

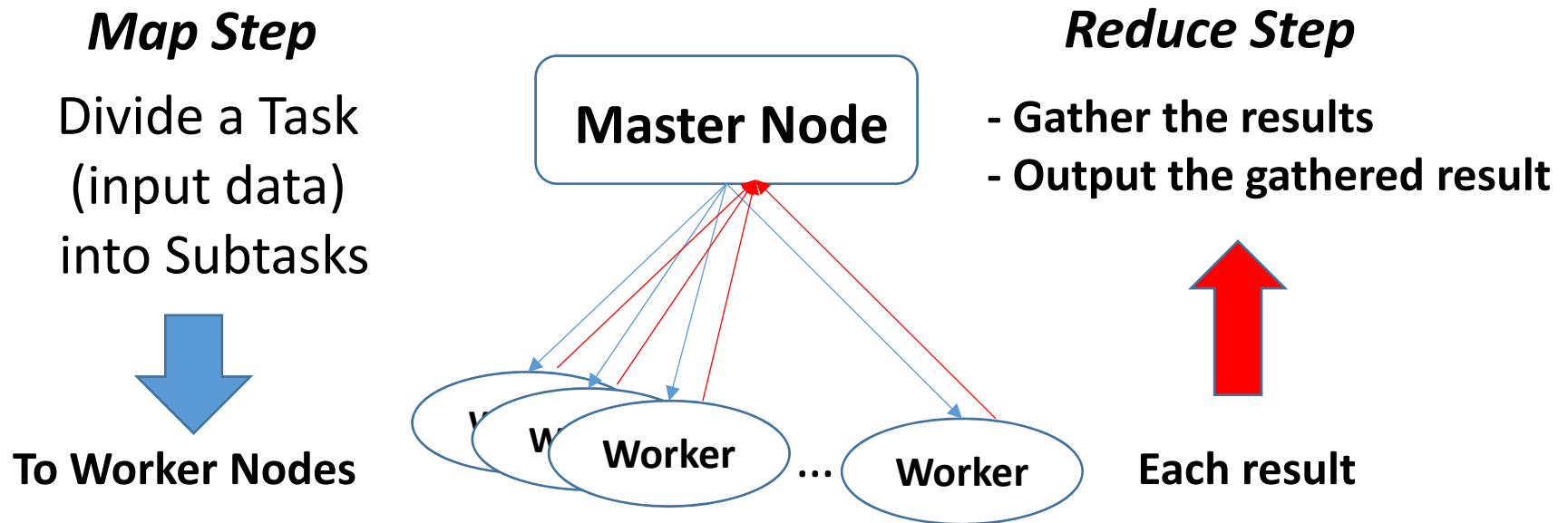
Yet Another Parallel Hypothesis Search for Inverse Entailment

Hiroyuki Nishiyama and Hayato Ohwada

Faculty of Sci. and Tech.,
Tokyo University of Science

Background

Example of Distributed Computing Model: **MapReduce**



However...

