Disconnected Operation Service in Mobile Grid Computing[*]

Sang-Min Park, Young-Bae Ko, and Jai-Hoon Kim

Graduate School of Information and Communication
Ajou University, South Korea
{smpark, youngko, jaikim}@ajou.ac.kr

Abstract. In this paper, we discuss on the extension of grid computing systems in mobile computing environments, where mobile devices can be effectively incorporated into the grid either as service recipients or as more valuable service providers. First, based on the present grid architecture, we try to figure out what would be the newly required services in such a mobile/grid integrated architecture. There are a number of challenging issues when taking mobile environment into account, such as intermittent connectivity, device heterogeneity, and weak security. Among these issues to solve, we particularly focus on a disconnected operation problem in this paper since mobile resources are prone to frequent disconnections due to their confined communication range and device mobility. We develop a new job scheduling algorithm for mobile grid system and evaluate it by various methods such as mathematical analysis, simulation, and prototype implementation.

1 Introduction

A grid computing system [1] is a large-scaled distributed system, designed for effectively solving very complicated scientific or commercial problems such as gene analysis, drug design, and climate simulations. In the grid, the computing resources are autonomously managed at different locations in a distributed manner. They are aggregated through a global Internet-wide network, forming a computational grid, by which a huge workload can be run and completed with more improved performance in terms of computation speed and throughput. Mobile computing [2] is another distinct paradigm of traditional distributed systems, considering mobility, portability and wireless communications. The recent advances in wireless communication technologies and portable mobile appliances make more number of people be eligible to

[*] This work is supported by a grant of the International Mobile Telecommunications 2000 R&D Project, Ministry of Information & Communication in South Korea, by grant No. R05-2003-000-10607-0 from Korea Science & Engineering Foundation, by grant No. (R01-2003-000-10794-0) from the Basic Research Program of the Korea Science & Engineering Foundation, and by ITA Professorship for Visiting Faculty Positions in Korea (International Joint Research Project) by Ministry of Information & Communication in South Korea.

access information services through a shared network infrastructure (e.g., Internet) with their own mobile computing devices, regardless of their physical location.

The current grid architecture and algorithms do not take into account the mobile computing environment since mobile devices have not been seriously considered as valid computing resources or interfaces in grid communities. It has been just recently given attention to integrate these two emerging techniques of mobile and grid computing −for example, in [3,4], although they do not elaborate on how the mobile devices may be incorporated in the current grid architecture. In our view of the mobile grid computing integration, there are two possible roles of mobile devices in grid. First, mobile devices can be used as interfaces to the grid. Thus, a mobile device can initiate the use of grid resources, monitor the jobs being executed remotely, and take any results from the grid. Secondly and more interestingly, mobile devices can be assumed to participate in grid as computing resource providers, not just service recipients. We believe that recent advancement of technologies on mobile devices and wireless communications make this scenario more feasible.

Of course, clearly, many issues become the challenges when we consider mobile devices as one of grid computing resources or interfaces. Some examples of limitations that possibly hinder the integration are relatively poor local resources (in terms of computation speed, memory), battery constraints, unreliable connectivity status, weak security and so on. These limitations and constraints should be dealt with accordingly before mobile grid integration is fully enabled. We present several technical issues of mobile grid system in the first part of this paper. However, among those arise, we particularly focus on a dynamic nature of device connectivity and develop a new scheduling algorithm that provides reliable performance in a mobile grid system where the devices are prone to frequent disconnections.

2 Related Works

One of the most critical characteristics of the mobile grid system is the intermittent connectivity of mobile devices. We can find similar situations in Peer-to-Peer computing area. In general, P2P system consists of huge number of computing devices and they can act either as a client or a server. In P2P, each machine's CPU cycles, storages, and contents can be shared in order to broaden their resource limitations [5,6,7,8]. SETI@home project [7] provides a successful story that large P2P system can be effectively used for high performance applications by aggregating cycles of desktop PCs. In a P2P System, users are free to join and leave the network and one study showed that a node remained in a connection state only for 28% of time on average [6]. These frequent disconnections of the node degrade the computing performance significantly and providing the redundancy of workloads is the way used to compensate the deficiency [8]. In other words, the same work unit is disseminated into some number of nodes and the first result generated by the most reliable and the fast node is received. However, this strategy of redundancy cannot be applied in mobile grid systems. While the redundancy guarantees the improvement of performance, it also wastes the resources greatly. Although the strategy can be successfully applied in P2P systems where resources are abundant, relatively small number of nodes and high sensitivity of resource dissipation in mobile grid system will not allow large number of duplication of work units. In this paper, we present different approach to deal with performance degradation. We propose a new scheduling algorithm that takes into account of the intermittent connectivity of mobile nodes.

