Adaptive Grid Computation Approach in the Peer-to-Peer Grid Computing Systems

MaengSoon Baik, SungJin Choi, ChongSun Hwang, JunMin Gil and HeonChang Yoo

Dept. Computer Science & Engineering

Korea University and

KISTI

Presented by: SungJin Choi

ABSTRACT

1. The peer-to-peer Grid computing systems has been regarded as an attractive platform to support massively parallel applications based on peer-to-peer computing technology in Grid computing area.
2. As the number of volunteers is increased, the job management overhead of the central job management server is more and more increased.
3. An adaptive group computation approach in the peer-to-peer Grid computing systems.
4. Reduce the job management overhead and the total computation time
5. KOREA@Home Project

OUTLINE

1. Introduction
2. Traditional Peer-to-Peer Grid Computing Systems Model
3. Adaptive Group Computation Approach
4. Implementation of Proposed Approach
5. Conclusion

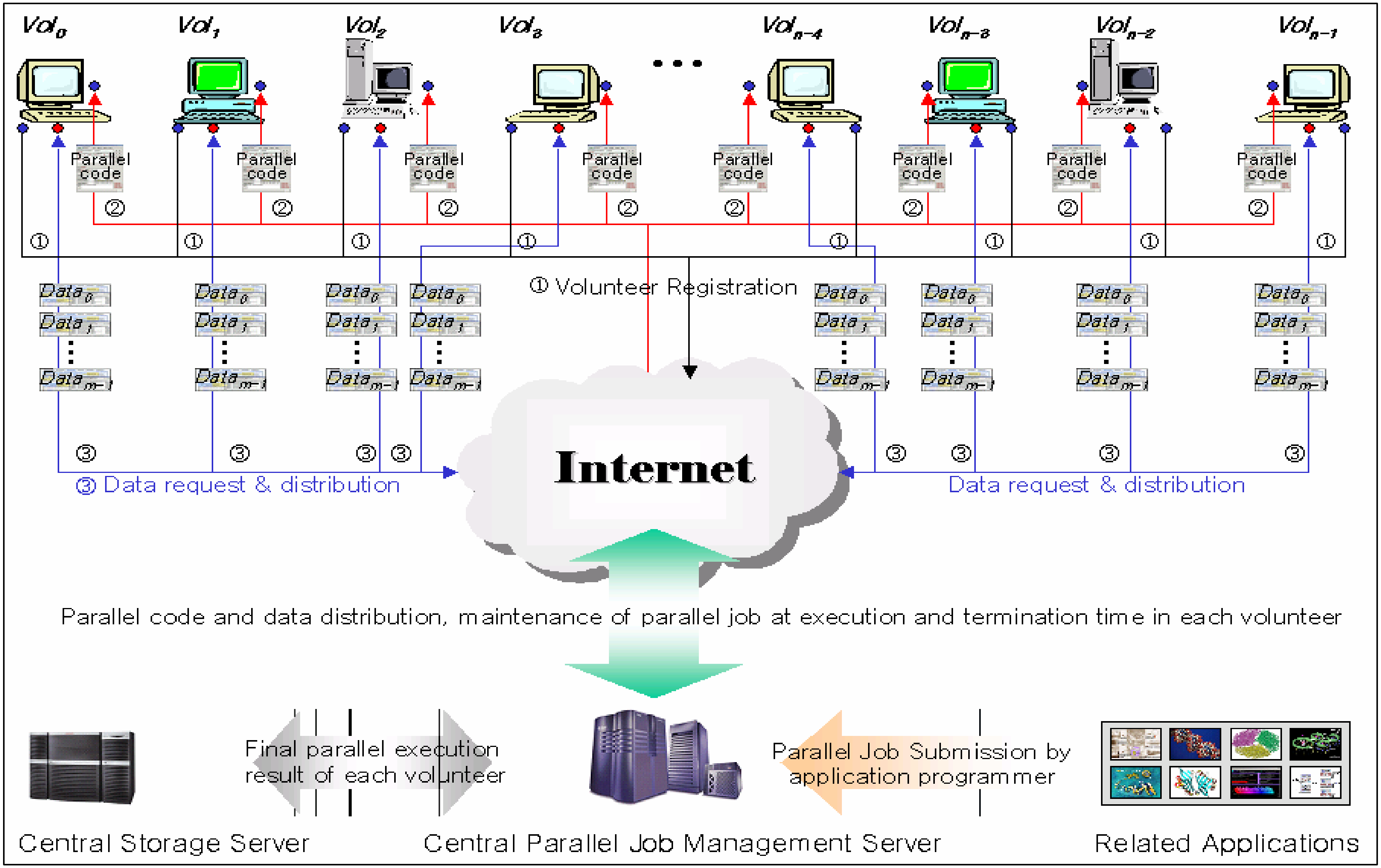
1. Introduction I

- Grid computing
    - Aims to offer pervasive access to a diverse collection of resources owned by different institutions through making virtual organization from resources in computation time.
- Peer-to-Peer Grid computing systems
    - Does not make any virtual organization in computation time
    - As the number of volunteers is increased, the job management overhead of the Central Parallel Job Management Server(CPJMS) is more and more increased.
- The architecture of the peer-to-peer Grid computing systems
    - Autonomous property of volunteers
        - Prevent volunteers from grouping according to their allocated job or registered resource semantics
    - Direct connection between the CPJMS and volunteers
        - High overhead

1. Introduction II

- Adaptive group computation approach
  - Group coordinator => Intermediate Job Management Deputy(IJMD)
  - Computation members
  - Flowing volunteers
- Computation groups
  - IJMD
  - Computation members
  - Maintained by proposed algorithm
- Contribution
  - Reduces the overhead of the CPJMS
  - Shorten the total computation time using IJMD
- KOREA@Home Project

2. Traditional Peer-to-Peer Grid Computing Systems Model I



① Volunteer Registration

③ Data request & distribution

Data request & distribution

Parallel code and data distribution, maintenance of parallel job at execution and termination time in each volunteer

