: Speedup vs σ in different community division

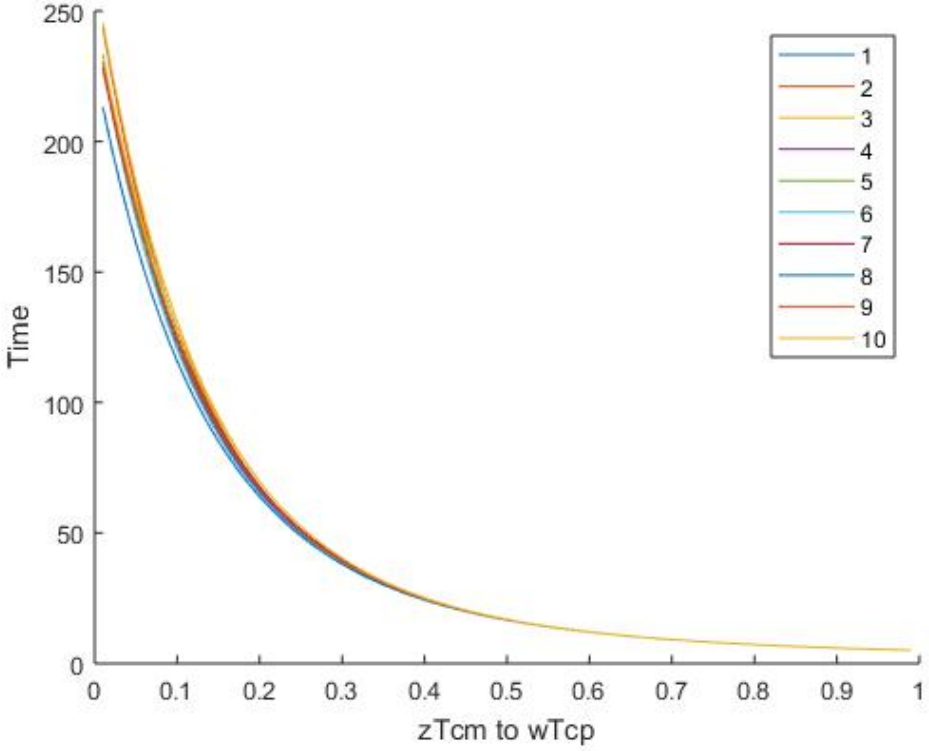

Figure 37: Speedup vs σ in different community division

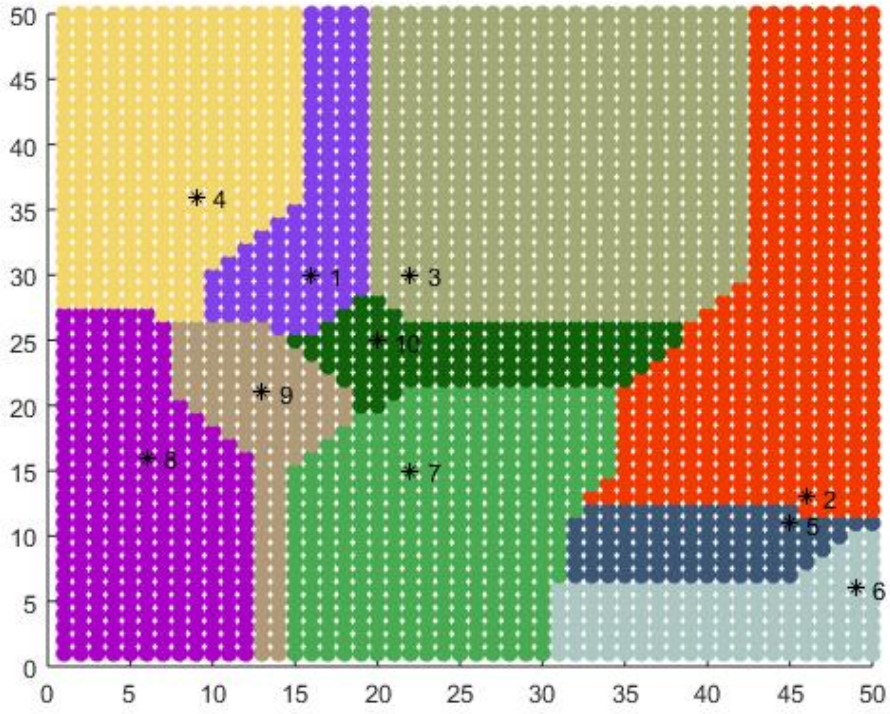