Since grid consists of several number of autonomously managed local resources (MPPs, clusters, and workstations), the scheduling paradigm would fall into two categories, global and local scheduling. Various global scheduling algorithms have been operational such as application-level scheduling [9], high-throughput scheduling [10], economy-based scheduling [11], and data-centric scheduling [12]. Once the grid job is scheduled and submitted by global scheduler, it is scheduled again by local resource managers like PBS [13], LSF [14], and Condor [10]. First-in, first-out is the most prominent scheduling policy in the local managers, yet more sophisticated algorithms are provided within the specific managers. The scheduling algorithm that we present in this paper would be classified in local scheduling since the scheduling decision is only applicable to local mobile resources.

3 Two Roles of Mobile Devices in Mobile Grid Computing

In this section, we present possible architecture of the mobile grid system and several technical issues that are to be dealt with in further researches. We depict an expected view of mobile grid system in Figure 1.

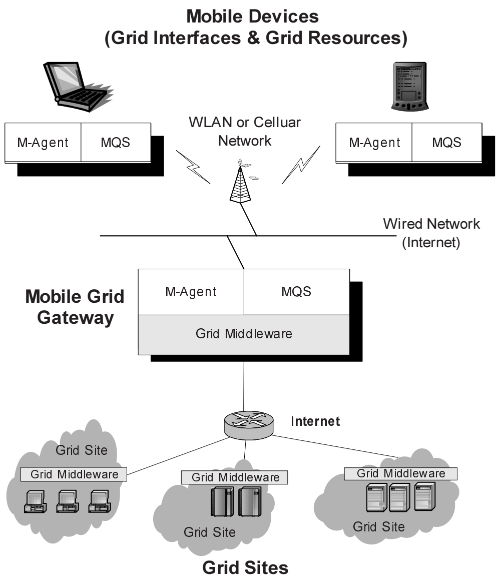

Fig. 1. Mobile Grid System

As illustrated in the Figure, the grid system is divided into three parts: static grid sites, a group of mobile devices, and a gateway interconnecting static and mobile resources. Mobile devices are wirelessly connected to Internet through a wireless LAN or a long-range cellular network like GSM and CDMA. Note that, in the figure, grid middleware, e.g., Globus [15], is installed on the gateway to provide interoperability between the virtual mobile grid site and other static grid sites. Also note that such a middleware layer is not necessary to be in a mobile device itself. This is important because our model does not require a heavy grid middleware, like Globus, to be installed on the thin mobile device. Newly added grid components realizing mobile grid computing are comprised of M-Agent (Mobile Agent), and MQS (Mobile Queuing Server). They are both on the gateway and mobile device side.

3.1 Mobile Interface Role to Grid

Mobile devices are exposed to frequent disconnections from the network. Also the mobile system itself is unreliable. Hence, it is not a good idea for the mobile devices to interact with static grid sites directly. Simply imagine that the mobile devices are suddenly disconnected during the data exchange with grid sites. Sometimes, the jobs running on grid sites need interactions with mobile user while mobile user is not in the connection state. Thus, a reliable static system should perform job submission, monitoring, cancellation, and completion process on behalf of the mobile user. In our architecture, M-Agent on the gateway performs that role of agent for the unreliable mobile devices. The M-Agent may perform two key roles as follows:

- **Adapting to various interfaces of mobile devices:** M-Agent on mobile device interacts with the mobile user, that is, the mobile user submits, monitors the grid job, and observes the results, through M-Agent. Various kinds of mobile devices have different appearances. While laptop has similar interfaces with PCs, interfaces of PDA or Handheld device is far from that of PC's. They have smaller displays, no keyboards, etc. Hence, the interface of M-Agent on mobile device should adapt to the specific class of mobile devices while sustaining the same functionality.
- **Reliable job management:** The mobile device has several constraints. The most serious problem is that its availability is unreliable due to intermittent connectivity or limited battery life problems. Thus, it is not a good approach to manage the running job directly from the mobile device. In order to solve problems induced by unreliable availability, we propose an object, **Job-Proxy**. When a mobile user submits a job through M-Agent, the M-Agent on gateway creates a Job-Proxy which performs job related tasks according to the received information from the mobile user. If the M-Agent detects an error with mobile devices, Job-Proxy becomes operational and starts to monitor executing job on the grid sites. If some interactions between the executing job and mobile user are needed (e.g., when an additional input data should be given), the Job-Proxy performs the interaction on behalf of the mobile user, based on the information received from the mobile user beforehand. When the grid job is completed, the result from the job is temporarily stored on the Job-Proxy. If the mobile device is in a good condition to receive the results, it immediately relays the result data. Otherwise, it waits until the mobile device becomes ready for receiving the result data. The mobile user may set the time-out duration of the Job-Proxy, i.e., the