Final parallel execution result of each volunteer

Parallel Job Submission by application programmer

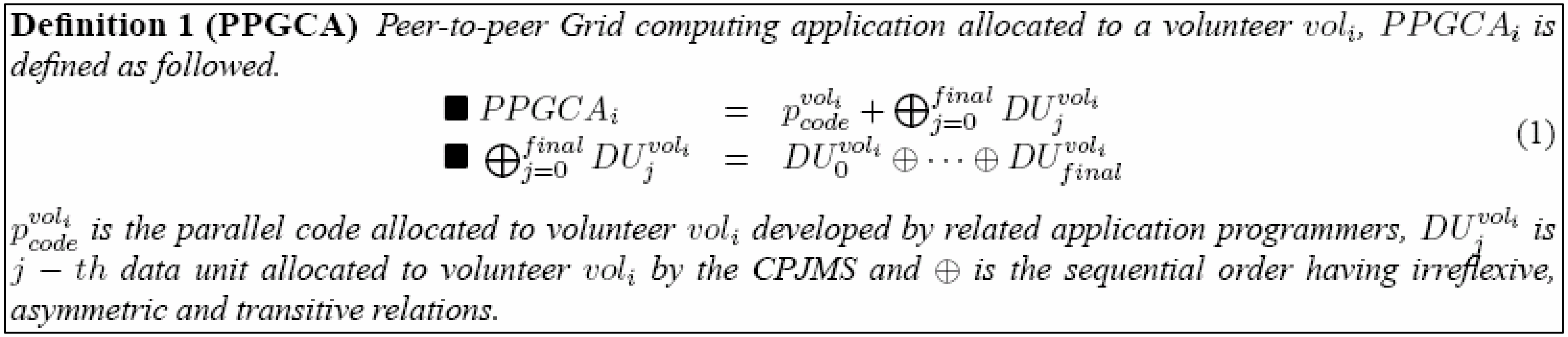
Central Storage Server

Central Parallel Job Management Server

Related Applications

2. Traditional Peer-to-Peer Grid Computing Systems Model II

- Peer-to-Peer Grid computing application

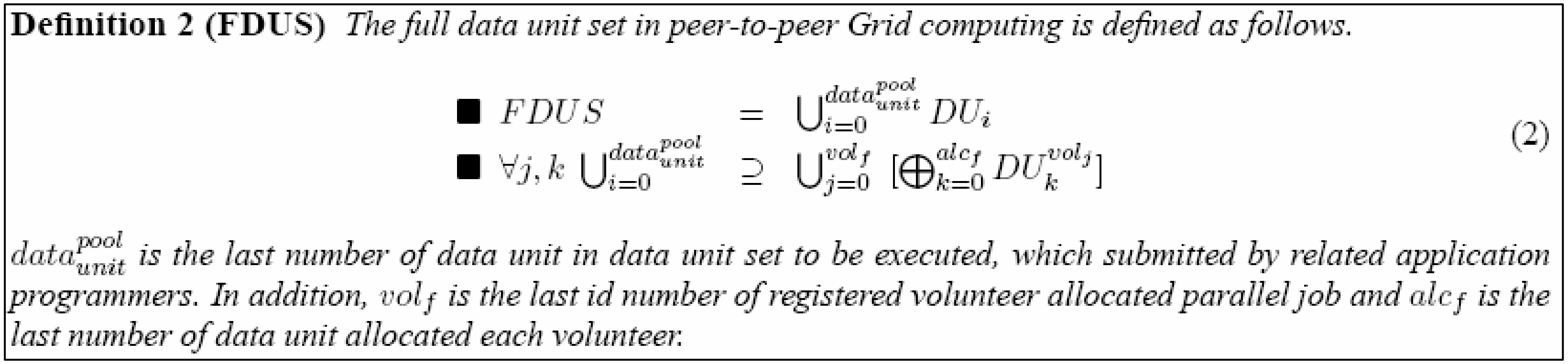


Definition 1 (PPGCA) *Peer-to-peer Grid computing application allocated to a volunteer vol_i, $PPGCA_i$ is defined as followed.*

$$\begin{aligned} \blacksquare\ PPGCA_i &= p_{code}^{vol_i} + \bigoplus_{j=0}^{final} DU_j^{vol_i} \\ \blacksquare\ \bigoplus_{j=0}^{final} DU_j^{vol_i} &= DU_0^{vol_i} \oplus \cdots \oplus DU_{final}^{vol_i} \end{aligned} \tag{1}$$

$p_{code}^{vol_i}$ is the parallel code allocated to volunteer vol_i developed by related application programmers, $DU_j^{vol_i}$ is $j-th$ data unit allocated to volunteer vol_i by the CPJMS and $\oplus$ is the sequential order having irreflexive, asymmetric and transitive relations.

- Full data unit set



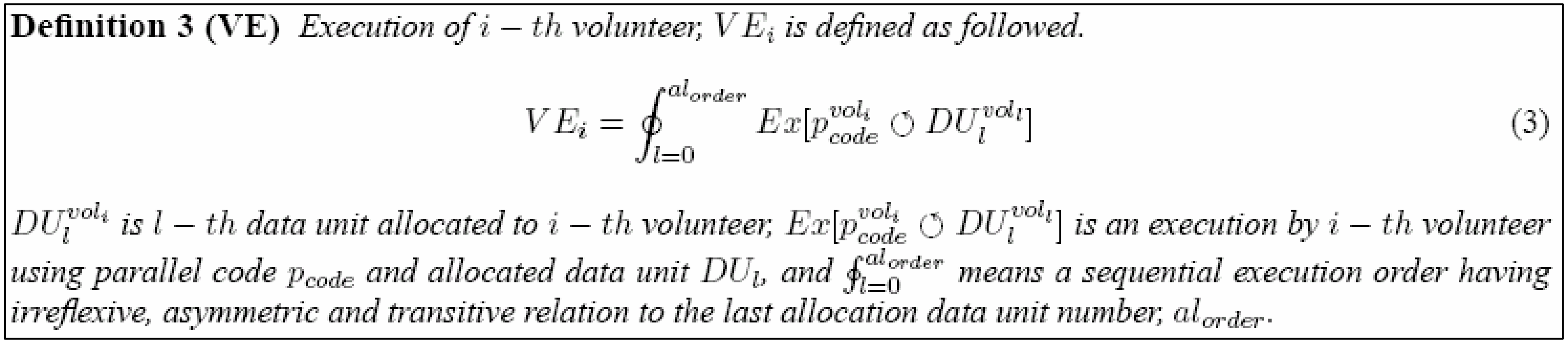
Definition 2 (FDUS) *The full data unit set in peer-to-peer Grid computing is defined as follows.*

$$\begin{aligned} \blacksquare\ FDUS &= \bigcup_{i=0}^{data_{unit}^{pool}} DU_i \\ \blacksquare\ \forall j,k \bigcup_{i=0}^{data_{unit}^{pool}} &\supseteq \bigcup_{j=0}^{vol_f} \left[\bigoplus_{k=0}^{alc_f} DU_k^{vol_j}\right] \end{aligned} \tag{2}$$

$data_{unit}^{pool}$ is the last number of data unit in data unit set to be executed, which submitted by related application programmers. In addition, vol_f is the last id number of registered volunteer allocated parallel job and alc_f is the last number of data unit allocated each volunteer.

2. Traditional Peer-to-Peer Grid Computing Systems Model III

- Execution of volunteer

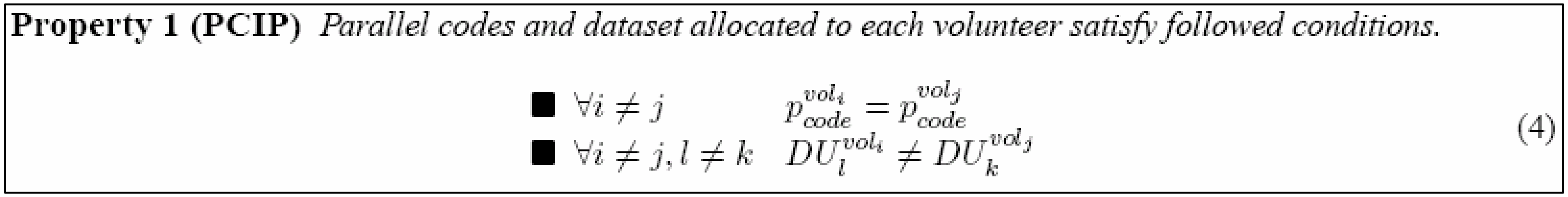


Definition 3 (VE) *Execution of $i-th$ volunteer, VE_i is defined as followed.*

$$VE_i = \oint_{l=0}^{al_{order}} Ex[p_{code}^{vol_i} \circlearrowleft DU_l^{vol_i}] \qquad (3)$$

$DU_l^{vol_i}$ is $l-th$ data unit allocated to $i-th$ volunteer, $Ex[p_{code}^{vol_i} \circlearrowleft DU_l^{vol_i}]$ is an execution by $i-th$ volunteer using parallel code p_{code} and allocated data unit DU_l, and $\oint_{l=0}^{al_{order}}$ means a sequential execution order having irreflexive, asymmetric and transitive relation to the last allocation data unit number, al_{order}.

