Scheduling Divisible Workloads from Multiple Sources in Network-on-chip Plan

Junwei Zhang

Stony Brook, New York

Abstract

This paper is about multiple sources workloads scheduling in Network-on-chip.

Keywords: Divisible Load Theory, Processor equivalence, Voronoi Diagram, Optimal Mass Transport, Network-on-chip, Monte Carlo Method, Manhattan Distance

1. Introduction

Processor Equivalence[1] [2]
There are some properties about the NOC.

- Each unit core has 4 ports.
- Each unit core's computation ability and communication ability is the same.

There has some variable situation need to be considered.

1. Each unit core has front-end or not.
2. The number of sources.
3. The source position,for example, corner Fig.1 Fig.2 Fig.3 Fig.4 Fig.5 ,edge or inner grid position.
4. The load fraction of each sources. They are even or not.
5. Computation and Communication schema
    - Simultaneously computing after receiving the first bit.
    - Computing after receiving the whole fraction.

2. Processor Equivalence

2.1. 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 $m = 4, n = 10$ Fig.5
- Sensitivity Analysis

For a heterogeneous 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 $\omega_0 T_{cp}$.

2.1.1. Workstation with front-end

According to Fig.1, the left corner is the data load injection position. There are four core to handle the whole workload.

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

According to Fig.2, the left corner is the data load injection position. There are four core to handle the whole workload.

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

$$\sigma = \frac{z T_{cm}}{\omega T_{cp}}$$
$$\begin{bmatrix} 1 & 2 & 2 & 1 \\ 1 & -1 & 0 & 0 \\ 0 & \sigma - 1 & 1 & 0 \\ 0 & \sigma - 1 & \sigma & 1 \end{bmatrix} \times \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_3 \\ \alpha_5 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{2}$$

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

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

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

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

So the Speedup is

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

.

2.1.2. Workstation without front-end

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

$$\begin{bmatrix} 1 & 2 & 1 \\ 1 & -(\sigma + 1) & 0 \\ 1 & -\sigma & -(\sigma + 1) \end{bmatrix} \times \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \tag{4}$$

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

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

$$\begin{bmatrix} 1 & 2 & 2 & 1 \\ 1 & -(\sigma+1) & 0 & 0 \\ 1 & -\sigma & -(\sigma+1) & 0 \\ 1 & -\sigma & -\sigma & -(\sigma+1) \end{bmatrix} \times \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_3 \\ \alpha_5 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{5}$$

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

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

$$\begin{bmatrix} 1 & 2 & 3 & \cdots & 3 & 2 & 1 \\ 1 & -(\sigma+1) & 0 & \cdots & 0 & 0 & 0 \\ 1 & -\sigma & -(\sigma+1) & \cdots & 0 & 0 & 0 \\ 1 & -\sigma & -\sigma & -(\sigma+1) & 0 & \cdots & 0 \\ 1 & -\sigma & -\sigma & -\sigma & -(\sigma+1) & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & \\ 1 & -\sigma & -\sigma & \cdots & -\sigma & -\sigma & -(\sigma+1) \end{bmatrix} \times \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_3 \\ \alpha_5 \\ \vdots \\ \alpha_{2\times n-1} \\ \alpha_{2\times n+1} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{6}$$