Figure 38: 50*50 regular mesh with 10 data injection community division

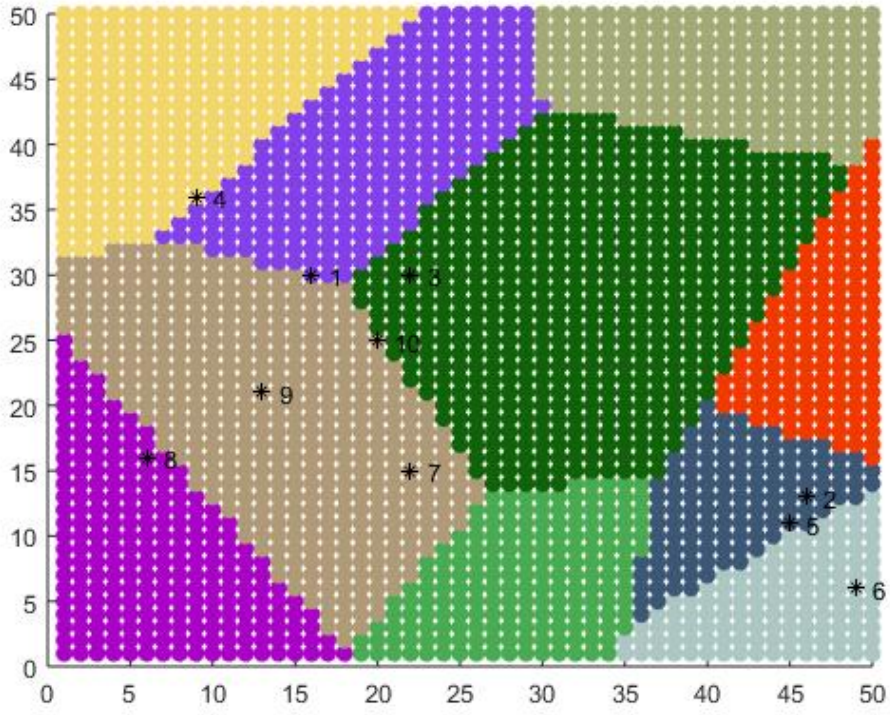

Figure 39: 50*50 regular mesh with 10 data injection community division

