Effective Distributed Parallel Scheduling Methodology for Mobile Cloud computing

Hiromasa Yamauchi† Koji Kurihara† Toshiya Otomo† Yuta Teranishi‡ Takahisa Suzuki† Koichiro Yamashita†

†Processor Solution Development Lab.
Fujitsu Laboratories Ltd.
1-1, Kamikodanaka 4-chome,
Nakahara-ku, Kawasaki 211-8588 Japan
Tel : +81-44-754-2781
Fax : +81-44-754-2744
e-mail : { yamauchi.h, kouji3211,
otomo.toshiya, suzuki.takahisa,
yamashita.ko-05}@jp.fujitsu.com

‡Engineering Dept.I
Fujitsu Kyushu Network Technologies
Limited
Fujitsu Kyushu R&D Center Bldg, 2-1
Momochihama 2-chome, Sawara-ku,
Fukuoka 814-8588, Japan
Tel : +81-92-852-8159
Fax : +81-92-852-3241
e-mail : teranishi.yuta@jp.fujitsu.com

Abstract – There is a category of the device such as mobile phones and the sensor devices. If each device is considered as a node, these devices will be considered to be a distributed parallel processing system. It is defined as "Mobile Cloud computing (MC)". The collaborated processing between mobile phones, calculation by sensor devices, etc. are practical usage of MC. This MC differs from traditional parallel processing among servers, mainframe or HPC in respect of dynamic fluctuation of battery power and mobile network quality. We propose a distributed parallel scheduling methodology for MC and developed a simulator to analyze these characteristics and the bottleneck of MC.

I. INTRODUCTION

The performance of mobile phone is improving in this decade, as shown in Figure 1 [1][2]. The speed of mobile network is also improving and it will be faster in LTE-Advanced generation [3]. In the past, mobile phones had few functions, calling and SMS. However, currently, mobile phones have enormous functions, web browser and movie player, maps, and so on.

There is a new system that embedded devices collaborate with each other with network such as mobile phones and sensor network devices [4]. There are some use-cases, for example, the system for the agriculture [5], as shown in Figure 2. Another use-case is Inter-Vehicle Communication (IVR) [6]. Each car communicates with other cars to collect various information about traffic jams and the following distance, as shown in Figure 3. In the sensor network system, the performance and battery power of each device are very small. On the other hand, the total number of devices is large. In this network, each device collects some information and shares a large amount of information among all devices.

There is a new idea of a calculation model, MC [1]. It is one of the new distributed parallel processing systems that handles each embedded device as a node, likes grid computing by servers. Devices can share not only the user local data and process but also global data and process. For example, cars also can control themselves and prevent any traffic jams and troubles. Furthermore, cars might drive themselves.

However, MC differs from traditional parallel processing. The characteristics of MC are listed below.

- Due to the characteristics of mobile devices, dynamic change of the total number of devices
- Due to the characteristics of diversity of specification, the variability of a performance

In MC, they have to be considered. In this paper, we propose a distributed parallel scheduling methodology for MC and developed a simulator to analyze these characteristics and the bottleneck of MC.

