International Conference

Parallel and Distributed Computing Systems

PDCS 2014

Collection of scientific papers

March 4-6, 2014
Kharkiv, Ukraine
National Academy of Sciences of Ukraine, Institute for Scintillation Materials NAS of Ukraine and V. Glushkov Institute of Cybernetics NAS of Ukraine are pleased to announce the International Conference on Parallel and Distributed Computing Systems (PDCS 2014) being held March 4-6, 2014 in Kharkiv, Ukraine.

http://hpc-ua.org/pdcs-14/

All materials are given in the authors’ editions.

Editor-in-Chief: Cherepynets V.

ISMA Publishing House

Program Committee Co-Chairs

- Academician Boris Grynyov, Director of ISMA NASU, Ukraine
- Academician Ivan Sergienko, Director of IC NASU, Ukraine

Program Committee Members

- Dr. Andrey Aksenov, Tesis LTD, Russia
- Dr. Peter Berczik, National Astronomical Observatory, China
- Dr. Leonid Bilous, ILTPE NASU, Ukraine
- Dr. Maksym Bilous, IC NASU, Ukraine
- Dr. Oleksandr Dyomin, ISMA NASU, Ukraine
- Prof. Dr. Andreas Glatz, Argonne National Laboratory, USA
- Dr. Andrii Golovynskyi, IC NASU, Ukraine
- Dr. Oleksii Grechanovsky, Institute of Geochemistry, Mineralogy and Ore Formation NASU, Ukraine
- Dr. Roman Iakymchuk, Institut Mines-Télécom – Télécom SudParis, Evry, France
- Dr. Ruslan Iushchenko, IC NASU, Ukraine
- Prof. Dr. Oleksandr Khimich, IC NASU, Ukraine
- Dr. Oleksii Koshulko, IC NASU, Ukraine
- Dr. Dmitry Kuzmin, Syberian Federal University, Krasnoyarsk, Russia
- Dr. Andrii Malenko, IC NASU, Ukraine
- Prof. Hans Meuer, ISC General Chair, University of Mannheim, Germany
- Prof. Sergii Minukhin, KNUE, Ukraine
- Prof. Volodymyr Opanasenko, IC NASU, Ukraine
- Prof. Oleksandr Palagin, IC NASU, Ukraine
- Dr. Oleksandr Rozhenko, Institute of Organic Chemistry NAS, Ukraine
- Dr. Oleksandr Sobolev, ISMA NASU, Ukraine
- Dr. Vadim Tulchinsky, IC NASU, Ukraine
- Dr. Sergiy Yershov, IC NASU, Ukraine
- Prof. Sergii Znakhur, KNUE, Ukraine

Organizing Committee

- Head of Organizing committee Dr. Petro Stadnik, Acting Director of ISMA NASU
- Sergii Barannik, ISMA NASU
- Oleksandr Dyomin, ISMA NASU
- Andrii Golovynskyi, IC NASU
- Andrii Malenko, IC NASU
- Valentyna Cherepynets, IC NASU
## CONTENTS

1. Akhmedov D., Yelubayev S., Abdoldina F., Bopeyev T., Muratov D.  
   Efficiency of application the liquid cooling system in a personal hybrid computing system  
   based on graphics processors

2. Anjomshoa M., Salleh M.  
   Overview on Clouds@home: Virtualization Mechanism for Volunteer Computing

3. Bantikyan H.  
   CUDA Based Implementation of 2-D Discrete Haar Wavelet Transformation

   Polyakov V., Rubin Y., Smirnov S., Stepanian S., Torgonin E., Chagovets V.  
   Grid Infrastructure of B. Verkin ILTPE NASU and its practical use in scientific research

5. Belan V., Belous L., Polyakov V., Torgonin E., Huthaifa A.  
   Application of heterogeneous computing systems for description of processes in virtual reality systems

6. Grechanovsky A., Ogar T.  
   Molecular dynamics study of irradiation damage in LaPO4 and YbPO4

7. Khodakovskyy M., Zolot A., Merjvinskyy P.  
   Study of pattern identification processes in highly productive human synaptic networks