running job in static grid sites will be canceled after a certain amount of time and all the resources in gateway allocated for the Job-Proxy is freed. This is because the mobile user may not be able to explicitly request cancellation of the job although running job is not needed anymore. If we do not cancel the job automatically, sustaining job execution unnecessarily consumes resources in both static grid sites and the mobile grid gateway.

3.2 The Service Provider Role to Grid

We present another aspect of mobile grid system that supports the integration mobile devices as grid resources. Even though, currently, both long-range access technologies (GSM and CDMA) and short-range connectivity (WLAN) is available, we assume that the WLAN is provided as a wireless medium for the mobile grid system. This is because not only does the wireless LAN guarantee much faster data transfer rate (11Mbps for the IEEE 802.11b), but also it is easier to control with respect to number of nodes, security concerns, and network administrations. In addition to them, WLAN is the most favored method of wireless communication for computing devices (laptops and PDAs), while the cellular networks are still dominantly used for voice communications. Except that the computing nodes are not connected through physical lines, the mobile grid system becomes similar to the conventional high performance clusters.

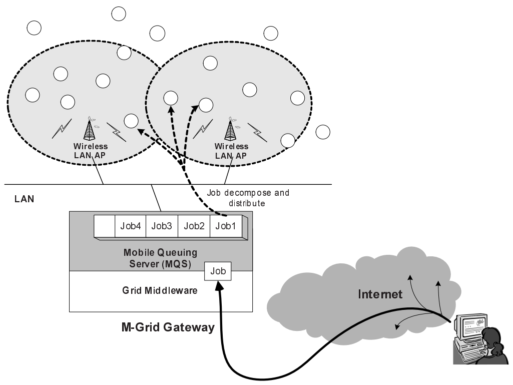

Fig. 2. Mobile resource provision to Grid using MQS

As shown by Figure 2, MQS (Mobile Queuing Server) on the gateway is a core component for the integration. The grid user submits jobs to several grid sites, and one of the grid sites performing the job is comprised of mobile devices. The group of mobile devices in the mobile grid system is viewed as a single virtual machine having several computing nodes. The virtual composing is provided by MQS in our architecture. When the job is submitted to this mobile grid site, the grid middleware (e.g., Globus) on the gateway relays the job information such as executable file location, input parameters and the location to store output files, to the MQS. Basic functions of MQS are similar to the conventional Job Queuing systems such as PBS, LSF, and NQE [16]. In addition to the basic job queuing functions, MQS cares the heterogeneity and unreliability of mobile nodes in order to hide unique characteristics of them. The mobile resources are viewed as a reliable single machine like the one in static grid site so that the grid job initiator can make use of them without further complex considerations. Followings are detailed explanation of services MQS should provide. Except for the scheduling issue, we leave these services as future research topics.

- **Fault Tolerant Service:** Unlike job queuing service in static systems, availability of resources changes dynamically in mobile grid system. The node may suddenly disappear due to disconnections or exhausted power, thus MQS should carefully monitor the status of mobile nodes. Fault tolerant services should be provided by MQS so that the unreliability of mobile node does not decrease the performance much. Time out strategy which re-submits the job when the result is not received in a determined time limit would be one of the viable approaches.
- **Hiding Platform Heterogeneity:** One of the most complex problems to deal with in mobile grid system is the hardware and OS heterogeneity. Large portion of laptop PC's are operated in Microsoft Windows, while, currently, most grid applications are running on Unix-based platforms. When we take the PDA and Handheld devices into account, the problem becomes more serious. Enforcing the care of heterogeneity to grid programmer would not be feasible, thus the MQS should hide the heterogeneity of individual mobile device.
- **Job Decomposing Service:** The job may contain many workloads. As an example, if we assume the job is Bioinformatics application which analyzes the gene sequences, the job will be composed of many gene sequences to be analyzed. A sequence is viewed as a workload in this case. The overall workloads in a job can be processed by a single node in a static grid site within a reasonable time unit. However, a single mobile node may not process the same amount of workloads in a reasonable time due to its resource limitation and unreliability, thus the workloads in a job should be decomposed and distributed to several mobile nodes according to their performance and reliability. MQS needs information about the application's workload composition to perform the job decomposing service, and this information can be provided by job initiator.
- **Job Scheduling:** In a static grid environment, job scheduling policy on a grid site is generally based on a simple FCFS (First Come, First Served). In some systems, it is based on the computing capability, i.e., when the job arrives at a grid site, the job is distributed to the node which is likely to compute it within the shortest time. Reliability of computing node is not a first-class concern for the job scheduling in the static grid environment. However, if the mobile environment is taken into account, we should carefully consider the inherent unreliability of mobile devices in order to make optimal scheduling decision. We cover this scheduling issue in the following section 4.