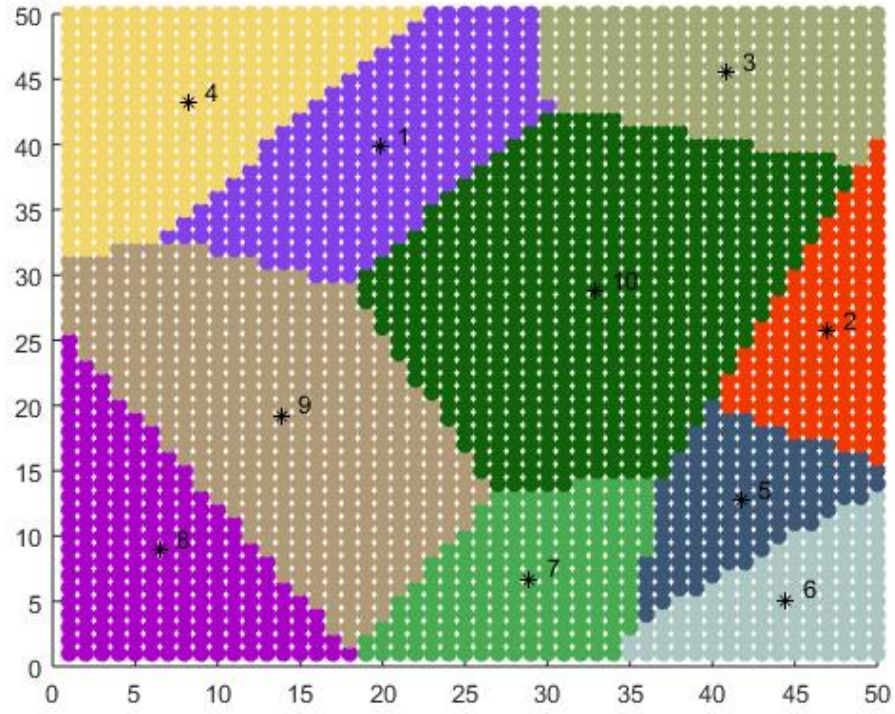

Figure 40: 50*50 regular mesh with 10 data injection community division

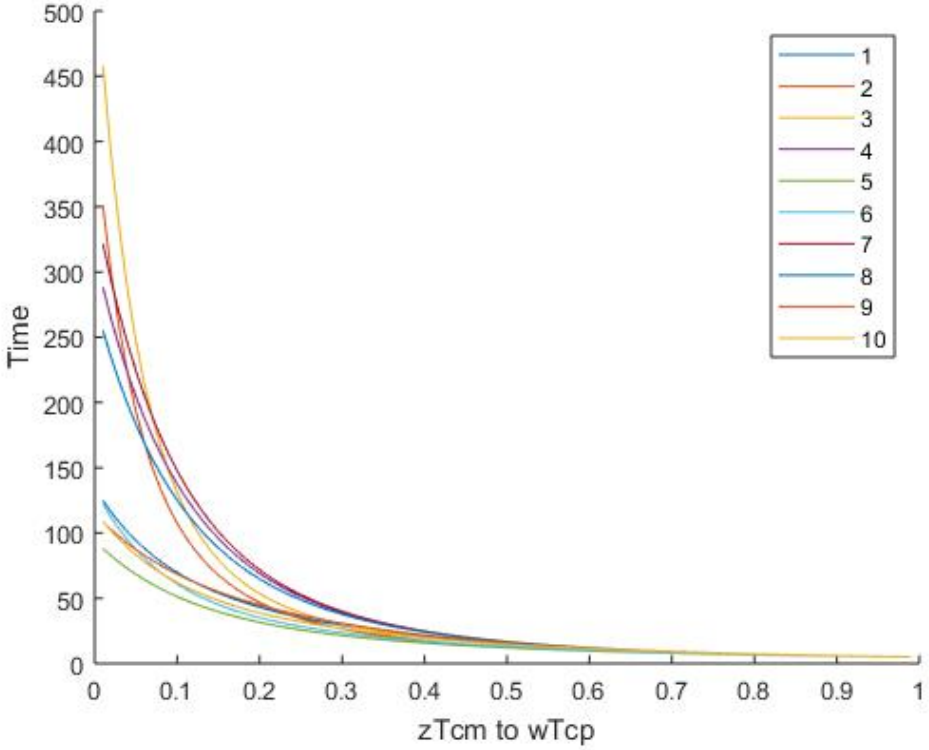

Figure 41: Speedup vs σ in different community division