- Property of parallel code and data unit



Property 1 (PCIP) *Parallel codes and dataset allocated to each volunteer satisfy followed conditions.*

$$\begin{array}{lll} \blacksquare & \forall i \neq j & p_{code}^{vol_i} = p_{code}^{vol_j} \\ \blacksquare & \forall i \neq j, l \neq k & DU_l^{vol_i} \neq DU_k^{vol_j} \end{array} \qquad (4)$$

3. Adaptive Group Computation Approach I

- Application program submission
  - Parallel code and full data unit set
  - The average computation time list per resource

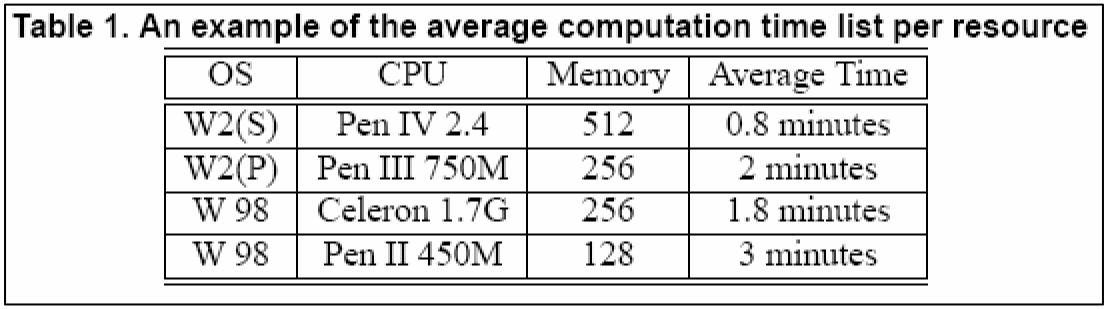


Table 1. An example of the average computation time list per resource

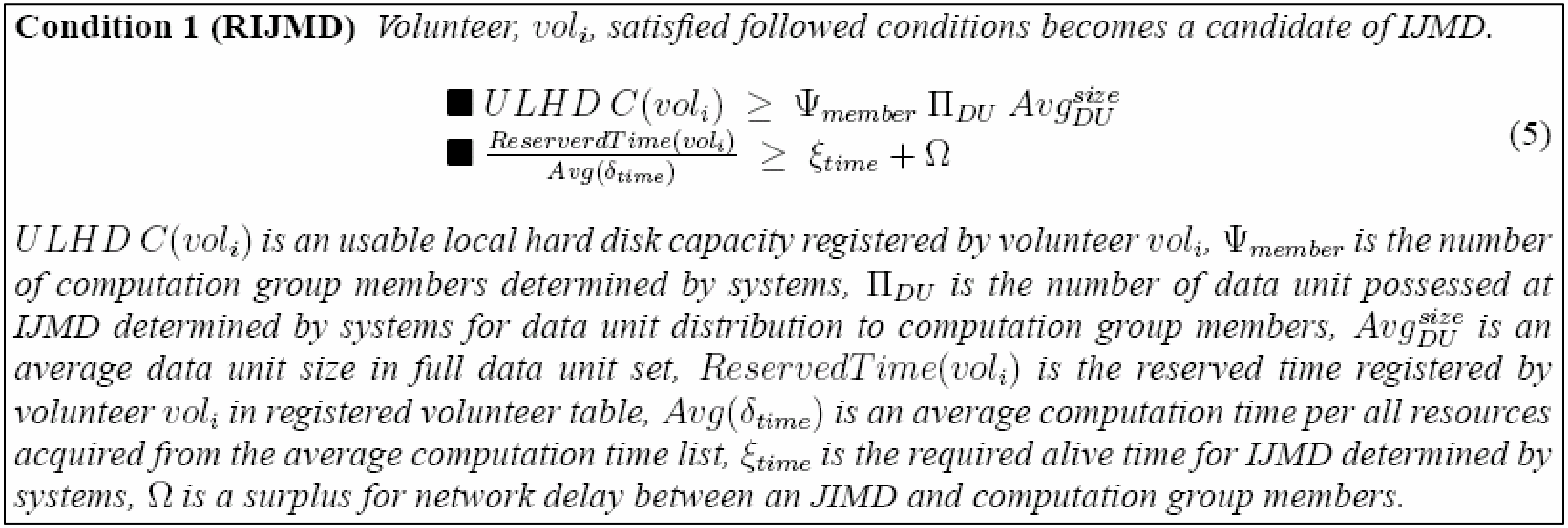
OS	CPU	Memory	Average Time
W2(S)	Pen IV 2.4	512	0.8 minutes
W2(P)	Pen III 750M	256	2 minutes
W 98	Celeron 1.7G	256	1.8 minutes
W 98	Pen II 450M	128	3 minutes

3. Adaptive Group Computation Approach II

- Volunteer registration
  - Local operating systems type
  - CPU type
  - Memory capacity
  - Usable local hard disk capacity
  - Volunteer address type
  - Using network bandwidth type
  - Resource submitting state
    - Time reserved : stably providing the resources during the reserved time
    - Only registered : not-stably providing the resources
  - Registered volunteer table

3. Adaptive Group Computation Approach III

- Making computation group members I
  - Selection procedure
    - Latest state message : to update the registered volunteer table
    - Required IJMD



Condition 1 (RIJMD) *Volunteer, vol_i, satisfied followed conditions becomes a candidate of IJMD.*

$$\begin{array}{l} \blacksquare\; ULHDC(vol_i) \geq \Psi_{member}\,\Pi_{DU}\,Avg_{DU}^{size} \\ \blacksquare\; \frac{ReserverdTime(vol_i)}{Avg(\delta_{time})} \geq \xi_{time} + \Omega \end{array} \quad (5)$$

$ULHDC(vol_i)$ is an usable local hard disk capacity registered by volunteer vol_i, Ψ_{member} is the number of computation group members determined by systems, Π_{DU} is the number of data unit possessed at IJMD determined by systems for data unit distribution to computation group members, Avg_{DU}^{size} is an average data unit size in full data unit set, $ReservedTime(vol_i)$ is the reserved time registered by volunteer vol_i in registered volunteer table, $Avg(\delta_{time})$ is an average computation time per all resources acquired from the average computation time list, ξ_{time} is the required alive time for IJMD determined by systems, Ω is a surplus for network delay between an JIMD and computation group members.