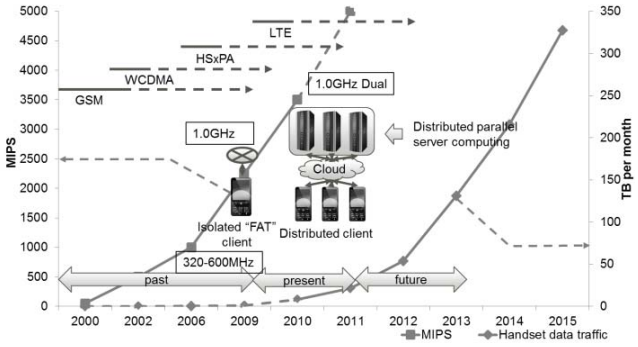

Figure 1 The trace of evolution of mobile phone field

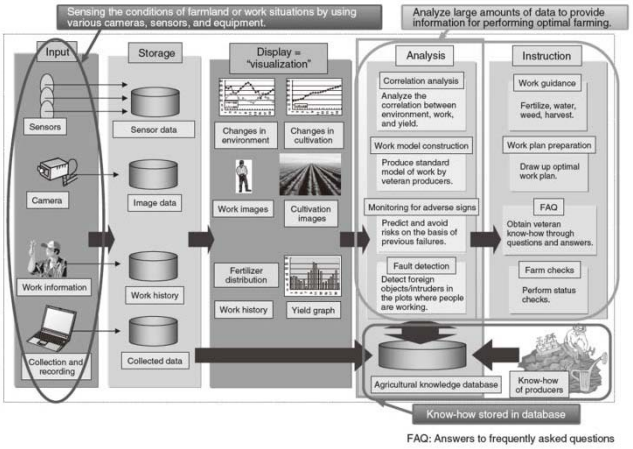

Figure 2 Sensor Network System for agriculture

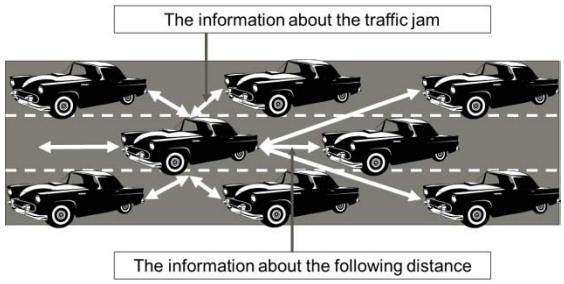

Figure 3 Inter-vehicle communication

II. CHALLENGES

In MC, following 3 problems should be solved.

1. Running out of battery power
2. Device appearance and disappearance due to dynamic fluctuation of the network quality
3. Dynamic heterogeneous load imbalance under these conditions

Because target of traditional parallel processing is server or mainframe or HPC, consideration for these issues is not enough. These problems are derived from characteristics of mobile devices. In the sensor network, the characteristics and problems of each device are different. It is shown in Table 1. However, it is important to consider a lot of characteristics and problems to apply MC to a variety of kinds of devices.

II-1. Running out of battery power

Basically mobile devices are powered by a battery, for example mobile phones, body sensors. Because a battery capacity is limited, a battery life is limited. A battery life depends on the application. The battery level is declining slowly during the execution of light weight applications and it is declining rapidly during the execution of heavy weight applications as shown in Figure 4 [8] and Figure 5. If the battery level becomes empty during the execution, the system can't keep on executing. If a battery of a device that is assigned to threads of application is going to run out, these threads must be migrated to other devices before running out of battery power to retrieve the data under execution.

Table 1 The characteristics of several kind of devices

	(A)Mobile Phone	*(B)Automotive*	*(C)Sensors in the field*
CPU performance	*Large*	*Small – Large*	*Small*
Battery power	*Small*	*Large(≒∞)*	*Small or nothing*
Data rates	*Large*	*Small – Large*	*Small*
The number of thread	*Large*	*Large*	*Small*

II-2. Device appearance and disappearance due to dynamic fluctuation of the network quality

Mobile devices have high mobility and the fluctuation of the network quality due to the movement as shown in Figure 5. The important components of the network quality are signal strength and network data rate and so on. If the signal strength is too weak, mobile devices can't catch the signal and communicate with each other. And the system can't keep on executing. In this situation, a thread of application assigned a device that is difficult to keep a connection must be migrated to other device before disappearance of source device of migration from the system.