MapReduce is effective when each subtask is full independent



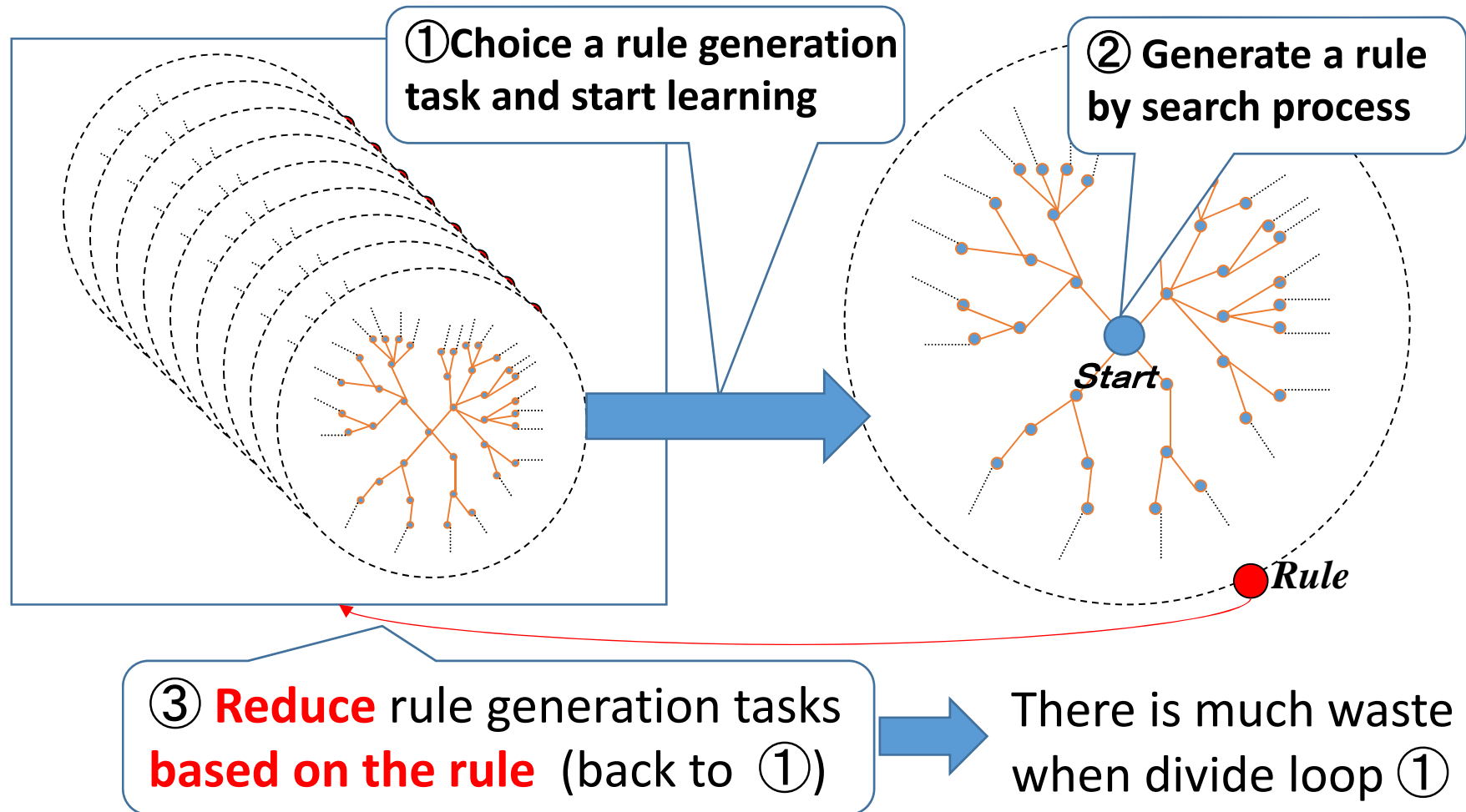
Not suitable for Inductive Logic Programming

Flow of ILP (Inductive Logic Programming)

ILP has 2 repetition processing loops in learning

Many rule generation tasks

Executing a rule generation



Our Purpose

Speed up ILP tool by parallel processing

We divide the search process for generate a rule



Design and Implement a Parallel Hypothesis Search System

- Divide the search area using Worker Nodes (CPUs)
- The Worker Node helps other Worker Nodes when finish own search task