So the Speedup is

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

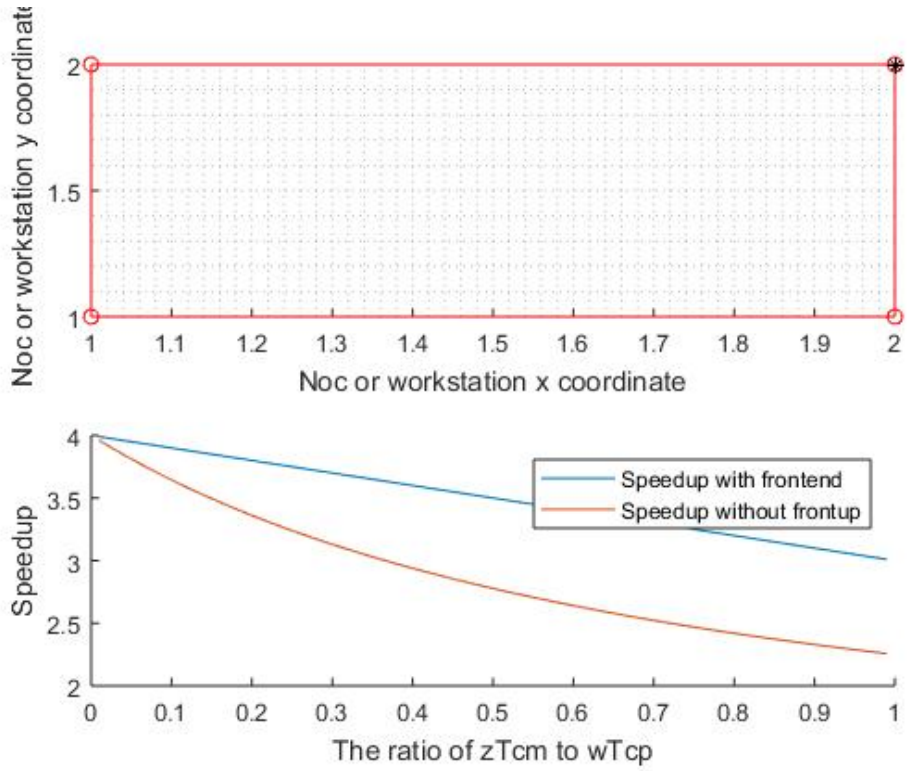

Figure 1: 2*2 regular mesh

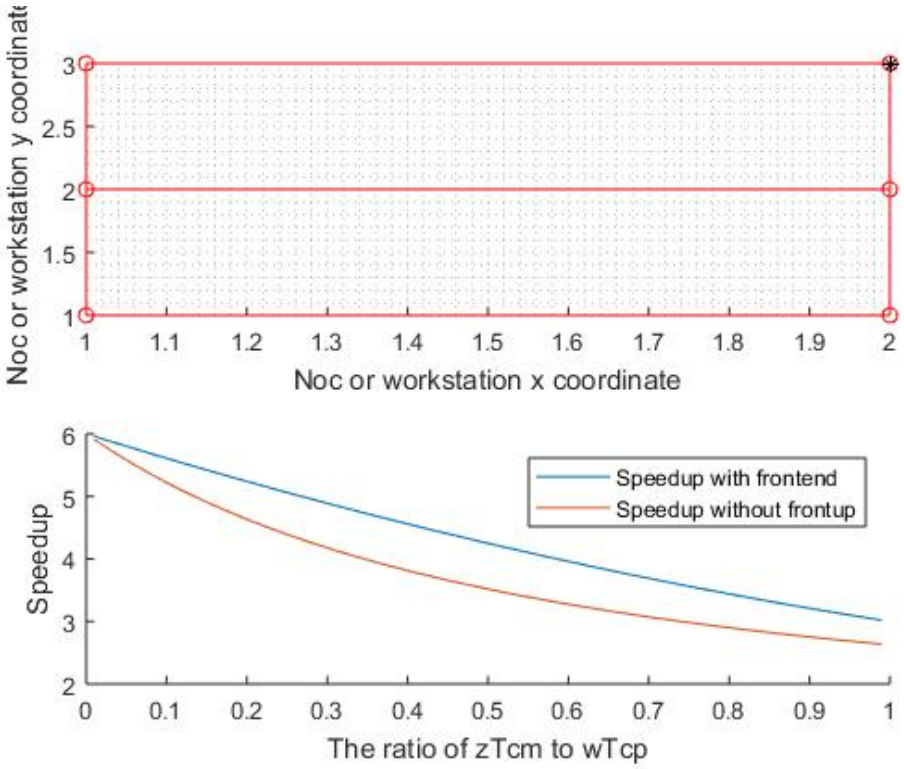

Figure 2: 2*3 regular mesh

2.2. Processor Equivalence Formula

- Simultaneously computing after receiving the first bit.
- Computing after receiving the whole fraction.