4 The Proposed Job Scheduling Algorithm

Among various challenging issues like energy sensitivity and weak security, we concentrate on the unreliable connectivity of mobile environment hereafter. Frequent disconnections of mobile node affect the job performance significantly. Job scheduling is made based on the expected execution time on various mobile nodes. When there is available mobile node, job (workload) is extracted from the job queue and delivered to the mobile node with related input data. After completing job execution on a mobile node, job output is returned to the MQS (Mobile Queuing Server) and the mobile node performs next job. Figure 3 represents the job execution process performed on the mobile node. Since Master-Worker style applications, whose tasks can be scheduled independently of one another, are preferred form of applications in grid [10,11,17], the job on the mobile node does not need communication during its execution. However, the mobile node should be in connection state during input and output data transfer. If the mobile node is not in the connection state, MQS waits for its reconnection. Figure 4 shows the state transition diagram of a mobile node. State C and D denotes the connection state (C) and disconnection state (D), respectively. λ is the disconnection rate, and μ is the reconnection rate of the mobile node, which are governed by Poisson process.

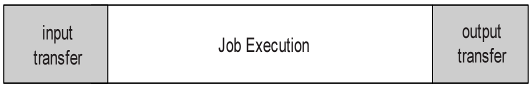

Fig. 3. Job execution process on a mobile node

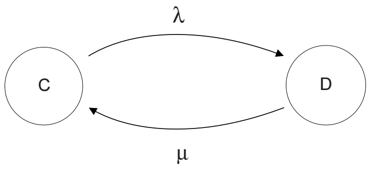

Fig. 4. State transition diagram of mobile node

When a number of jobs arrive at the MQS, or a number of workloads are decomposed from a single job, the MQS selects certain mobile nodes which perform the jobs. In a static grid environment, we achieve improving performance as the number of participating computing nodes grows. However, the rule changes when the mobile devices participate in the computing. Links of some mobile nodes are unreliable, that is, they are likely to be in the disconnection state for a long time while others are not. The unreliable mobile nodes may decrease the overall job execution performance because the MQS should wait for the reconnection of any disconnected node in order to complete the jobs. We should determine whether it is better to include a mobile node in the computing or not. Followings are the analysis to determine the participating nodes which result in the best job performance.

We consider all the mobile nodes (i.e., member nodes) as the potential participants in the computing whether they are initially in the connection state or not, at the time of job distribution. Followings are the analysis to obtain the expected time to execute a job on the mobile nodes.

Let $f_c(t)$ and $f_d(t)$ denotes the expected time required to transfer the data (input and output) of *t* time unit (data transfer requires *t* time unit without disconnection) when the mobile node is initially in the connection state and disconnection state, respectively. Let P_C and P_D be the probability of being in connection state and disconnection state, respectively. We can obtain following equations from state transitions shown in Figure 4:

$$P_C + P_D = 1$$
$$-\lambda P_C + \mu P_D = 0$$

We can compute P_C and P_D as follows:

$$P_C = \frac{\mu}{\mu+\lambda}, \qquad P_D = \frac{\lambda}{\mu+\lambda}$$

Then, $C(t,\varepsilon) = f_c(t+\varepsilon) - f_c(t)$ is the time required to perform ε time units of data transfer starting from time *t* when the mobile node is initially in the connection state. We can compute $C(t,\varepsilon)$ for two cases as follows:

- No disconnection occurs during ε (probability is $e^{-\lambda\varepsilon}$): ε time units are required to transfer the data.
- A disconnection occurs during ε (probability is $1-e^{-\lambda\varepsilon}$): $\frac{1}{\mu}$ time units are expected to stay in disconnection state.