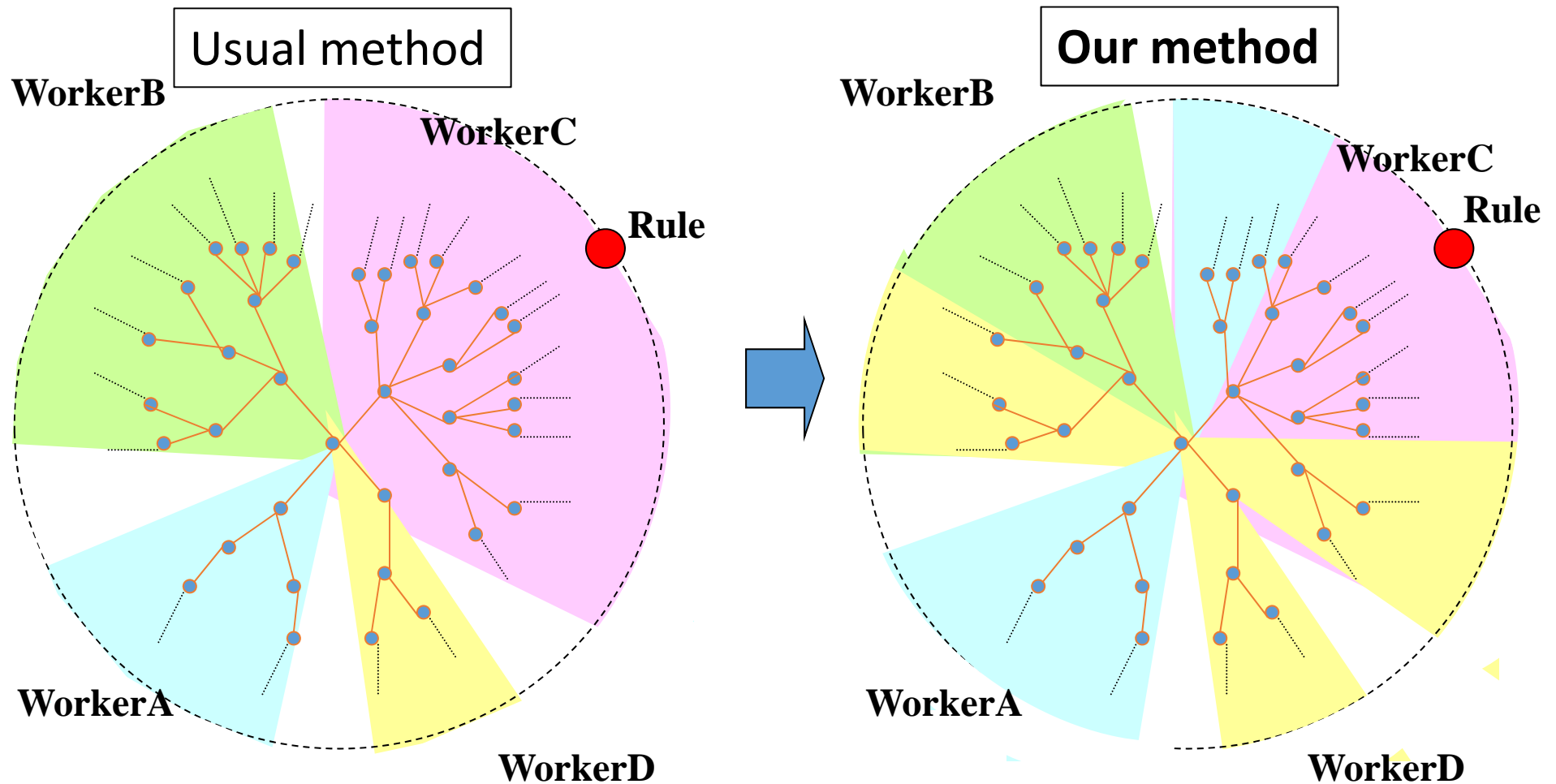
Realize effective parallel processing

Our Approach

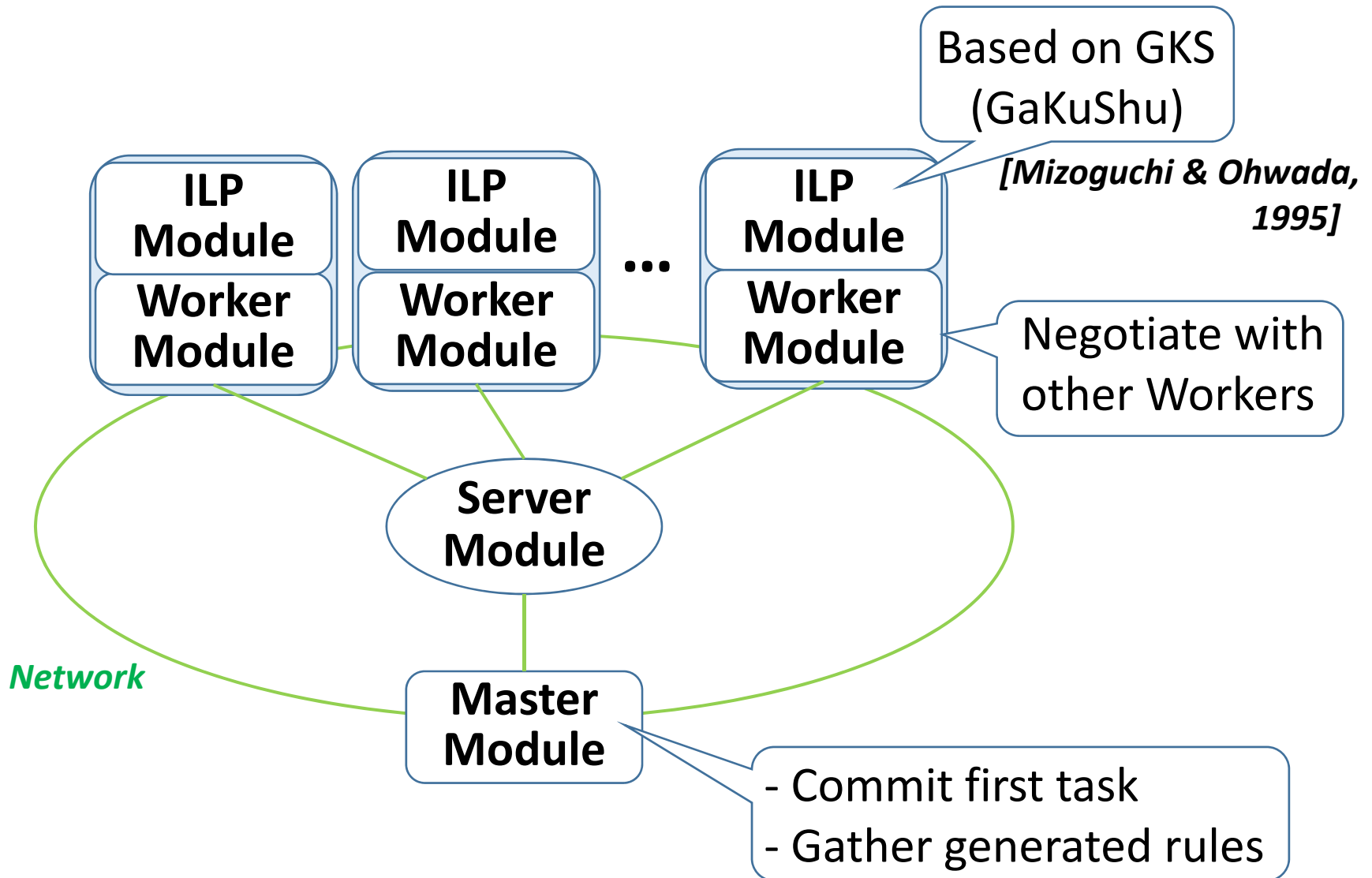
Usual method only assigns subtasks to workers