2.3. Load From Edge

- Load from edge grid point. Fig.6 Fig.7

2.3.1. With front end

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

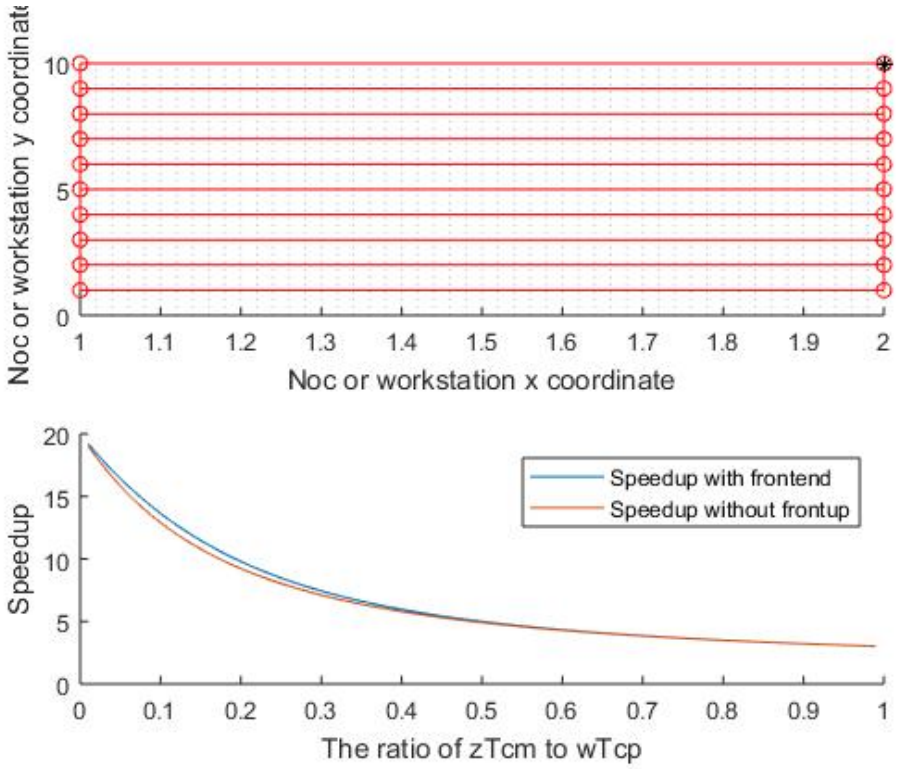

Figure 3: 2*10 regular mesh

$$\alpha_1 zT_{cm} + \alpha_4 \omega T_{cp} = T_{f,m}$$
$$\alpha_1 zT_{cm} + \alpha_5 \omega T_{cp} = T_{f,m}$$
$$\alpha_1 zT_{cm} + \alpha_6 \omega T_{cp} = T_{f,m}$$

$$(\alpha_1 + \alpha_4) zT_{cm} + \alpha_7 \omega T_{cp} = T_{f,m}$$
$$(\alpha_1 + \alpha_4) zT_{cm} + \alpha_8 \omega T_{cp} = T_{f,m}$$

$$\begin{bmatrix} 1 & 3 & 3 & 2 \\ 1 & -1 & 0 & 0 \\ 0 & \sigma - 1 & 1 & 0 \\ 0 & \sigma - 1 & \sigma & 1 \end{bmatrix} \times \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_4 \\ \alpha_7 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{7}$$

For example, there are β_i workstation on level i.