8. Komukhaev E., Cherepynets V.  
   Supercomputers: review of the new ratings and effective accelerators

9. Loutsky G., Mukhin V.  
   The Adaptive Routing Algorithm Taking Into Account the Trust Level to the Remote Nodes

10. Mashanko P., Rudenko P.  
    Component model of teaching system with open test form on a basis of grid technologies

11. Minukhin S.  
    Dynamic and static methods of task planning in distributed computing systems

12. Mukhin V., Kornaga Y.  
    Structural Organization of the Router for the Multi-channel Computer Systems

13. Nallasivam V., Kamalam G.  
    An Effective Cluster Score Job Scheduling Algorithm for Grid Computing

14. Rubio Bonilla D., Schubert L.  
    A Minimum Overhead Operating System Kernel for Better Scalability in HPC

15. Sharma A., Thakur J.  
    An Energy Efficient Network Life Time Enhancement Proposed Clustering Algorithm for  
    Wireless Sensor Networks
Efficiency of application the liquid cooling system in a personal hybrid computing system based on graphics processors

D. Akhmedov, S. Yelubayev, F. Abdoldina, T. Bopeyev, D. Muratov

Institute of space technique and technology, Almaty, Republic of Kazakhstan

akhmedov.d@istt.kz, s.elubaev@gmail.com, farida_mail@mail.ru, bopeyev.t@mail.ru, m_daulet@mail.ru

Abstract. Article presents a structural scheme of a personal hybrid computer system (PHCS) based on graphics processors with liquid cooling system. Here are presented the results of experiments to determine the effectiveness of the liquid cooling system. Results of measurements the level of acoustic noise from PHCS, temperature of heating the PHCS components in case of using the system of air and liquid cooling in normal mode and at maximum load are presented in this paper.

Keywords

Parallel computing, high-performance computing, hybrid computing system, GPU accelerator, CUDA technology, liquid cooling system.

1 Introduction

The problem of cooling computer components becomes more acute every year. Processors and GPU accelerators power are increases, and with it power consumption increasing and proportionally increases the heat dissipation capacity, which can reach 130-225 watts today. Manufacturers of central and graphics processors adopt new, more subtle technological process in order to keep processors heat generation, but it still not enough.

CPU and GPU modern cooling system is characterized by the performance and the level of acoustic noise. For personal computers its size is limited to 50 dBA, and for workstations and servers - 70 dBA. Using of personal computer is comfortable when the noise of cooling system is minimal.

The main advantage of the liquid cooling system, as compared to aerogenic cooling system is considerably large performance and a low noise level. Liquid cooling system has high performance because the thermal conductivity of the fluid in the five - to seven times higher than that of air, respectively, is less than its thermal resistance and higher heat flux. Another feature of the liquid cooling system is that the temperature of the cooled object is changed relatively slowly, due to the thermal inertia of the liquid [1].

2 Personal hybrid computing system with liquid cooling system

Personal hybrid computing system based on graphics processors ISTT HPC 2000 (figure 1) was developed at the Institute of Space Equipment and Technology [2].

Generally personal hybrid computer system is a high-performance personal computer, which may be located directly at the workplace of the user. The composition of personal hybrid computing system generally coincides with the composition of the personal computer, except that it includes specialized high-performance graphics processors Nvidia Tesla, actually realizing quick calculations. Powerful CPU and several GPU together form high-performance computing system, where intensive tasks can be shared between the processors, thereby providing a high parallelism and computational speed.

PHCS configuration includes the following components: two CPUs Intel Xeon E5-2690 2.9Ghz; motherboard Asus Z9PE-D8 WS; four GPUs Nvidia Tesla K20c; eight modules of RAM DDR3 16Gb 1600MHz.

The real performance of personal hybrid computing system based on four GPU Nvidia Tesla K20c in the Linpack test was 3353 gigaflops (3.4 teraflops) of double precision; it is 71.65% of peak performance.
In designing a hybrid personal computer system main difficulty is the calculation of scheme of powerful CPU and GPU optimum ratio, cooling system optimal power scheme.

To provide necessary cooling level with heavy use of graphics processors it have been developed two variants of PHCS cooling system.

In the first variant an effective air-cooling system has been developed, constructed in such a way that air flow can freely circulate between the computing nodes of personal computing system, preventing them from overheating. For this purpose it was applied the most efficient mechanical elements of cooling system - fans with low level of noise emissions as well, as a special type of system block case, allowing to some extent arbitrarily change the location of the cooling devices.