Each worker communicates with each other and requests (or accepts) a part of the subtask



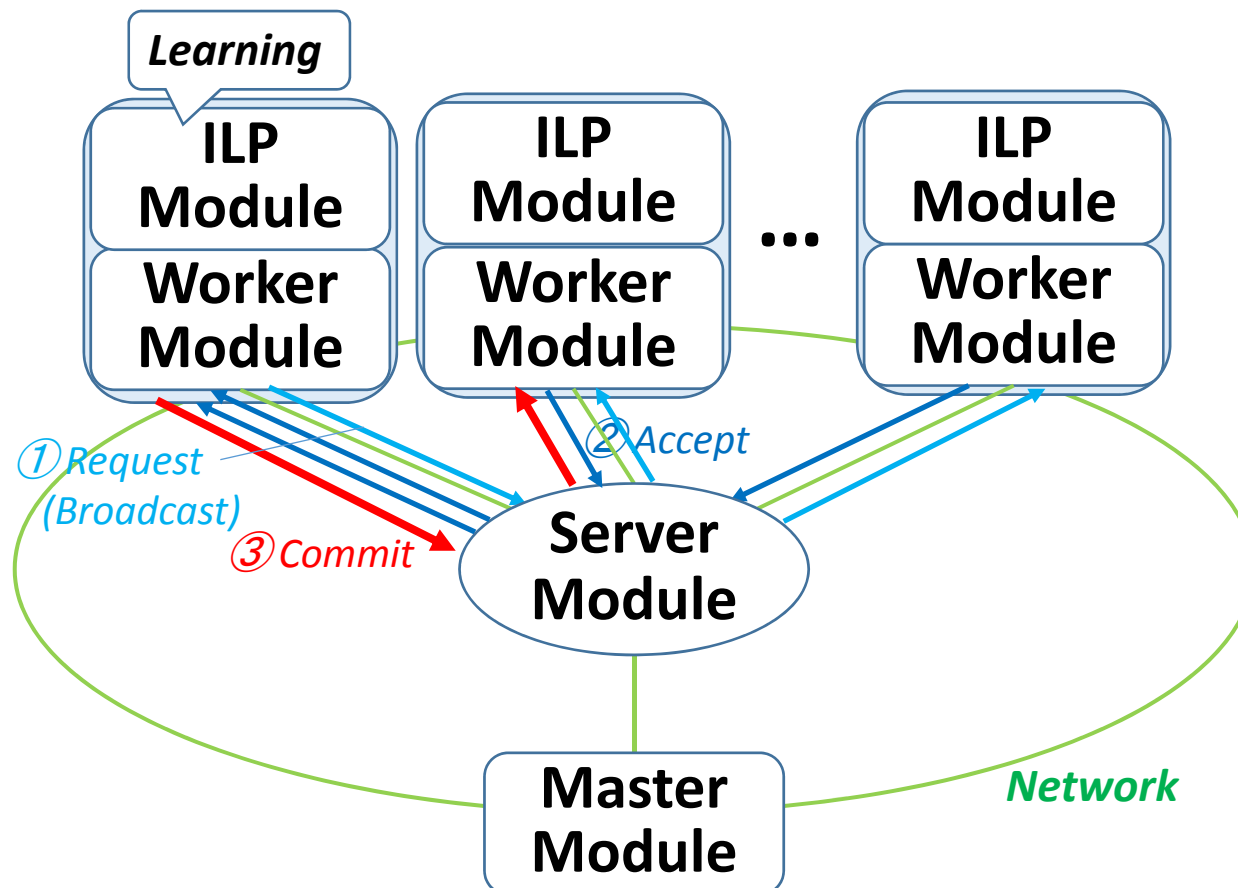
System Configuration



(All modules were implemented by Java language)