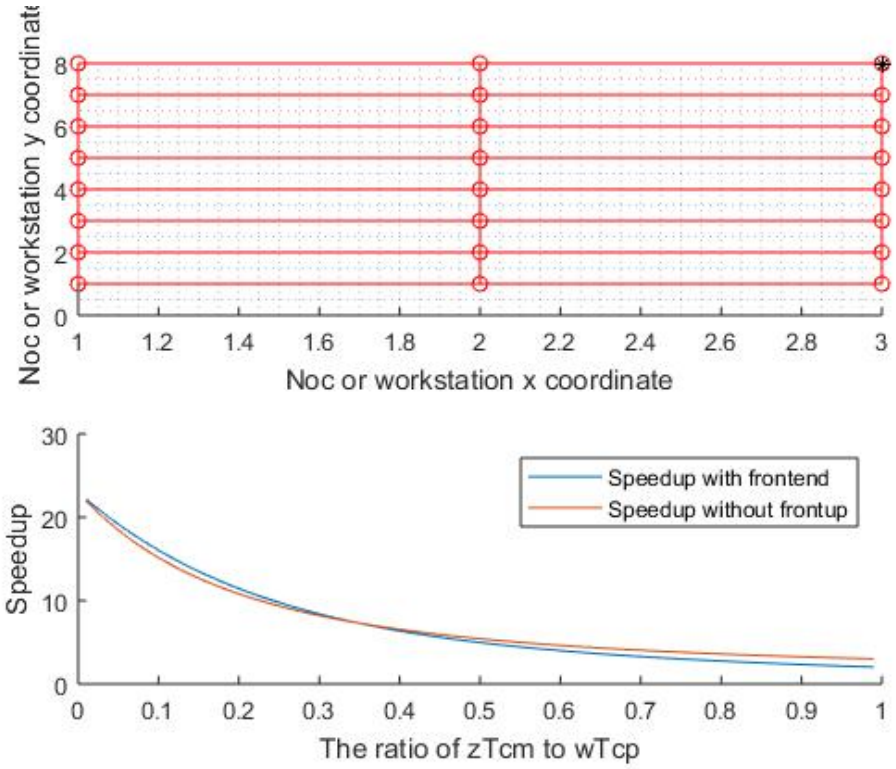

Figure 4: 3*8 regular mesh

2.3.2. Without front end

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

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

$$(\alpha_1 + \alpha_4 + \alpha_7) z T_{cm} + \alpha_7 \omega T_{cp} = T_{f,m}$$

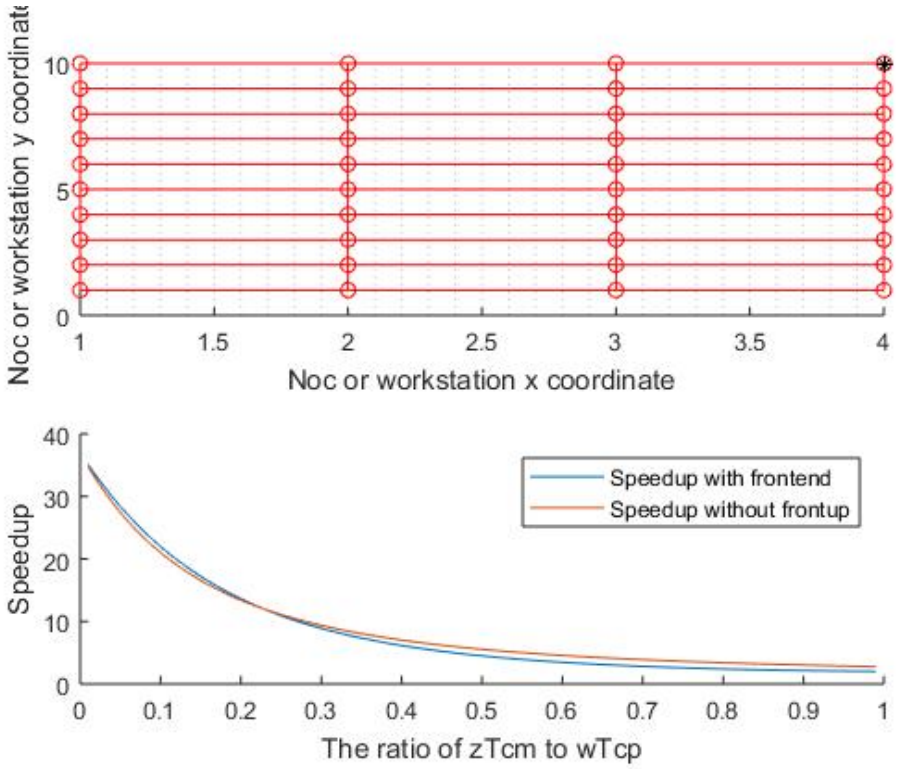

Figure 5: 4*10 regular mesh

$$(\alpha_1 + \alpha_4 + \alpha_7) zT_{cm} + \alpha_8 \omega T_{cp} = T_{f,m}$$

$$\begin{bmatrix} 1 & 3 & 3 & 2 \\ 1 & -(\sigma+1) & 0 & 0 \\ 1 & -\sigma & -(\sigma+1) & 0 \\ 1 & -\sigma & -\sigma & -(\sigma+1) \end{bmatrix} \times \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_4 \\ \alpha_7 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{8}$$

2.4. Load From Inner Grid Point

- Load from inner grid position.