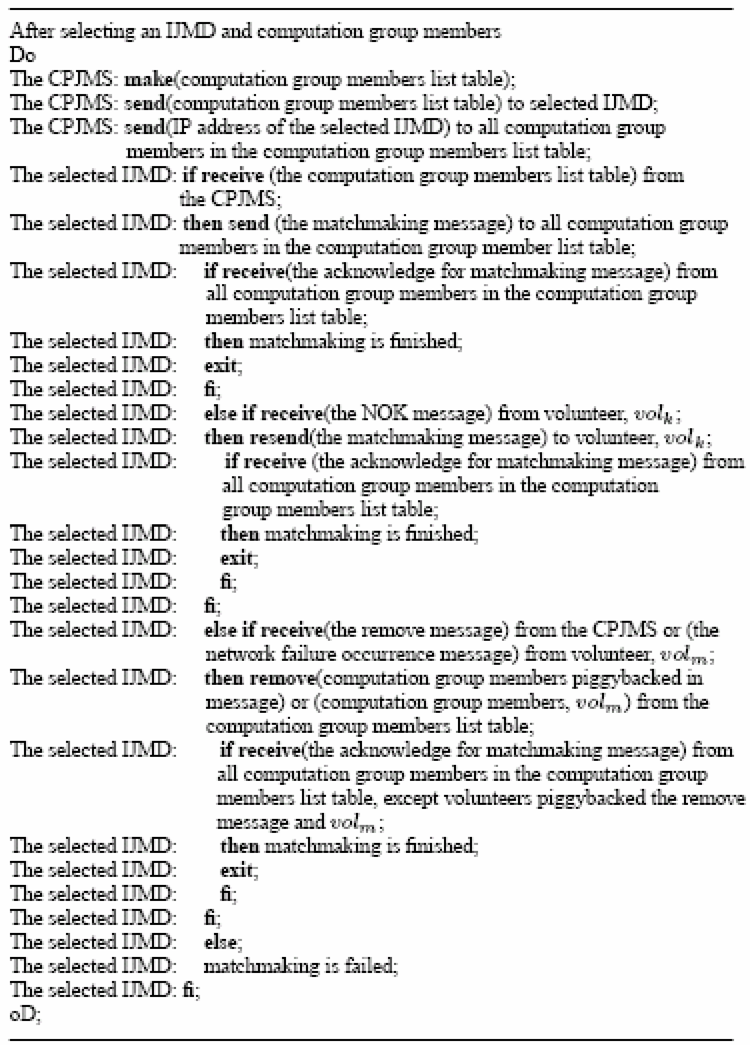
    - Reserving ready message
    - Reserving message

3. Adaptive Group Computation Approach IV

- Making computation group members II
  - Matchmaking computation group
    - Computation group member list table : CPJMS -> IJMD, Members
    - Matchmaking message : IJMD -> Members
    - Acknowledge for matchmaking message : Members -> IJMD
    - NOK message : Members -> IJMD if not received the matchmaking msg.
    - Network failure occurrence message : Members -> CPJMS
    - Remove message : CPJMS -> IJMD

3. Adaptive Group Computation Approach V

- Making computation group members III
  - Computation group matchmaking algorithm



```
After selecting an IJMD and computation group members
Do
The CPJMS: make(computation group members list table);
The CPJMS: send(computation group members list table) to selected IJMD;
The CPJMS: send(IP address of the selected IJMD) to all computation group
                members in the computation group members list table;
The selected IJMD: if receive (the computation group members list table) from
                      the CPJMS;
The selected IJMD: then send (the matchmaking message) to all computation group
                      members in the computation group member list table;
The selected IJMD:    if receive(the acknowledge for matchmaking message) from
                      all computation group members in the computation group
                      members list table;
The selected IJMD:    then matchmaking is finished;
The selected IJMD:    exit;
The selected IJMD:    fi;
The selected IJMD:    else if receive(the NOK message) from volunteer, $vol_k$;
The selected IJMD:    then resend(the matchmaking message) to volunteer, $vol_k$;
The selected IJMD:       if receive (the acknowledge for matchmaking message) from
                         all computation group members in the computation
                         group members list table;
The selected IJMD:       then matchmaking is finished;
The selected IJMD:       exit;
The selected IJMD:       fi;
The selected IJMD:    fi;
The selected IJMD:    else if receive(the remove message) from the CPJMS or (the
                      network failure occurrence message) from volunteer, $vol_m$;
The selected IJMD:    then remove(computation group members piggybacked in
                      message) or (computation group members, $vol_m$) from the
                      computation group members list table;
The selected IJMD:       if receive(the acknowledge for matchmaking message) from
                         all computation group members in the computation group
                         members list table, except volunteers piggybacked the remove
                         message and $vol_m$;
The selected IJMD:       then matchmaking is finished;
The selected IJMD:       exit;
The selected IJMD:       fi;
The selected IJMD:    fi;
The selected IJMD:    else;
The selected IJMD:    matchmaking is failed;
The selected IJMD: fi;
oD;
```

3. Adaptive Group Computation Approach VI

- Parallel code and data unit distribution
  - Parallel job matching table(data unit distribution table)

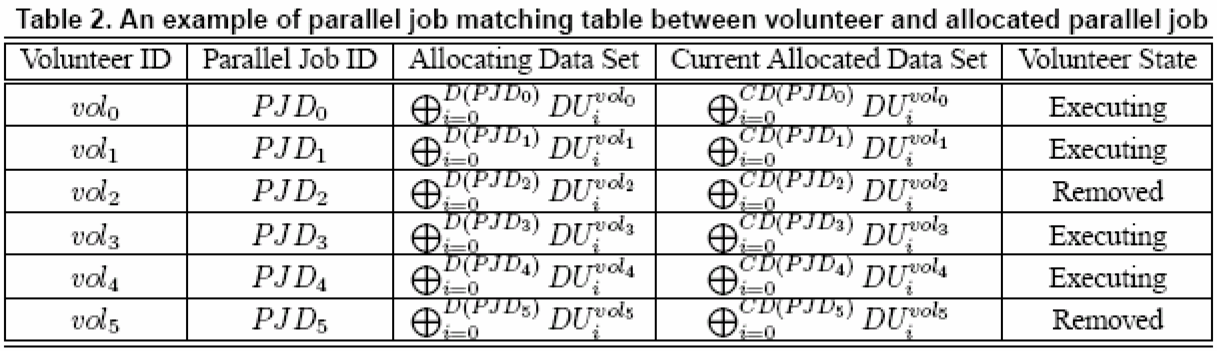


Table 2. An example of parallel job matching table between volunteer and allocated parallel job