Network data rate relates the performance improvement of the system by MC. The performance improvement is along with Amdahl's law.

$$T(N) = T(1) \times \left(1 - P + \frac{P}{N}\right) + \tau \ . \qquad \text{--- (1)}$$

(T(N):Execution time with N devices,
T(1):Execution time with a single device,
P:The ratio of parallel region,
N:A total of number of devices,
τ: Overheads of parallel processing)

The overheads consist of the scheduling time and the communication time and etc. Scheduling time is the time to select the device to assign threads by the master device.

$$\text{communication time} = \frac{\text{memory footprint}}{\text{network data rate}} \ . \qquad \text{--- (2)}$$

If network data rate is low, the communication overheads become large. Figure 7 shows the speedup ratio with several overheads. As shown in Figure 7, decreasing communication overheads are absolutely imperative to improve the performance. In order to assign threads of applications to devices connected by high network data rate, network data rate must be observed periodically. The influence of performance by the scheduling overheads is described later. In traditional parallel processing among PC and server and HPC connected by LAN cable, although the fluctuation of the network quality is considered, it fluctuates more gradually than in MC and there are few risks of disconnection of the network.

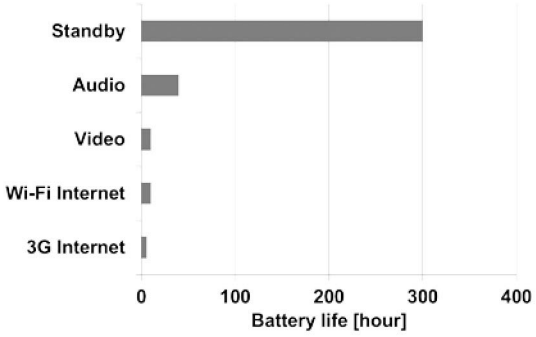

Figure 4 Maximum battery life for several applications

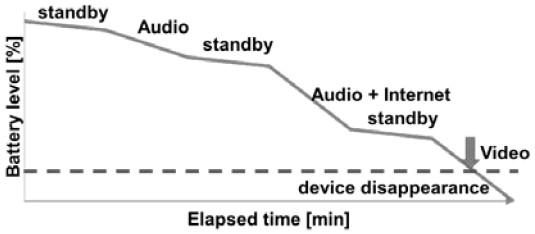

Figure 5 Battery consumption along with use-cases

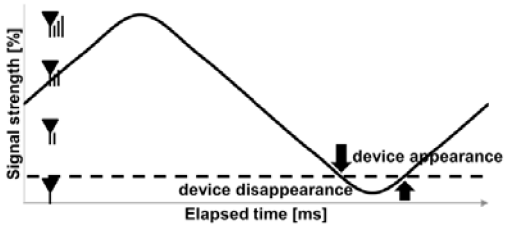

Figure 6 Fluctuation of the network quality

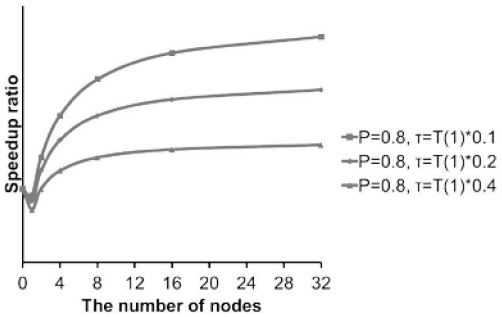

Figure 7 Speedup ratio along with Amdahl's law

II-3. Dynamic heterogeneous load imbalance under conditions, II-1 and II-2

Basically, because the specification of each device is different, the performance of each device is different. Some devices, for example the sensor network devices are single function devices. Automotive has a lot of functions and a lot of micro controller unit (MCU) for the sensors. Each MCU is in charge of a single function for the reliability. On the other hand, mobile phones are multiple functions computers, there are several running applications. Load balance is determined by computing resource of each device.

$$\text{Idle} = 1 - \text{utilization},$$
$$\text{computing resource} = \text{the performance of each device} \times \text{Idle} \quad . \qquad \text{--- (3)}$$

In MC, in order to balance the load of each device, the ratio of utilization also must be observed periodically. Because the performance is along with Amdahl's law, enlarging tasks for parallel processing and increasing parallel region are inevitable in order to obtain the high performance improvement. And so are increasing a total of number of devices and decreasing overheads.. In order to enlarge tasks, the extraction of threads is important. Increasing a total of number of devices has good effect for the performance if parallel region is enough. However, even if a total of number of devices is enough, large numbers of devices are not effective in small parallel region and it makes side effect. It makes scheduling overheads larger.