There are 4 pumping and 4 exhaust axial fans in the PHCS system block. Fans flow characteristics, sufficient to remove the heat generated by the elements of PHCS is calculated according to the procedure described in [3]. Figure 2 shows a scheme of the cooling system of PHCS system block case based on three GPUs.

Due to the fact that personal hybrid computing system can be used both in educational institutions and in permanent jobs, according to the established norms of the noise intensity, the maximum value of the noise generated by the computer system shall not exceed 50 dB.

<table>
<thead>
<tr>
<th>Workplace</th>
<th>Sound level, dBA</th>
<th>Sound pressure levels dB, in octave bands with center frequencies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Creative, scientific activity, training</td>
<td>50</td>
<td>86  71  61  54  49  45  42  40  38</td>
</tr>
<tr>
<td>Working places in industrial premises</td>
<td>80</td>
<td>107 95 87 82 78 75 73 71 69</td>
</tr>
</tbody>
</table>

Fig. 1. Personal hybrid computing system based on graphics processors

Fig. 2. PHCS system block case cooling scheme

Tab. 1. Norms of the noise intensity for commercial and industrial premises
Table 1 shows the intensity of the noise standards for commercial and industrial premises.

In the second variant it has been developed an effective liquid cooling system. Figure 3 shows a structural scheme of PHCS based on four GPUs with liquid cooling system.

The following is a description of the purpose of components placed inside the case of personal hybrid computing system based on graphics processors.

Component 1 - case. Size and structure of the case allow to locate the required amount of graphics processors and provide sufficient cooling level.

Component 2 - motherboard. Provides linking and control other devices, allow to accommodate a sufficient number of graphics processors to achieve high-performance systems.

Components 3-4 - CPUs with a clock frequency of 2.9 GHz, are managing all system components, have 40 lines PCI-Express, which allows the most use of the resources of GPUs.

Components 5-8 - graphics processors with peak performance of 1.17/3.52 teraflops of single and double precision, respectively, carry out high-performance computing.

Component 9 - random access memory with capacity of 128GB, is designed for temporary storage of processed data. High total amount of RAM allow reducing the number of references to the hard drives, thereby increasing productivity of system.

Components 10-13 - hard drives up to 1TB each, designed for long-term storage. Number of hard drives may differ depending on the requirements for the minimum amount of stored data.

Component 14 - blowing fan with a diameter 120 mm, provides cold air flows blowing into the case to the hot components of the system.

Component 15 - back exhaust fan with a diameter 120 mm, ensures the removal of hot air flows from the central processors.

Components 16-19 - top exhaust fans with a diameter 120 mm, provide output of hot air out of the system.
Component 20 - pump that provides forced circulation of the fluid in the liquid cooling system.
Components 21-22 - water blocks of CPUs, designed to remove heat from the central processor and its transmission to working fluid.
Components 23-26 - GPU water blocks, designed to remove heat from the GPU and its transfer to working fluid.
Component 27 - radiator, designed to dissipate the heat of the working fluid in the loop of liquid cooling system.
Component 28 - working fluid reservoir, provide compensation of thermal expansion of the liquid, increase the thermal inertia of the liquid cooling system and the convenience of filling and draining the working fluid.

3 The results of experiments

Noise emission measuring.
To determine the corrected sound power level it is used technical method for determining the corrected sound power level in free field over a reflecting plane according to State Standard 12.1.026-80 «Occupational safety standards system. Noise. Determination the noise characteristics of noise sources in the free field over a reflecting plane. Technical method» or method to determining the corrected sound power level in the reverberant room according to State Standard 12.1.027-80 «Occupational safety standards system. Noise. Determination of the noise characteristics of noise sources in reverberant room. Technical method». These methods allow to get the values of sound power levels with the maximum values of the standard deviation of sound power levels in accordance with State Standard 23941-79 «Noise. Methods for determination noise characteristics. General requirements». Requirements for the placement of personal hybrid computing system during the noise measurement: PHCS devices of floor type installed in the operation against the wall, would be located on a reflecting floor in front of a reflecting wall at a distance of 0.1 m from the wall. During measurements of PHCS device mode is installed in accordance with appendix 3 of State Standard 26329-84 «Computing machines and data processing system. Permissible noise levels of technical means and methods of their determination». Prior to testing, the device operated a sufficient time necessary to ensure a steady temperature.
Supporting parallelepiped was used as the measurement surface for personal hybrid cooling system. Its faces are located at a distance of measurements from them, equal to 1 m. Data from these measurements of sound power are shown in table 2. Number of measuring point corresponds to the measuring point of supporting parallelepiped shown in figure 4.