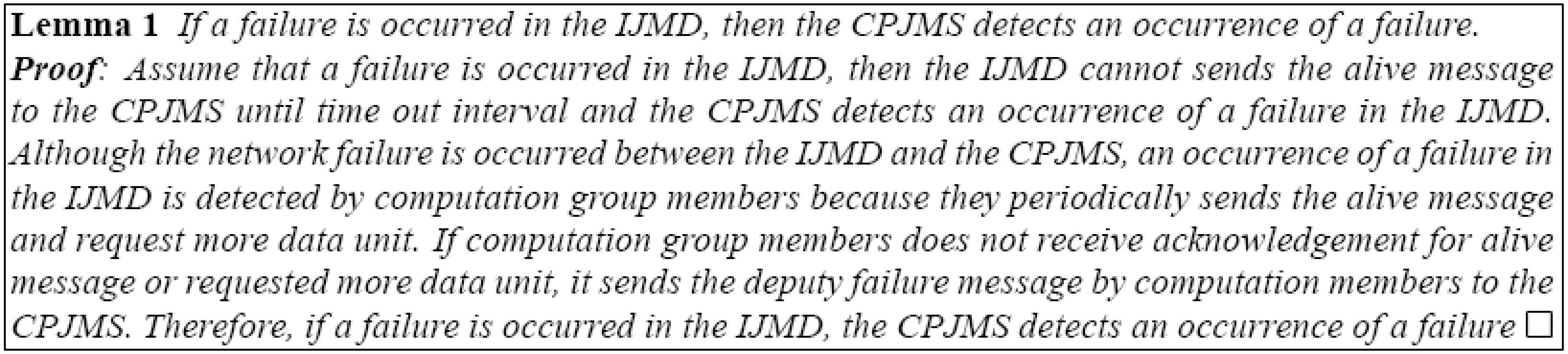
Volunteer ID	Parallel Job ID	Allocating Data Set	Current Allocated Data Set	Volunteer State
vol_0	PJD_0	$\bigoplus_{i=0}^{D(PJD_0)} DU_i^{vol_0}$	$\bigoplus_{i=0}^{CD(PJD_0)} DU_i^{vol_0}$	Executing
vol_1	PJD_1	$\bigoplus_{i=0}^{D(PJD_1)} DU_i^{vol_1}$	$\bigoplus_{i=0}^{CD(PJD_1)} DU_i^{vol_1}$	Executing
vol_2	PJD_2	$\bigoplus_{i=0}^{D(PJD_2)} DU_i^{vol_2}$	$\bigoplus_{i=0}^{CD(PJD_2)} DU_i^{vol_2}$	Removed
vol_3	PJD_3	$\bigoplus_{i=0}^{D(PJD_3)} DU_i^{vol_3}$	$\bigoplus_{i=0}^{CD(PJD_3)} DU_i^{vol_3}$	Executing
vol_4	PJD_4	$\bigoplus_{i=0}^{D(PJD_4)} DU_i^{vol_4}$	$\bigoplus_{i=0}^{CD(PJD_4)} DU_i^{vol_4}$	Executing
vol_5	PJD_5	$\bigoplus_{i=0}^{D(PJD_5)} DU_i^{vol_5}$	$\bigoplus_{i=0}^{CD(PJD_5)} DU_i^{vol_5}$	Removed

3. Adaptive Group Computation Approach VII

- Maintenance mechanism for computation group
  - Failure of IJMD(the CPJMS detecting)
    - Deputy failure message by the CPJMS : CPJMS -> Members
    - Failure event confirm message : Members -> CPJMS
  - Failure of IJMD(computation member detecting)
    - Deputy failure message by computation members : Members -> CPJMS
    - Failure confirm message : CPJMS ->IJMD
    - Network failure message : CPJMS->IJMD
  - Failure of computation members
    - The member failure message by the IJMD : IJMD -> CPJMS

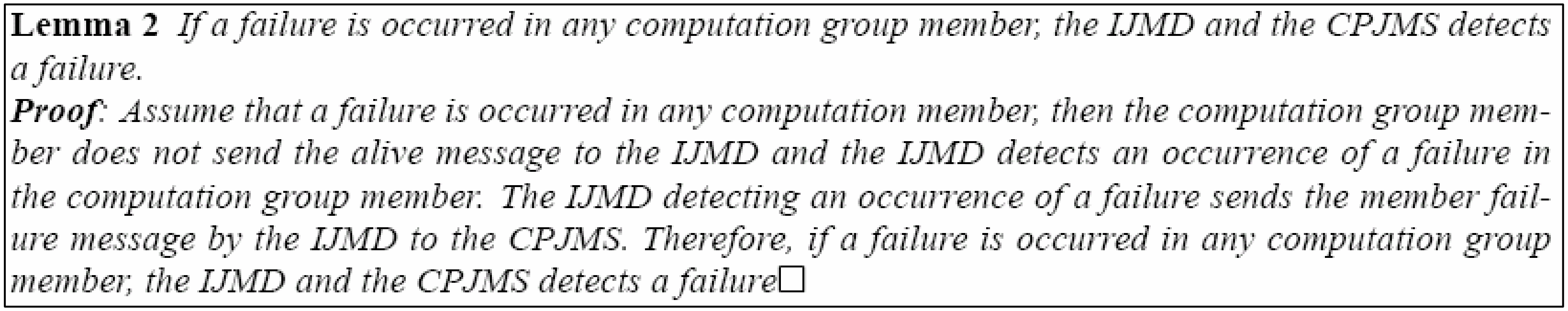
3. Adaptive Group Computation Approach VIII

- Correctness I



Lemma 1 *If a failure is occurred in the IJMD, then the CPJMS detects an occurrence of a failure.*

Proof: *Assume that a failure is occurred in the IJMD, then the IJMD cannot sends the alive message to the CPJMS until time out interval and the CPJMS detects an occurrence of a failure in the IJMD. Although the network failure is occurred between the IJMD and the CPJMS, an occurrence of a failure in the IJMD is detected by computation group members because they periodically sends the alive message and request more data unit. If computation group members does not receive acknowledgement for alive message or requested more data unit, it sends the deputy failure message by computation members to the CPJMS. Therefore, if a failure is occurred in the IJMD, the CPJMS detects an occurrence of a failure* □

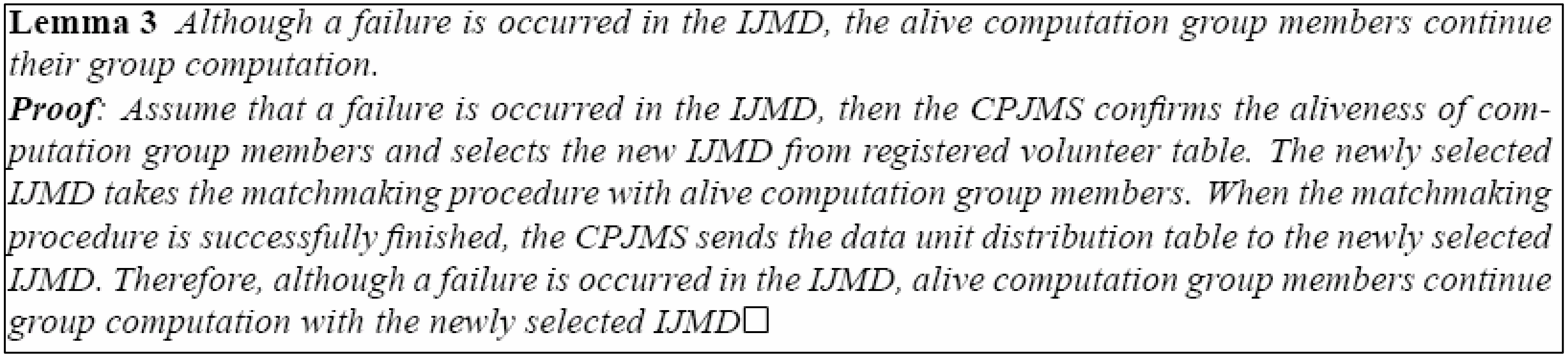


Lemma 2 *If a failure is occurred in any computation group member, the IJMD and the CPJMS detects a failure.*