$$\text{scheduling time} = \text{observing overheads of each devices} \times \text{a total of number of devices} \quad . \qquad \text{--- (4)}$$

In order to decrease the overheads, decreasing a total number of devices in small parallel region and selecting the threads with small data transfer are effective. Determining the appropriate number of devices along with the situation is important. In MC, these problems such as running out of a battery power and the fluctuation of the network quality and dynamic fluctuation of load imbalance must be solved.

III. RELATED WORK

Mobile devices can be connected with each other by adhoc network. The practical application of mobile phones connected by adboc network has been developed already [9]. But there is no consideration of key issues for MC. Parallel processing among PC and server, HPC has been studied [10] and has been used in practical applications. Several scheduling algorithms were introduced [11][12][13]. Some algorithms consider dynamic fluctuation of computing resource and network bandwidth [14][15][16]. However, since the evaluation environment in those papers is PC and server and HPC, the device disappearance due to a battery and the network quality are not considered. These traditional scheduling algorithms are not appropriate for the proposed method.

IV. PROPOSED SCHEDULING METHODOLOGY

In MC, there is a master device and multiple slave devices in one system. The key parameters such as the battery level and the network quality and the load balance of all connected slave devices, are observed by the master device. If these parameters of some devices are not enough to continue parallel processing, the master device retrieves threads from slave devices and selects other slave devices to dispatch them again. We propose the scheduling methodology including this mechanism. They are managed as below.

IV-1. Managing the battery level and loading balance

The battery level is declining along with the elapsed time from the charge level and the ratio of utilization of each device. The master device predicts when the battery level of

each slave device is not enough to continue parallel processing. On mobile phones, users launch some application at random on mobile phones. However, the master device can predict which applications will be activated with a log of application activation as shown in Figure 8. On automotive, the master device can predict it with driving log, too. Or the master device periodically determines whether the battery level is lower than threshold or not. In this simulator, the battery level and the ratio idle are defined as below.

$$Battery\ level_{n+1} = \text{Battery level}_n - \text{Interval time}^{*1} * \text{power consumption rate}^{*2} * \text{ratio of utilizaion}, \quad \text{--- (5)}$$

$$\text{Ratio of idle} = 1 - \text{Ratio of utilizaion} * (1 \pm \frac{random()^{*3}}{INT\_MAX} * \text{Fluctuation range}^{*4}) \ . \quad \text{--- (6)}$$

*1 : It is an interval of observing slave devices by the master device
*2 : This parameter is based on Figure 4
*3 : This function returns from 0 to INT_MAX
*4 : It is range of a fluctuation of a parameter in one interval

IV-2. Managing the network quality

Because devices move with users at random, the network quality fluctuates dynamically. However, the master device predicts the fluctuation of the signal strength with the latest value and determines the signal strength will be enough or not as shown in Figure 9. Or the master device periodically determines whether the signal strength is lower than the threshold or not. In this simulator, the signal strength and the network data rate are defined as below.

$$Signal\ strength_{n+1} = Signal\ strength_n * (1 \pm \frac{random()}{INT\_MAX} * \text{Fluctuation range}) \ , \quad \text{--- (7)}$$

$$Data\ rate_{n+1} = Data\ rate_n * (1 \pm \frac{random()}{INT\_MAX} * \text{Fluctuation range}) \ . \quad \text{--- (8)}$$

To continue parallel processing, the master device collects some information such as of the battery level, the network quality, the ratio utilization, application activation logs etc. periodically. A total of number of device is defined as below.