ε time units are required for data transfer and $\frac{1}{\mu}$ time units are required on average for waiting for reconnection. Hence, we can obtain $C(t,\varepsilon)$ as follows:

$$C(t,\varepsilon) = f_c(t+\varepsilon) - f_c(t)$$
$$= e^{-\lambda\varepsilon}\varepsilon + (1-e^{-\lambda\varepsilon})\left(\varepsilon + \frac{1}{\mu}\right)$$

To compute $f_c(t)$:

$$\frac{\partial f(t)}{\partial t} = \lim_{\varepsilon\to 0}\frac{f(t+\varepsilon)-f(t)}{\varepsilon}$$
$$= \lim_{\varepsilon\to 0}\frac{e^{-\lambda\varepsilon}\varepsilon + (1-e^{-\lambda\varepsilon})\left(\varepsilon+\frac{1}{\mu}\right)}{\varepsilon} = \frac{\lambda}{\mu}+1$$

Thus, we obtain expected time required to transfer the data of *t* time unit connection (data transfer requires *t* time unit without disconnection) when the mobile node is initially in connection state as follows:

$$f_c(t) = \left(\frac{\lambda}{\mu}+1\right)t$$

Expected time required to transfer the data of *t* time unit connection when the mobile node is initially in disconnection state are obtained as follows (In addition, $\frac{1}{\mu}$ time units are required for reconnection on average.):

$$\begin{aligned} f_d(t) &= \frac{1}{\mu} + f_c(t) \\ &= \frac{1}{\mu} + \left(\frac{\lambda}{\mu} + 1\right)t \end{aligned}$$

The average expected data transfer time, $f(t)$, is,

$$\begin{aligned} f(t) &= P_c \cdot f_c(t) + P_D \cdot f_d(t) \\ &= \frac{\mu}{\lambda+\mu} \cdot \left(\frac{\lambda}{\mu}+1\right)t + \frac{\lambda}{\lambda+\mu} \cdot \left\{\frac{1}{\mu} + \left(\frac{\lambda}{\mu}+1\right)t\right\} \end{aligned}$$

If $f(t)$ is a function of *t*, *λ*, and *μ*,

$$f(t,\lambda,\mu) = \left(\frac{\lambda}{\mu}+1\right)t + \frac{\lambda}{\lambda+\mu} \cdot \frac{1}{\mu}$$

Let t_{job}, t_{in} and t_{out} denotes job processing time on a mobile node, expected time required to transfer the input data to mobile node, and expected time required to transfer the output data from mobile node, respectively. The response time of single job execution on a mobile node *i* can be represented as follows:

$$g_i(t_{in}, t_{job}, t_{out}) = f_i(t_{in}) + t_{job} + f_i(t_{out}) \quad \ldots\ldots \ (1)$$

Figure 5 shows the algorithm to select the participating mobile nodes in the computing. In step 3 and 4, nodes are listed in an increasing order of response time using equation (1). In step 5, $g_i - g_{i-1}$ means expected additional waiting time, caused by including i^{th} node, after $(i-1)^{th}$ node return the result, while $\frac{W_{total}}{i-1} - \frac{W_{total}}{i}$ means the expected reduced execution time achieved by including i^{th} node in the computing. Thus, i^{th} node is included if it turns out to reduce overall response time comparing to including up to $(i-1)^{th}$ node.

Figure 6 demonstrates the effects of the algorithm. As we include more nodes in the grid computing, each node performs fewer amounts of jobs, thus job processing time per mobile node decreases. However, as we include more nodes, the worst node is likely to be in disconnection state long and it causes increased waiting delay for input and output data transfer. Thus, it is needed to find the optimal number of participating nodes from tradeoff between deficiency and efficiency.

```
Step 1. $N$ := number of mobile nodes in the mobile grid resource pool.
Step 2. $W_{total}$ := total execution time of jobs to be processed.
Step 3. Obtain the expected response time of all mobile nodes using the equation (1).
Step 4. List the mobile nodes in increasing order of response time.
Step 5. FOR (i=N; i>1; i--)
            if $\left[-(g_i - g_{i-1}) + \left(\frac{W_{total}}{i-1} - \frac{W_{total}}{i}\right) > 0\right]$
                break;
Step 6. Distributes the jobs to the nodes up to $i^{th}$ in the list
```