Proof: *Assume that a failure is occurred in any computation member, then the computation group member does not send the alive message to the IJMD and the IJMD detects an occurrence of a failure in the computation group member. The IJMD detecting an occurrence of a failure sends the member failure message by the IJMD to the CPJMS. Therefore, if a failure is occurred in any computation group member, the IJMD and the CPJMS detects a failure*□

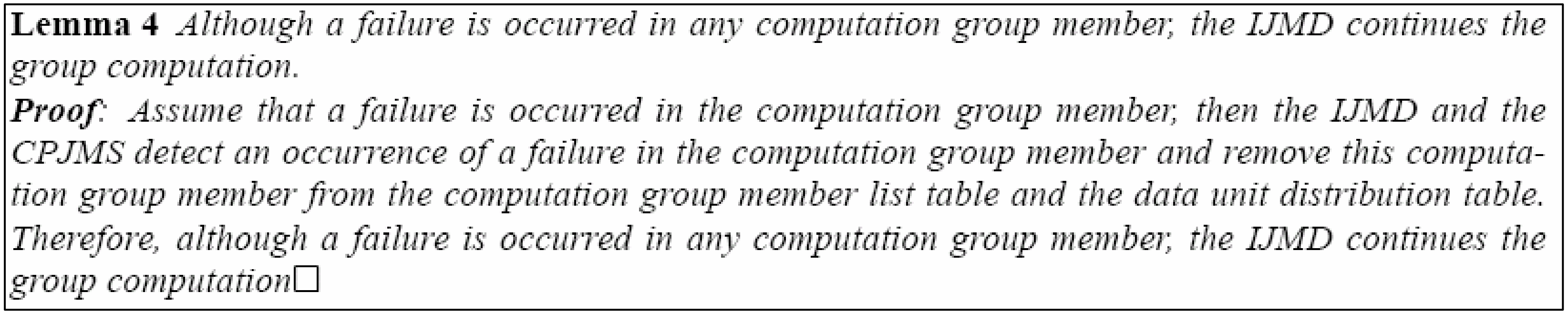
3. Adaptive Group Computation Approach IX

- Correctness II



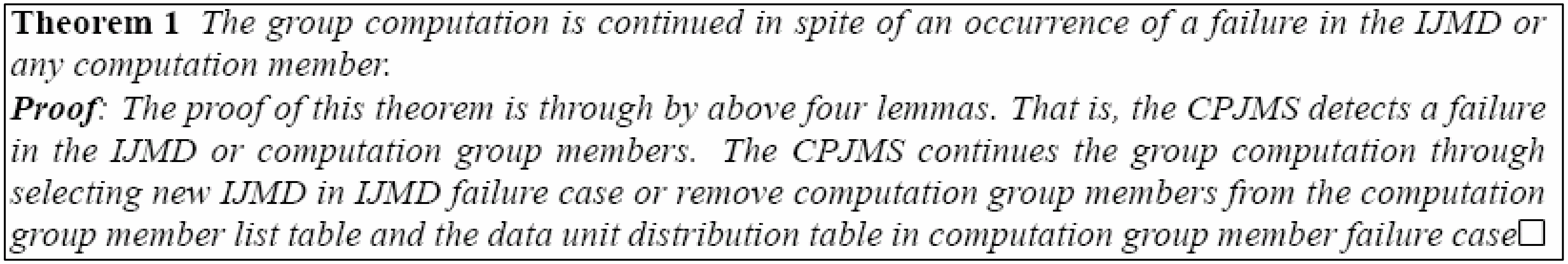
Lemma 3 *Although a failure is occurred in the IJMD, the alive computation group members continue their group computation.*

Proof: *Assume that a failure is occurred in the IJMD, then the CPJMS confirms the aliveness of computation group members and selects the new IJMD from registered volunteer table. The newly selected IJMD takes the matchmaking procedure with alive computation group members. When the matchmaking procedure is successfully finished, the CPJMS sends the data unit distribution table to the newly selected IJMD. Therefore, although a failure is occurred in the IJMD, alive computation group members continue group computation with the newly selected IJMD*□



Lemma 4 *Although a failure is occurred in any computation group member, the IJMD continues the group computation.*

Proof: *Assume that a failure is occurred in the computation group member, then the IJMD and the CPJMS detect an occurrence of a failure in the computation group member and remove this computation group member from the computation group member list table and the data unit distribution table. Therefore, although a failure is occurred in any computation group member, the IJMD continues the group computation*□