$$\text{A total of number of devices}_{n+1} = \text{A total of number of devices}_n + 1$$
(if Ratio of utilization $<$ Threshold of ratio of utilizaion^{*4}, if Signal strength $<$ Threshold of network quality^{*5})
A total of number of devicesA total of number of devices$_{n+1}$ $=$ A total of number of devices$_n$ -1
(if The battery level $>$ Threshold of battery^{*6}, if Ratio of utilization $>$ Threshold of ratio of utilizaion, if Signal strength $<$ Threshold of network quality) . --- (9)

*5 : This threshold is to identify whether the ratio of idle is enough to distributed parallel processing or not.
*6 : This threshold is to identify whether a signal is strong enough to distributed parallel processing or not.
*7 : This threshold is to identify whether the battery level is enough to distributed parallel processing or not.

In MC, when a new application is launched, the master device calculates an execution time and communication time on each device and selects one device that can execute it with minimum time. The calculation formula considering described problems is shown below.

$$\text{MinimumTime} = \min_{i\in\{0,\ldots,\text{SlaveNum}-1\}}(\text{ExecutionTime on Slave}_i + \text{CommunicationOverheads on Slave}_i) = \min_{i\in\{0,\ldots,\text{SlaveNum}-1\}}(\frac{\text{Occupation time of new thread}^{*7}}{\text{Clock frequency of Slave}_i * \text{the ratio of idle of Slave}_i} + \frac{\text{memory footprint of new thread}}{\text{network data rate of Slave}_i}) \ . \quad \text{--- (10)}$$

*8 : It is an occupation time on CPU of standard clock frequency
The master device compares with minimum time and sequential execution time on master device. If minimum time is smaller than sequential execution time, the master device dispatches this thread to the selected slave device. If not, the master device starts execution of this thread by itself.

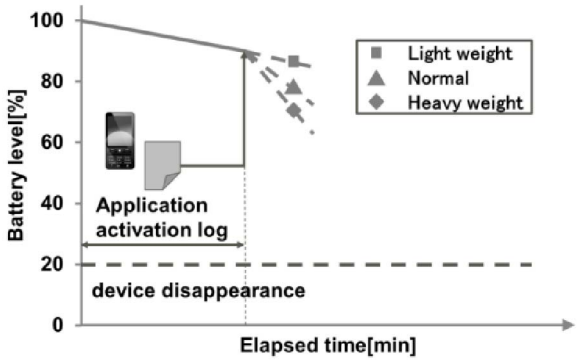

Figure 8 Battery consumption with application activation

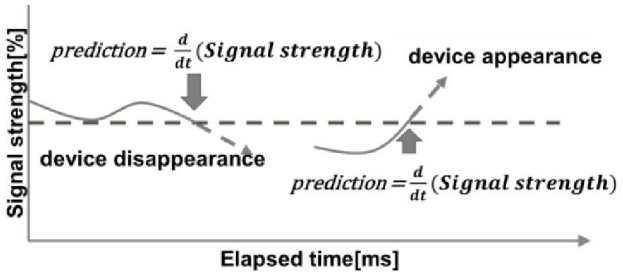

Figure 9 Prediction of signal strength

V. SIMULATION

We evaluated the performance improvement and the scalability of MC by a simulation. Figure 10 shows the image of the simulation model of this experiment. In this simulator, several parameters dealt in formula (5) – (10) are set to as Table 2, and initial values of these parameters are set to as Table 3. The conditions for measurement and comparison are listed in Table 4. In these tables, A means mobile phone, B means automotive and C means sensor. These parameters are setup for the use-cases. In these cases, target systems are the future model of mobile phone and automotive and the sensor network devices. Mobile phones connected to each other with LTE-A network communicate a large amount of data and automotive and the sensor devices connected to each other with ZigBee communicate very small data. The battery level of automotive is always full. The idle ratio of automotive and sensor is always 100%. The result is shown in Figure 11.