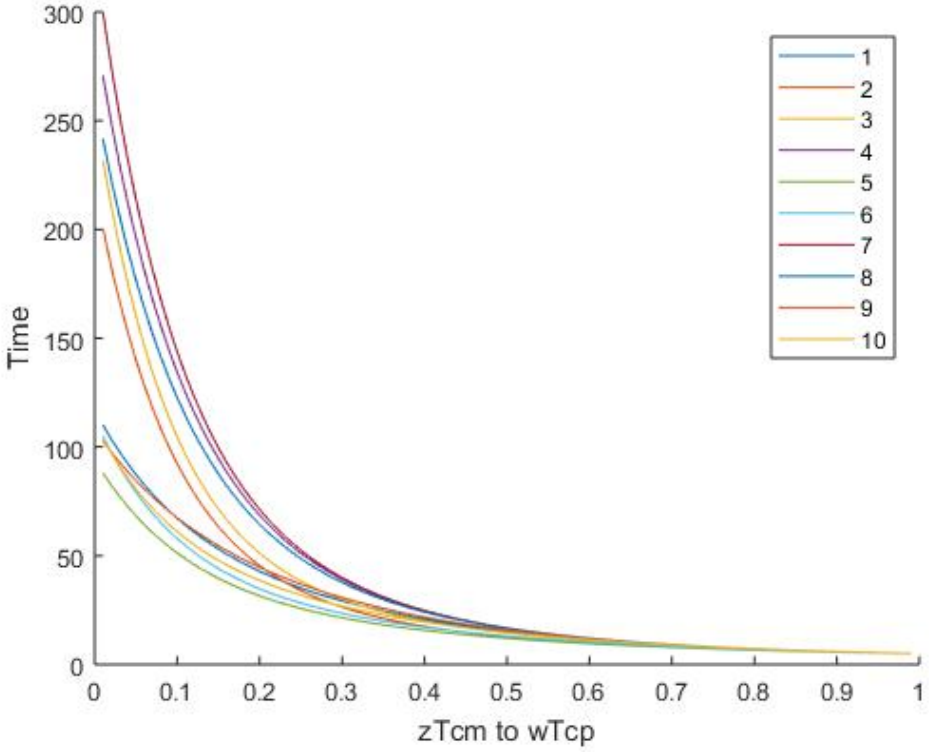

Figure 42: Speedup vs σ in different community division

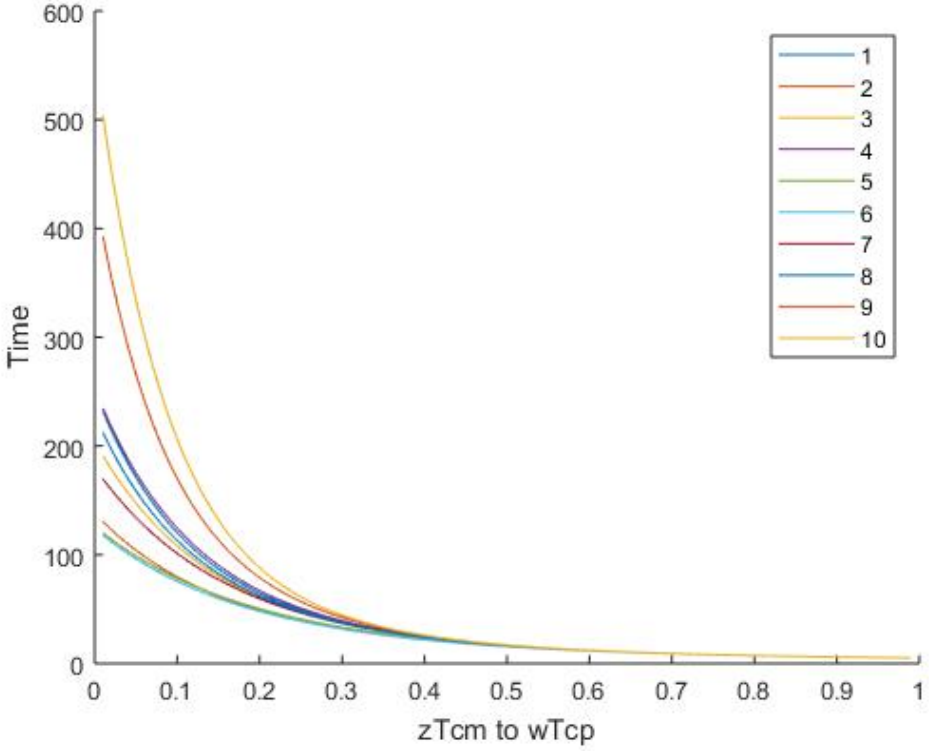

Figure 43: Speedup vs σ in different community division