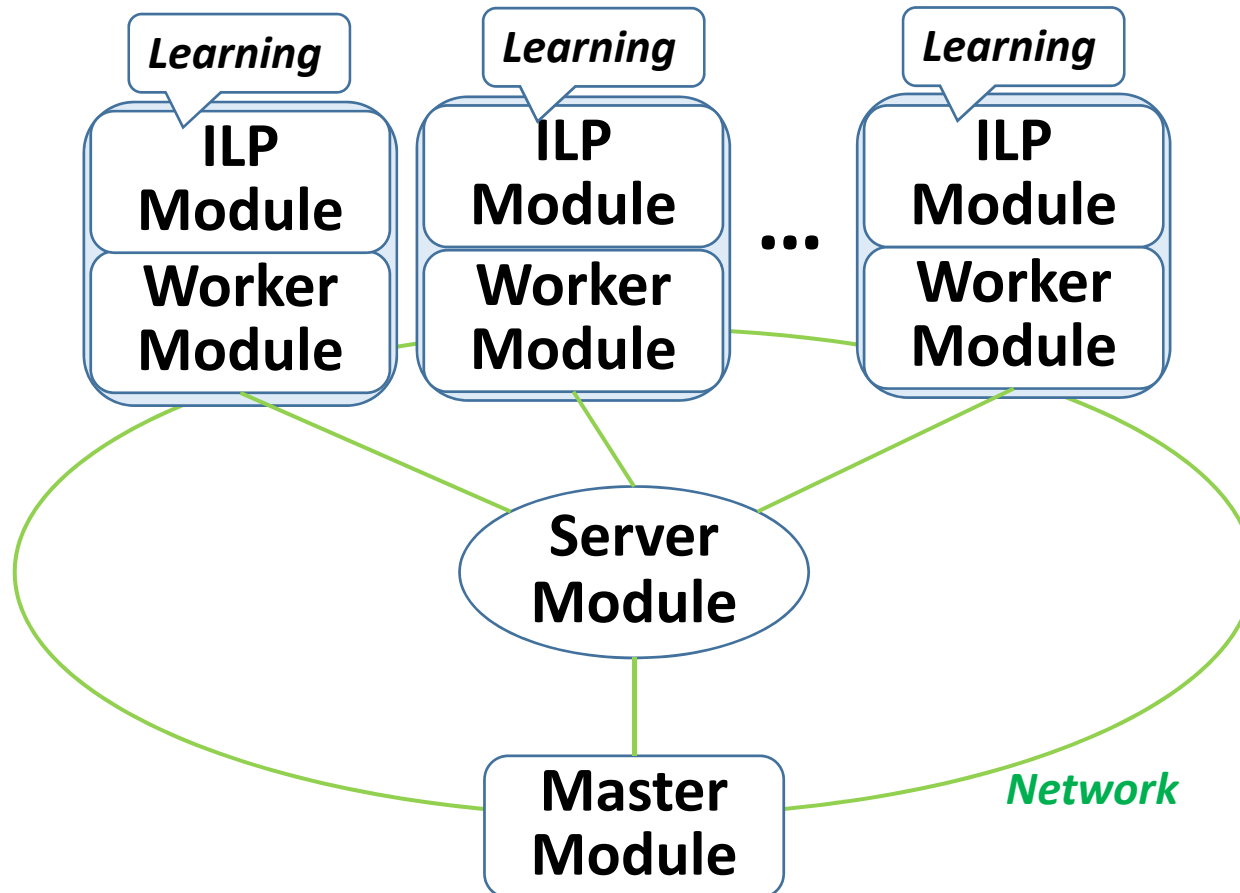
Flow of Negotiation to Divide a Task

- ① Learning worker **requests** a part of task to other all workers
- ② Non-learning worker **accepts** the requested task
- ③ Requesting worker **commits** to an accepting worker



State that all Workers are Assigned (Saturation of the Task)

- All workers sent a request message to other all workers
- All workers received request messages from other all workers