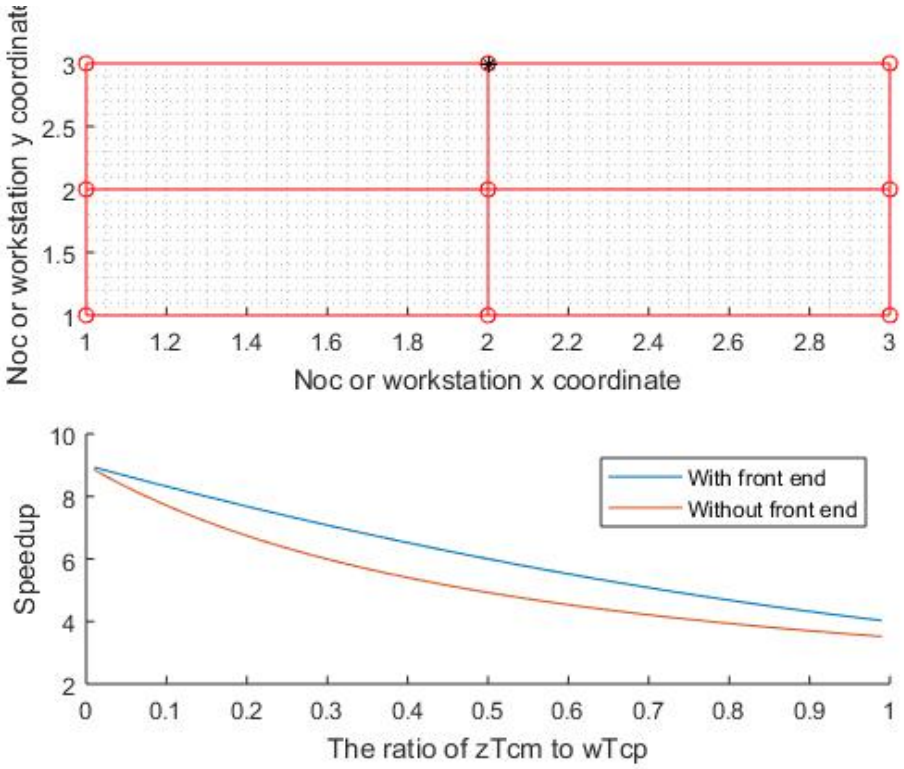

Figure 6: 3*3 regular mesh

2.5. General Case

2.5.1. With front end

$$\begin{bmatrix} 1 & m_1 & m_2 & \cdots & m_{n-2} & m_{n-1} & m_n \\ 1 & -1 & 0 & \cdots & 0 & 0 & 0 \\ 0 & \sigma-1 & 1 & \cdots & 0 & 0 & 0 \\ 0 & \sigma-1 & \sigma & 1 & 0 & \cdots & 0 \\ 0 & \sigma-1 & \sigma & \sigma & 1 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & \\ 0 & \sigma-1 & \sigma & \cdots & \sigma & \sigma & 1 \end{bmatrix} \times \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \vdots \\ \alpha_{n-1} \\ \alpha_n \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{9}$$