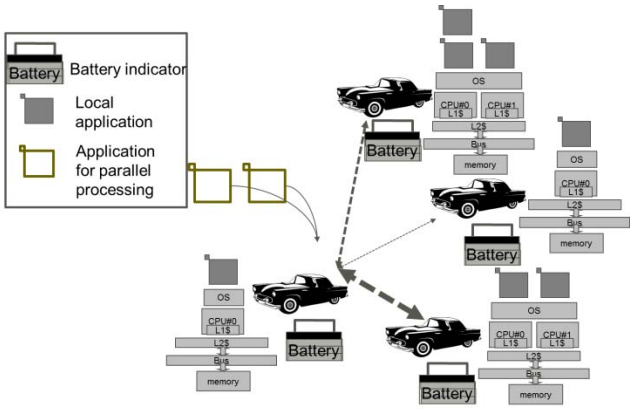

Figure 10 Simulation Model

Table 2 Parameter set of simulator

System	Maximum number of device	128	
HW	Clock frequency	A	2.5GHz quad
		B	10MHz
		C	10MHz
	Threshold of utilization	20%	
	Threshold of the battery level	20%	
SW	The number of thread	200	
	Occupation time	A	100sec(standard clock frequency = 1GHz)
		B	100msec(standard clock frequency = 10MHz)
		C	100msec(standard clock frequency = 10MHz)
	Memory footprint of thread	A	100MB
		B	100Byte
		C	100Byte
Network	Network data rate	A	1Mbps- 1Gbp
		B	20Kbps-250Kbps
		C	20Kbps-250Kbps
	Signal strength of network	20%	
Simulator	Fluctuation range	0.2	
	Interval time	1sec	

Table 3 Initial value

Battery level	100%	
Signal strength	100%	
Data rate	A	1Gbps
	B	20Kbps
	C	20Kbps
Ratio of idle	A	50%
	B	100%
	C	100%

Table 4 Conditions for measurement and comparison

Measurement	Times	10
	Handling	average excepting maximum and minimum
Comparison target	A	2.5GHz single core system, ratio of idle = 100%
	B	10MHz single core, ratio of idle = 100%
	C	10MHz single core, ratio of idle = 100%

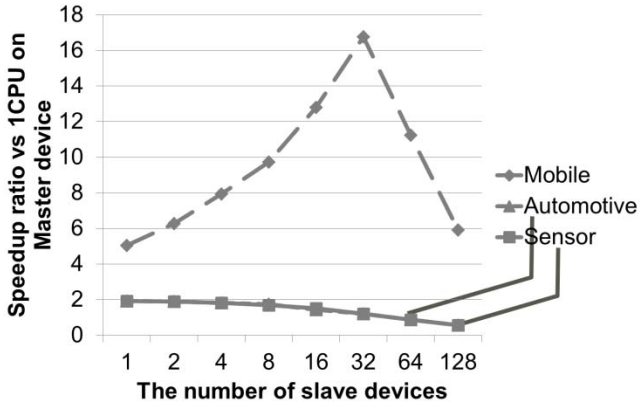

Figure 11 Result of simulation

VI. CONSIDERATION

As shown in Figure 11, MC can improve the performance on mobile phones. Because the occupation time for mobile phones is large, the scheduling overheads and the communication overheads can be concealed. As also shown in Figure 11, the effect of this method is declining with 64 devices and 128 devices. There might be large overheads because they are proportional to a total number of devices as described in II-3. Although the large number of devices is better in order to improve the performance, it makes their overheads large.

On the other hand, the results on Automotive and the sensor network devices are almost same and MC can't improve them. Because the performance of each device is low, overheads are relatively larger than overheads for mobile phones. It must be the bottleneck of the system. To apply MC for the low performance devices, the technique to make overheads small must be imperative. An automatic determination of an appropriate number of devices for scheduling might be one of the solutions.

VII. CONCLUSION

In this paper, we proposed a distributed parallel scheduling methodology for MC. In this simulation, we verified the performance improvement by this method and looked for a bottleneck of the system.