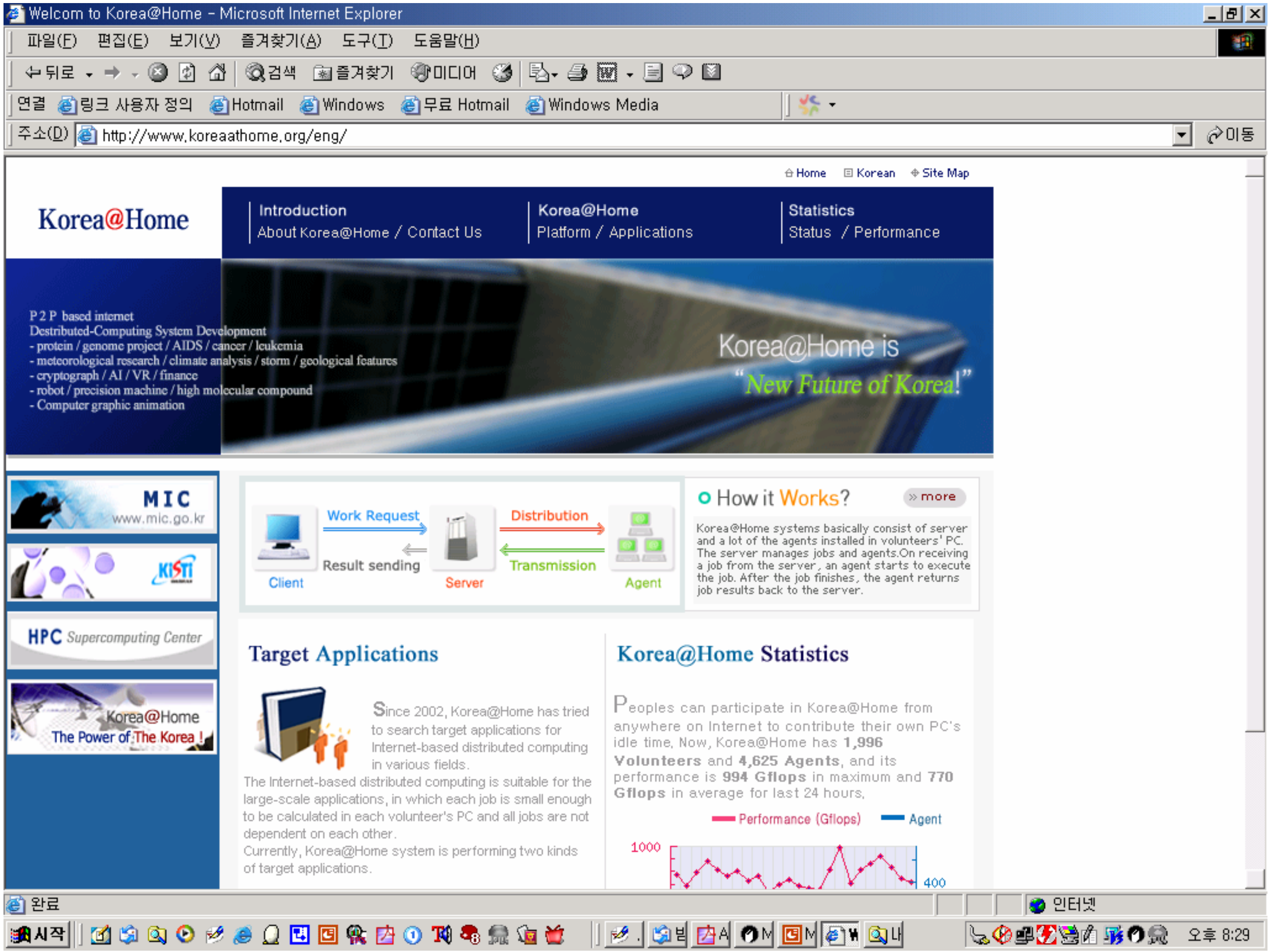


Theorem 1 *The group computation is continued in spite of an occurrence of a failure in the IJMD or any computation member.*

Proof: *The proof of this theorem is through by above four lemmas. That is, the CPJMS detects a failure in the IJMD or computation group members. The CPJMS continues the group computation through selecting new IJMD in IJMD failure case or remove computation group members from the computation group member list table and the data unit distribution table in computation group member failure case*□

4. Implementation of Proposed Approach I

The KOREA@Home Project I



P2P based internet
Destributed-Computing System Development
- protein / genome project / AIDS / cancer / leukemia
- meteorological research / climate analysis / storm / geological features
- cryptograph / AI / VR / finance
- robot / precision machine / high molecular compound
- Computer graphic animation

How it Works?

Korea@Home systems basically consist of server and a lot of the agents installed in volunteers' PC. The server manages jobs and agents.On receiving a job from the server, an agent starts to execute the job. After the job finishes, the agent returns job results back to the server.

Target Applications

Since 2002, Korea@Home has tried to search target applications for Internet-based distributed computing in various fields.
The Internet-based distributed computing is suitable for the large-scale applications, in which each job is small enough to be calculated in each volunteer's PC and all jobs are not dependent on each other.
Currently, Korea@Home system is performing two kinds of target applications.

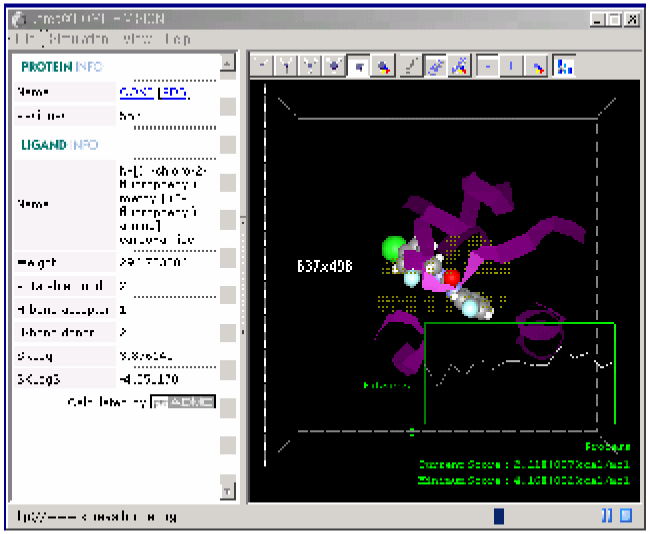
Korea@Home Statistics

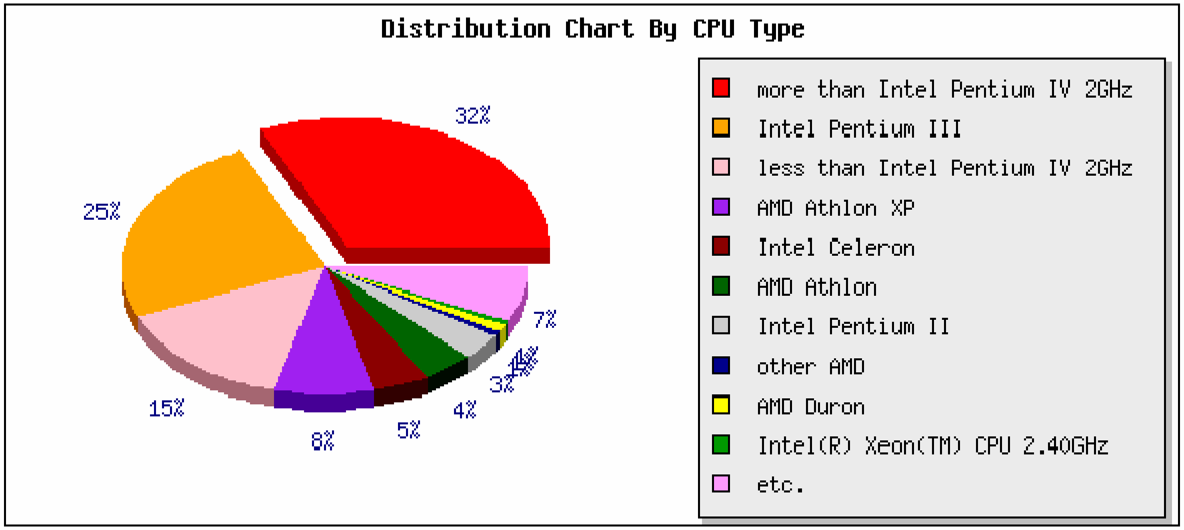
Peoples can participate in Korea@Home from anywhere on Internet to contribute their own PC's idle time. Now, Korea@Home has **1,996 Volunteers** and **4,625 Agents**, and its performance is **994 Gflops** in maximum and **770 Gflops** in average for last 24 hours.