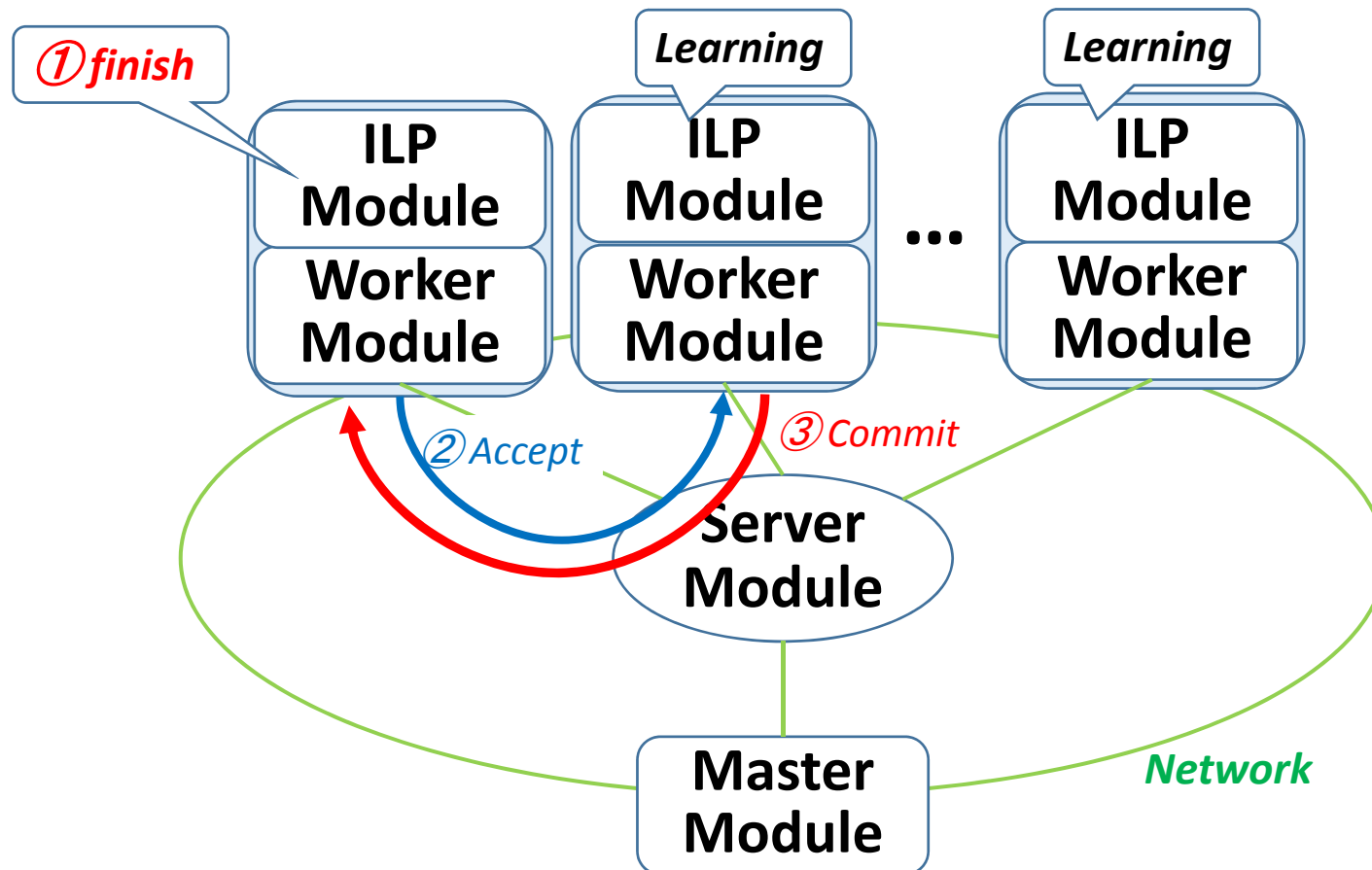


State that all Workers are Assigned (Saturation of the Task)

When one worker's learning task is finished



The worker accepts and is committed



Experiment

Environment

2 PC (total 12 CPUs)

(Intel(R) Core(TM) i7-5820K CPU @ 3.30GHz 16.0GB 64bit)

Data

- Drug Design for Human (Sample Problem)

From “In Silico Screening of Zinc (II) Enzyme Inhibitors Using ILP”

[Ito, Ohwada et al., ILP2015] (Yesterday’s presentation)

Using 1 CPU: 724 sec.

- Drug Design for Plant (Large Scale Problem)

From “Extracting the Common Structure of Compounds to Induce Plant Immunity Activation using ILP”

[Matsumoto, Ohwada et al., ILP2015] (Yesterday’s presentation)

Using 1 CPU: 56,372 sec. (15.66hours)

Main Display of Parallel Execution (12 workers)

The image displays a grid of 12 windows, each representing a SubAgent (numbered 1 to 12). Each window contains a log of task execution and a control panel at the bottom. The logs show the following sequence of events for each agent:

- 1: agentIDをサーバから受け取りました ID: 1
- 1: requestメッセージ受信: taskID: 1, from agentID: 0, subTaskID: 0
- 1: acceptメッセージを送信: accept 0 1 1 0
- 1: commitメッセージ受信: taskID: 1, from agentID: 0
- 1: Commit成立: タスクを受け入れて処理開始
- 1: 依頼元: 0, タスクID: 1, subTaskID: 0
- 1: タスクの受信開始
- 1: タスクの受信完了
- 1: 新しいTask, 古いTaskの実行

The control panel for each window includes:

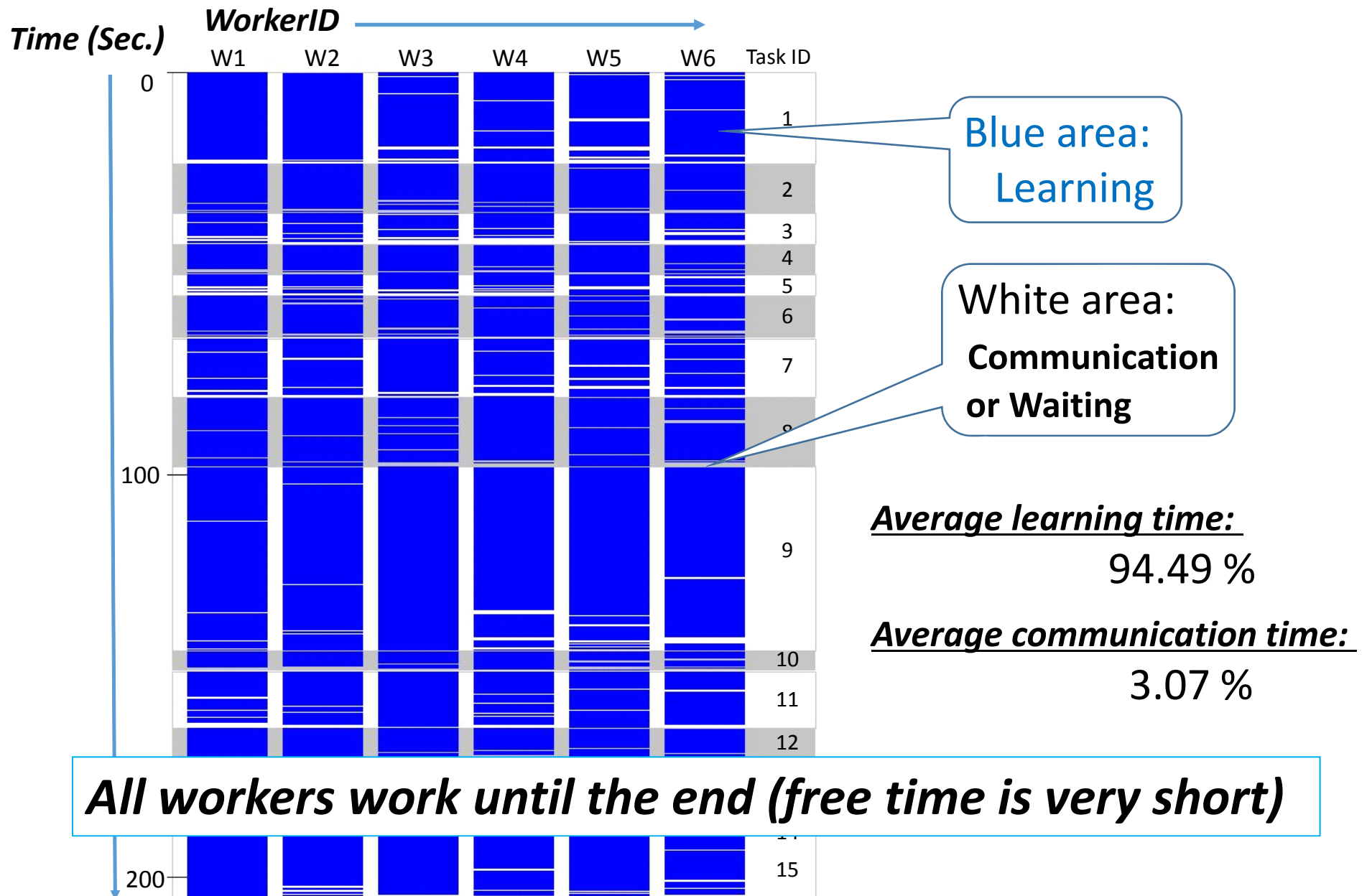
- Host: localhost
- Port: 28000
- Connect button
- Close button
- Quit button

The task details for each agent are as follows:

SubAgent	nowTask	requestTask	requestedNum
1	T:1,A:0,Sub:0	T:1,Sub:240	11
2	T:1,A:7,Sub:220	T:1,Sub:240	11
3	T:1,A:1,Sub:236	T:1,Sub:231	11
4	T:1,A:1,Sub:239	T:1,Sub:200	11
5	T:1,A:3,Sub:230	T:1,Sub:218	11
6	T:1,A:1,Sub:238	T:1,Sub:193	11
7	T:1,A:1,Sub:237	T:1,Sub:221	11
8	T:1,A:1,Sub:234	T:1,Sub:207	11
9	T:1,A:7,Sub:219	T:1,Sub:211	11
10	T:1,A:8,Sub:206	T:1,Sub:204	11
11	T:1,A:9,Sub:210	T:1,Sub:201	11
12	T:1,A:6,Sub:186	T:1,Sub:196	11

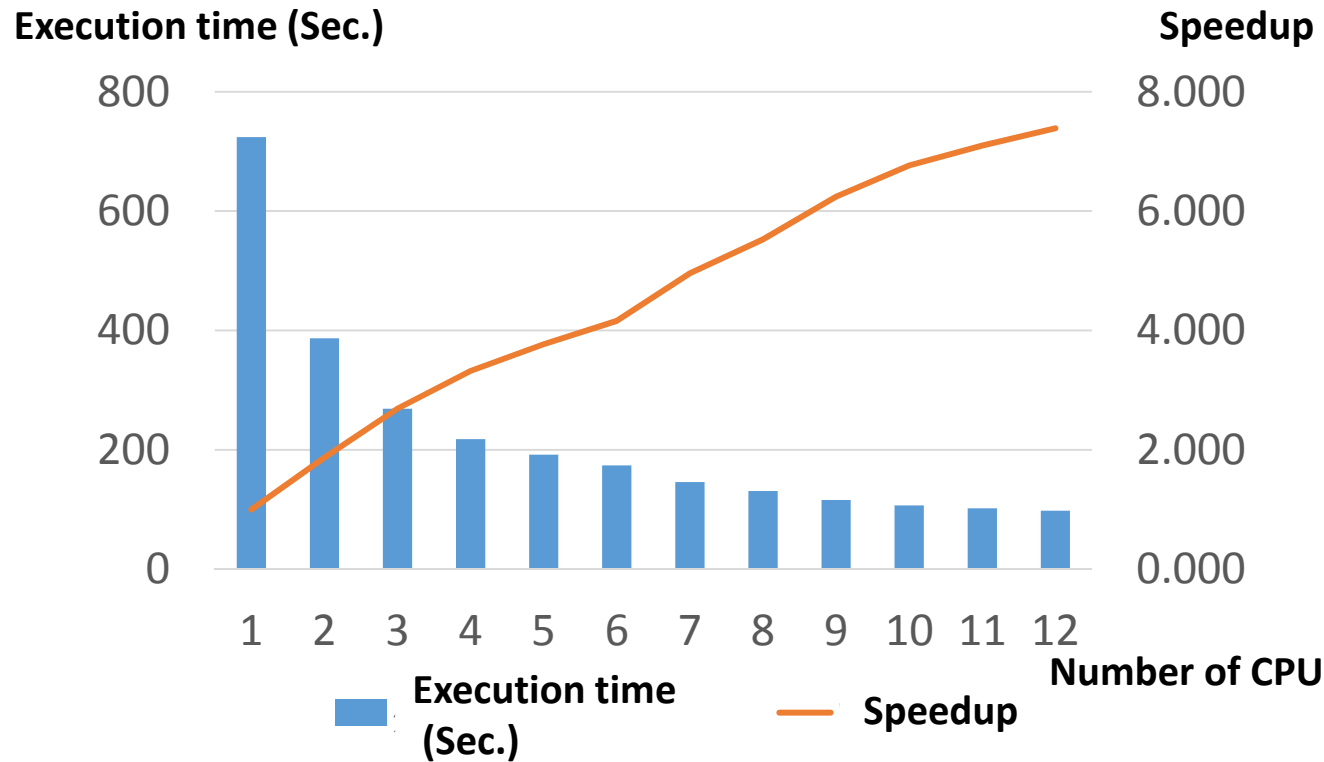
The Windows taskbar at the bottom shows the system clock as 20:45 on 2014/12/10.

Operational status of each worker in the parallel-processing experiment using six workers (6CPUs) for the Sample Problem



Experimental Results

Sample Problem (Drug Design for human) (12CPU: 6 CPU × 2 PC)



Add PC

Large-scale problem (Drug Design for plant) (6CPU × 2 + 4CPU × 2)

Number of CPU	Execution time (sec.)	Speedup
1	56372	1.000
12	5723	9.850
20	3563	15.821

Conclusion

Designed and Implemented a Parallel Hypothesis Search System

- Divide the search area using Worker Nodes (CPUs)
- The Worker Node helps other Worker Nodes when finish own search task



All workers work until the end (free time is very short)



Realize effective parallel processing