As future tasks, we will attempt to enhance this scheduling methodology. One of them is determining the appropriate number of devices automatically in order to improve the performance for the low power devices. And, in this paper, the total power consumption of the systems is not mentioned. We will attempt to develop another enhancement to decrease the power consumption of the systems.

Acknowledgements

Special thanks to my colleagues and supervisors for their valuable discussions and helpful supports.

References

[1] Patrick Blouet, "Mobile Cloud Computing trends and challenges", 11th International Forum on Embedded MPSoC and Multicore, http://www.mpsoc-forum.org/slides/3.3-Blouet.pdf, 2011

[2] Guildford, Surrey, Coda Research Consultancy, "Mobile internet and handsets in the US: A comprehensive assessment, with forecasts to 2015"

[3] http://www.3gpp.org/lte-advanced

[4] Jan M. Rabaey, "THE SWARM AT THE EDGE OF THE CLOUD - A NEW FACE OF WIRELESS", http://bwrc.eecs.berkeley.edu/People/Faculty/jan/presentations/SwarmKeynote%20VLSI11Final.pdf

[5] Hori, M.; Kawashima, E; Yamazaki, T;, "Application of Cloud Computing to Agriculture and Prospects in Other Fields", FUJITSU Sci. Tech.J., Vol.46, No.4 (October 2010)

[6] Saito, M.; Tsukamoto, J.; Umedu, T.; Higashino, T., "Design and Evaluation of Inter-Vehicle Dissemination Protocol for Propagation of Preceding Traffic Information", IEEE Transactions on Intelligent Transportation Systems, Vol. 8, No.3, pp.379-390, 2007.

[7] http://amorg.aut.bme.hu/projects/symtorrent

[8] http://www.apple.com/iphone/specs.html

[9] Ughetti, M.; Trucco, T.; Gotta, D., "Development of agent-based, peer-to-peer mobile application on ANDROID with JADE", Mobile Ubiquitous Computing, Systems, Services and Technologies, 2008. UBICOMM '08. The Second International Conference on, pp. 287 – 294, 2008

[10] Kasahara, H.; Obata, M.; Ishizaka, K.; Kimura, K.; Kaminaga, H.; Nakano, H.; Nagasawa, K.; Murai, A.; Itagaki, H.; Shirako, J., "Multigrain automatic parallelization in Japanese Millennium Project IT21. Advanced Parallelizing Compiler"', Parallel Computing in Electrical Engineering, 2002. PARELEC '02. Proceedings. International Conference on, pp.105-111, 2002

[11] Fernandez-Baca, D., "Allocating Modules to Processors in a Distributed System", Software Engineering, IEEE Transactions on, pp.1427-1463, 1989

[12] Kant Soni, V.; Sharma, R.; Kumar Mishra, M., "An Analysis of Various Job Scheduling Strategies in Grid Computing", Signal Processing Systems (ICSPS), 2010 2nd International Conference on, pp.V2-162 – V2-166, 2010

[13] Nithiapidary Muthuvelu, Junyang Liu, Nay Lin Soe, Srikumar Venugopal, Anthony Sulistio, Rajkumar Buyya, "A Dynamic Job Grouping-Based Scheduling for Deploying Applications with Fine-Grained Tasks on Global Grids", ACSW Frontiers '05: Proceedings of the 2005 Australasian workshop on Grid computing and e-research - Volume 44 , Volume 44, 2005

[14] Wu-Chun Chung, Ruay-Shiung Chang, "A new mechanism for resource monitoring in Grid computing", Future Generation Computer Systems, Volume 25, pp.1-7, 2009.

[15] Vladimir V. Korkhov, Jakub T. Moscicki, Valeria V. Krzhizhanovskaya, "Dynamic workload balancing of parallel applications with user-level scheduling on the Grid", Future Generation Computer Systems, Volume 25 Issue 1, 2009.

[16] Quan Liu, Yeqing Liao, "Grouping-based Fine-grained Job Scheduling in Grid Computing", Education Technology and Computer Science, 2009. ETCS '09. First International Workshop on, pp. 556-559, 2009