Fig. 5. Job scheduling algorithm for the mobile grid system

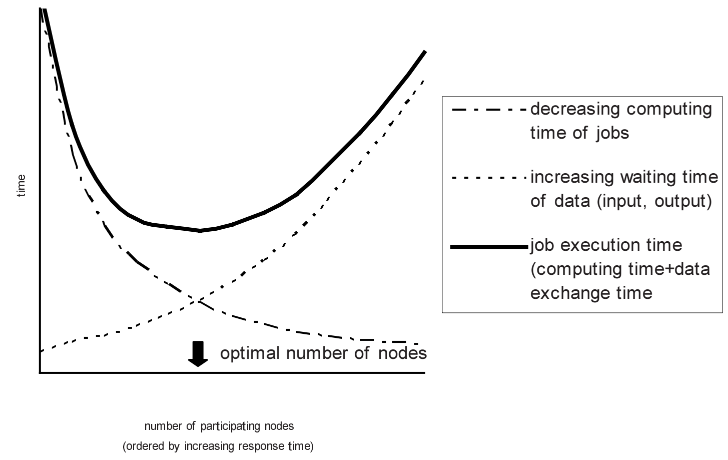

Fig. 6. Benefits and losses from increasing the participating nodes to the computing

5 Performance Evaluation

We evaluate the proposed job scheduling algorithm using mathematical analysis, simulations and experiments on the implemented prototype. We implement simple job queuing system as prototype of MQS. In the prototype, unreliable connectivity of mobile node is considered as usual behavior. Mobile nodes are repeatedly connected to and disconnected from the master system (MQS). When an available node appears (idle mobile node becomes connection state), the first job in the queue is distributed to the node and executed regardless of node's further connection state. After the job is completed, the node sends result data to master if it's in connection state, while it waits for the reconnection to the master when it's in disconnection state (e.g., out of communication range).

We assume that each mobile node in the mobile grid system has different value of λ (disconnection rate) and μ (reconnection rate). We generate the λ and μ value (uniformly random) of each mobile node between the fixed minimum and maximum of λ and μ. By differentiating the maximum and minimum value of λ and μ, we assume various mobile environments. We consider four different mobile environments with the low and high rates of disconnection (λ) and reconnection (μ). Assumed environments are stable (low λ, low μ), unstable (high λ, high μ), highly connective (low λ, high μ), and highly disconnective (high λ, low μ), as classified in [18]. The maximum and minimum value of λ and μ representing the environments are presented in Table 1.

Table 1. Assumed mobile environments

Mobile environments	Max λ	Min λ	Max μ	Min μ
Stable	0.003	0.001	0.003	0.001
Highly disconnective	0.027	0.009	0.003	0.001
Unstable	0.027	0.009	0.027	0.009
Highly connective	0.003	0.001	0.027	0.009

Table 2. Experimental environment

Mobile Node Specification (10 Nodes)	CPU	P4 1.6 GHz
	Memory	256 MB
	Network Connection	802.11b wireless LAN
Application	Blast (Bioinformatics)	
Workload amount	Number of protein sequences needed for 1000 seconds of processing	
Network disconnection & reconnection	Exponentially random value with respect to the λ and μ of mobile node	
Data	Input	Decomposed protein sequences (5 KB/*number of nodes*)
	Output	Information achieved from protein (200 KB/*number of nodes*)

Table 3. Parameters for mathematical analysis and simulations

W_{total} (total execution time of jobs)	*N* (Number of participating nodes)	t_{in} (time required for transferring input data)	t_{out} (time required for transferring output data
1000 sec.	10 nodes	1 sec.	1 sec.

In the simulation, the disconnection and reconnection of a mobile node is governed by Poisson process with the rate of λ and μ, respectively. We perform experiments on the implemented prototype. The experimental environment is presented in Table 2. We adopt a Bioinformatics application, Blast [19], for the experiments. Blast is the gene sequence analysis application comparing newly found protein sequences with well known sequence database and predicts the structure and functions of the newly found one. It needs very intensive processing cycles, thus grid is utilized for executing Blast in many Bioinformatics and grid projects. In the experiments, we analyze a number of sequences which need total 1,000 seconds to process on a single node. The sequences are divided according to the number of participating nodes, and then distributed to the nodes. We manually configured the disconnection and reconnection of mobile nodes in an exponentially random value with respect to the λ and μ in Table 1. Upon receiving the protein sequences and job related information (input data), the mobile nodes execute Blast and return the information about the protein (output data) to the master. Table 3 describes parameters used in the mathematical analysis and simulations. The parameters correspond to the experimental environment presented in Table 2.

Figure 7 describes the analytical results for four different mobile environments based on the analysis in previous section.