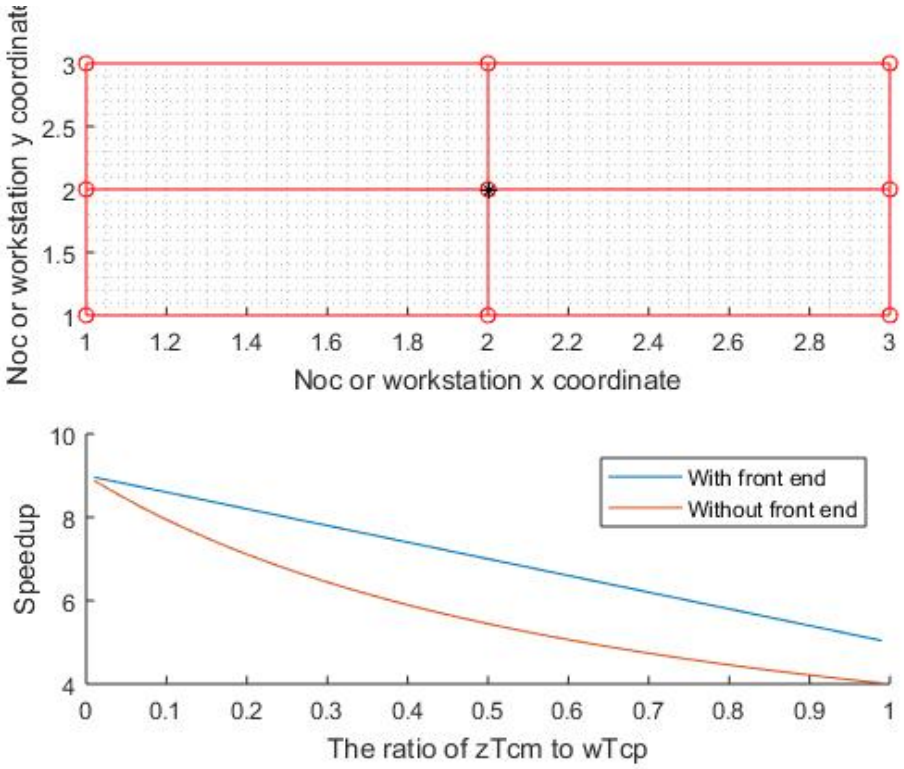

Figure 7: 3*3 regular mesh

2.5.2. Without front end

$$\begin{bmatrix} 1 & m_1 & m_2 & \cdots & m_{n-2} & m_{n-1} & m_n \\ 1 & -(\sigma+1) & 0 & \cdots & 0 & 0 & 0 \\ 1 & -\sigma & -(\sigma+1) & \cdots & 0 & 0 & 0 \\ 1 & -\sigma & -\sigma & -(\sigma+1) & 0 & \cdots & 0 \\ 1 & -\sigma & -\sigma & -\sigma & -(\sigma+1) & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & \\ 1 & -\sigma & -\sigma & \cdots & -\sigma & -\sigma & -(\sigma+1) \end{bmatrix} \times \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \vdots \\ \alpha_{n-1} \\ \alpha_n \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{10}$$

3. Calculate the source subgraph

- the sources consist of a whole larger source node.

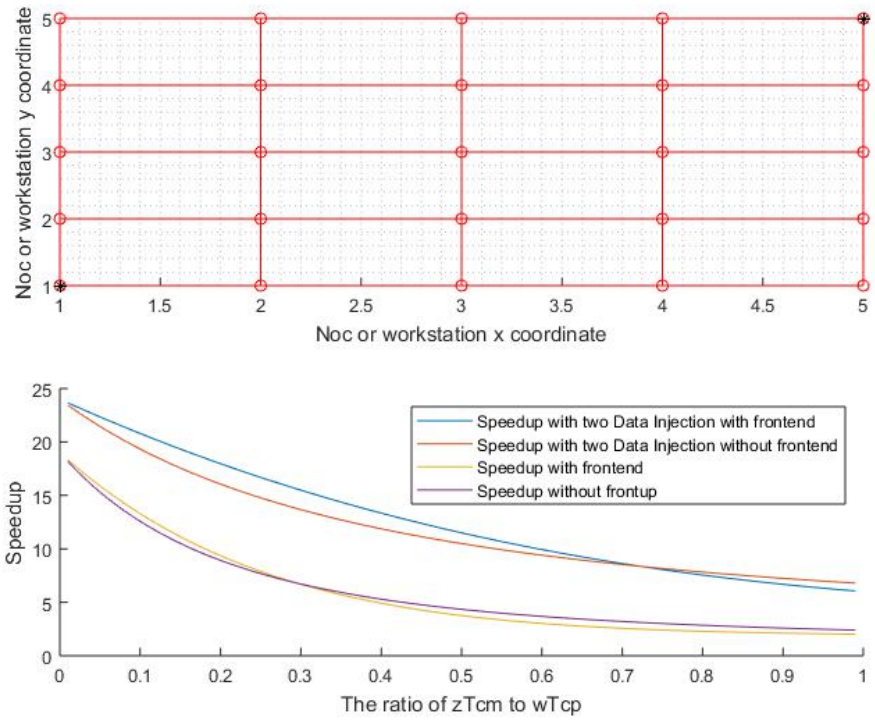

Figure 8: Two data injection 5*5 regular mesh

- the load fraction are even.We can calculate the Manhattan Voronoi diagram directly under the definition of Manhattan distance.
- the source load fraction are not even.
  - Calculate the Manhattan Voronoi Diagram.
  - Calculate the processor equivalence ability.
  - If the ability of each subgraph equals to the corresponding load fraction, then stop.
  - Otherwise, merge the furthest level leaves to other source node voronoi area.(I can explain with you on Tuesday discussion.)

4. Optimal Mass Transport

- If we know the number of source

- If we know the load fraction of each source
- To decide the location of each base.

The main assumption is here.

- All the unit core are the same.
- The number of core is approximate with the ability with the power of corresponding subgraph.
- The number of core is approximate with the area of each subgraph.

So the problem transfer a work load fraction to a problem which is divide the whole graph to a target area(measure).

If the target workload are equal, we also can choose the Centroidal Voronoi tessellation.

5. Experiment

- PDE simulation
- Superposition
- Processor Equivalence simulation.

6. Conclusion

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

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