4. Implementation of Proposed Approach II

- The KOREA@Home Project II

<Korea@Home Agent>



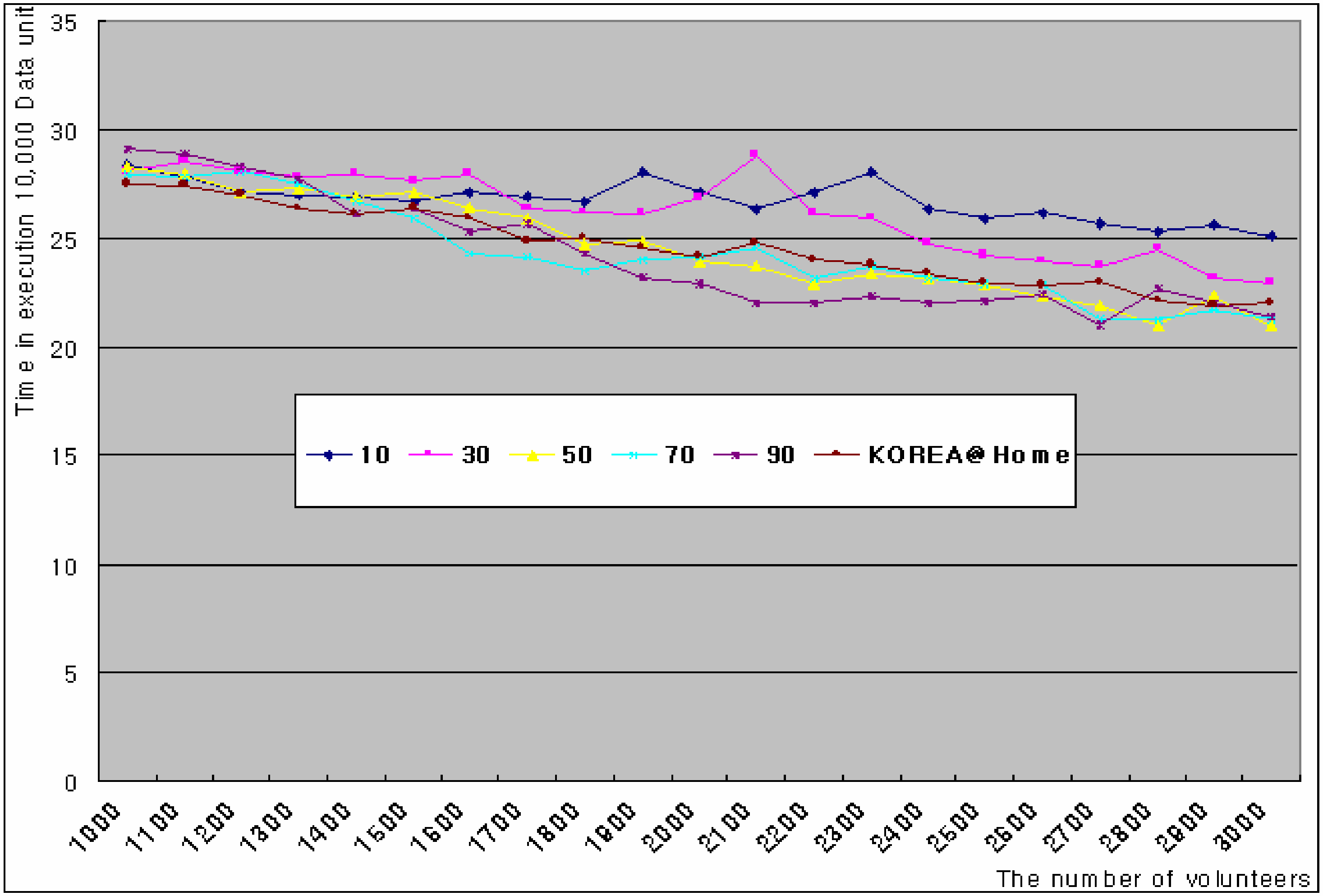


(a) Volunteer execution screen in KOREA@Home Project

(b) Distribution chart by CPU type of volunteers in KOREA@Home Project

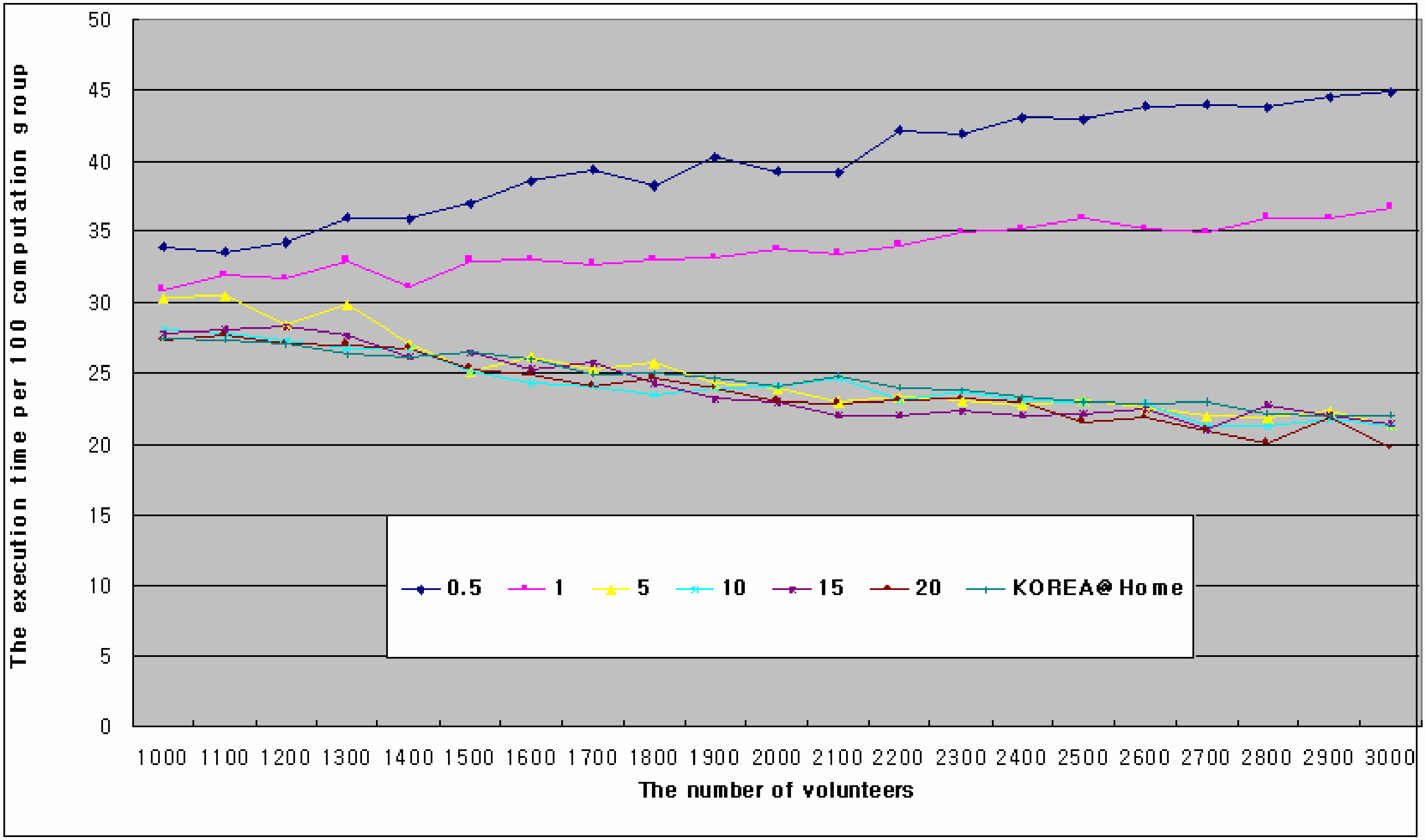
4. Implementation of Proposed Approach IV

- Current implementation of proposed approach I
  (according to the number of computation group members)



4. Implementation of Proposed Approach V

- Current implementation of proposed approach II (according to the registered time of the IJMD)



5. Conclusion

- Adaptive grid computation approach in the peer-to-peer Grid computing systems
- Computation group maintenance mechanism and correctness
- KOREA@Home Project (http://www.koreaathome.org/eng/)
- Experiment result of proposed approach