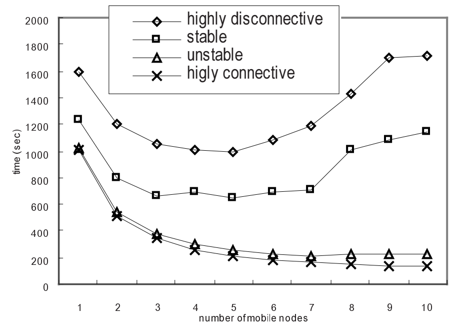

Fig. 7. Job execution time in four mobile environments (by mathematical analysis)

We use the turnaround time, which is the time required to complete a batch of tasks, as a performance metric. The x-axis represents the number of participating nodes to compute the jobs. The nodes are arranged in increasing order of expected response time (i.e., the first node is the best one). The y-axis is the turn around time. The turnaround time in highly disconnective environments is the longest among all, and that in highly connective environment is the shortest. The result in unstable environment is comparable to the highly connective case, and result of stable environment is in the middle of them. In unstable and highly connective environments, the mobile nodes remains in disconnected state shortly, thus short time period is consumed to wait to transfer the input and output data to and from the disconnected mobile nodes. On the other hand, longer time is needed to transfer the input and output data in highly disconnective, and stable environments.

As the number of participating nodes increases, the turnaround time in stable and highly disconnective environment decreases until including the nodes close to the critical point. However, it does not decrease any more as the number of participating nodes grows; instead it increases. Consequently, including nodes more than the x value of the critical point makes the turnaround time grow. Thus, the number of participating nodes should be limited to the x value of the critical point. The results from unstable and highly connective environment show different aspect. The turnaround time decreases steadily in these environments. This is because the nodes are not in the disconnection state for a long time so that having more nodes participate to the computing always produces better performance. Yet, the critical point may appear as the number of nodes grows.

Along with the analytical result, we demonstrate the simulation and experimental results of each assumed environment in Figures 8-11. In Figure 8 and 9, you can see that the turnaround time of job could be the shortest when participating node is limited to a certain number. The number of nodes which produces the best performance in real experiment is very close to the optimal number (pointed by an arrow) determined by mathematical analysis and simulation which are based on the proposed scheduling algorithm. In Figure 10 and 11, results of experiments, simulation and mathematical analysis show that including all available nodes produces the best performance. The reason is the same to the case of above analytical result.

To sum up, we are able to assure that the proposed job scheduling algorithm is a viable approach to adapt in mobile environment where links are prone to frequent disconnections, especially when mobile nodes remain in disconnection state for longer duration.

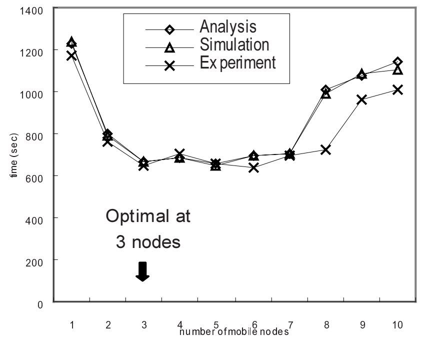

Fig. 8. Job execution time in a stable environment (By analysis, optimal number of nodes is 3)

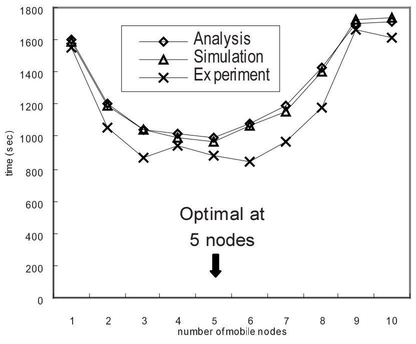

Fig. 9. Job execution time in a highly disconnective environment (By analysis, optimal number of nodes is 5)

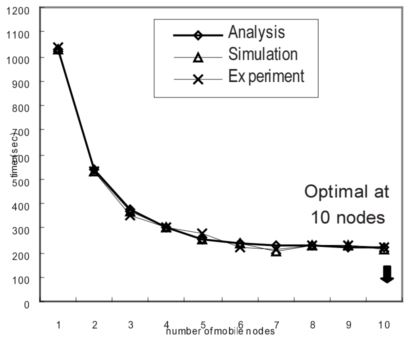

Fig. 10. Job execution time in an unstable environment (By analysis, optimal number of nodes is 10)

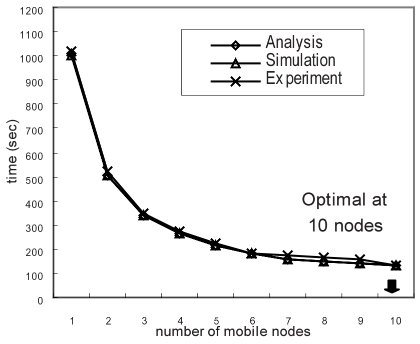

Fig. 11. Job execution time in a highly connective environment (By analysis, optimal number of nodes is 10)

6 Conclusion and Future Works

In this paper, we discuss on the issue of integrating mobile devices into grid. We propose our own view of the mobile/grid integrated system where a gateway links the static grid sites to the group of mobile devices, thus mobile users can make use of static grid resources, and also provide their mobile devices as grid resources. We present the newly emerging technical issues for realizing this mobile grid system, and particularly focus on the job scheduling algorithm to achieve more reliable performance.

We elaborate overcoming an unreliable connectivity of mobile environment thorough proposed scheduling algorithm in this paper. However, there are still challenging problems such as limited energy, device heterogeneity, security, and so on. We will tackle on these issues in future works and develop a prototype of mobile grid system.

References

[1] I. Foster, C. Kesselman and S. Tuecke. “The Anatomy of the Grid: Enabling Scalable Virtual Organizations,” International J. Supercomputer Applications, 15(3), 2001.

[2] M. Satyanarayanan. "Fundamental Challenges in Mobile Computing," In Proceedings of the fifteenth annual ACM Symposium on Principles of Distributed Computing, Philadelphia, Pennsylvania, 1996.

[3] T. Phan, L. Huang, and C. Dulan. "Challenge: Integrating Mobile Wireless Devices Into the Computational Grid," In Proceedings of the 8th ACM International Conference on Mobile Computing and Networking (MobiCom '02), September 25–27, 2002, in Atlanta, GA.

[4] Clarke, Brian and Marty Humphrey. "Beyond the 'Device as Portal': Meeting the Requirements of Wireless and Mobile Devices in the Legion Grid Computing System," In Proceedings of the Parallel and Distributed Computing Issues in Wireless Networks and Mobile Computing at the International Parallel and Distributed Processing Symposium. IEEE Press, 2002.

[5] Jonathan Ledlie, Jeff Shneidman, Margo Seltzer, John Huth. “Scooped, Again,” Proceedings of Second International Workshop on Peer-to-Peer Systems (IPTPS’03), 20–21 February 2003 in Berkeley, CA, USA.

[6] Bryce Wilcox-O’Hearn. “Experiences Deploying a Large-Scale Emergent Network,” Proceedings of Second International Workshop on Peer-to-Peer Systems (IPTPS’02), 7–8 March 2002 in Cambridge, MA, USA.

[7] SETI@home. http://setiathome.ssl.berkeley.edu, March 2001.

[8] Derrick Kondo, Henri Casanova, Eric Wing, Francine Berman. “Models and Scheduling Mechanisms for Global Computing Applications,” Proceedings of IPDPS 2002, April 15–19, 2002, Fort Lauderdale, California

[9] F. Berman and R. Wolski. “The AppLes project: A status report,” Proceedings of the 8th NEC Research Symposium, Berlin, Germany, May 1997.

[10] The Condor Project. http://www.cs.wisc.edu/condor

[11] D. Abramson, J. Giddy, I. Foster, and L. Kotler. “High Performance Parametric Modeling with Nimrod/G: Killer Application for the Global Grid?,” In Proceedings of the International Parallel and Distributed Processing Symposium, May 2000.
[12] Sang-Min Park and Jai-Hoon Kim. "Chameleon: A Resource Scheduler in a Data Grid Environment," 2003 IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID'2003), Tokyo, Japan, May 2003.
[13] OpenPBS. http://www.openpbs.org
[14] LSF. http://www.platform.com/products/LSF/
[15] The Globus Project. http://www.globus.org
[16] K. Czajkowski, I. Foster, N. Karonis, C. Kesselman, S. Martin, W. Smith, S. Tuecke. “A Resource Management Architecture for Metacomputing Systems,” Proc. IPPS/SPDP '98 Workshop on Job Scheduling Strategies for Parallel Processing, pg. 62–82, 1998.
[17] J.-P Goux, S. Kulkarni, J. T. Linderoth, and M. E. Yoder. "An Enabling Framework for Master-Worker Applications on the Computational Grid," Proceedings of the Ninth IEEE International Symposium on High Performance Distributed Computing, 2000.
[18] S. Radhakrishnan, N. S. V. Rao G. Racherla, C. N. Sekharan, and S. G. Batsell, "DST - a routing protocol for ad hoc networks using distributed spanning trees," IEEE Wireless Communications and Networking Conference, pp. 100–104, 1999.
[19] Cynthia Gibas. “Developing Bioinformatics Computer Skills,” O’REILLY, April 2001.