![Supporting parallelepiped](image)

**Fig. 4. Supporting parallelepiped**

<table>
<thead>
<tr>
<th>Measuring point number</th>
<th>PHSC with air cooling system</th>
<th>PHSC with liquid cooling system</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>In idle, Db</td>
<td>Under load, Db</td>
</tr>
<tr>
<td>1</td>
<td>50,4</td>
<td>65,4</td>
</tr>
<tr>
<td>2</td>
<td>50,3</td>
<td>64,6</td>
</tr>
<tr>
<td>3</td>
<td>51,4</td>
<td>64,4</td>
</tr>
<tr>
<td>4</td>
<td>50,7</td>
<td>62,3</td>
</tr>
<tr>
<td>5</td>
<td>50,5</td>
<td>62,6</td>
</tr>
<tr>
<td>6</td>
<td>52,1</td>
<td>64,2</td>
</tr>
</tbody>
</table>

The measurements of personal hybrid computing system sound power conducted during normal operation mode and maximum processors load. Experiments were conducted for PHCS with two variants of cooling system: air and liquid cooling system.
Analyzing the results we see that when using liquid cooling system at maximum processor load noise level is reduced by an average of 20%, which is 12 dB. When hybrid personal computing system work in normal mode, there isn't significant changes (Fig. 5).

**Fig. 5.** The results of measurements of sound power of personal hybrid computing system

Measurements of PHCS main components heating temperature also conducted in two modes: normal mode and at maximum load processors. Analyzing the results of temperature measurement shown in table 3 and figure 6, we see that the GPU heating temperature with a maximum load is reduced by almost twice, average by 46% and in normal mode by 30%. At the same time, the results of cooling for CPUs and motherboard do not have advantages that makes to think about the further improvement of the liquid cooling system.

<table>
<thead>
<tr>
<th>Component</th>
<th>PHSC with air cooling system</th>
<th>PHSC with liquid cooling system</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Temperature at maximum load, C</td>
<td>The temperature in the normal operation mode, C</td>
</tr>
<tr>
<td>CPU 1</td>
<td>83</td>
<td>47</td>
</tr>
<tr>
<td>CPU 2</td>
<td>74</td>
<td>41</td>
</tr>
<tr>
<td>GPU 1</td>
<td>81</td>
<td>42</td>
</tr>
<tr>
<td>GPU 2</td>
<td>84</td>
<td>42</td>
</tr>
<tr>
<td>GPU 3</td>
<td>83</td>
<td>42</td>
</tr>
<tr>
<td>GPU 4</td>
<td>84</td>
<td>42</td>
</tr>
<tr>
<td>System temperature (Motherboard)</td>
<td>59</td>
<td>34</td>
</tr>
</tbody>
</table>
4 Conclusion

The obtained results of measuring of personal hybrid computing system sound power showed a noise reduction when using liquid cooling system by 20% and reduction of operating temperature of GPUs almost twice, it proves that the liquid cooling system in a hybrid personal computing system is more efficient than air cooling system.

Reducing the level of noise even at maximum load of personal hybrid computing system with liquid cooling system to the requirements of a personal computer, makes the work with personal hybrid computing system comfortable and allows to place a high-performance computing system near the work area.

Efficient cooling of GPU reduces the probability of overheating and affects the lifetime beneficially.

References


Overview on Clouds@home: Virtualization Mechanism for Volunteer Computing

Mohammadfazel Anjomshoa¹, Mazleena Salleh²

¹Faculty of Computing, University technologi Malaysia, johor, Malaysia
²Faculty of Computing, University technologi Malaysia, johor, Malaysia

fazel_loyalboy@yahoo.com, mazleena@fsksm.utm.my

Abstract. The advent of Grid computing has enhanced the internet trend in the way that by using hardware resources of other computers, personal computers have become an unrivaled superpower. By utilizing several mechanisms such as downloading a screen saver large-scale scientific research systems are allowed to employ the resources of PCs during idle time. BOINC is a well-known volunteer computing middleware that establishes the powerful processing power from the resources offered by donors. BOINC has two limitations: first, it does not provide adequate security for users; second, applications that run on BOINC platform are confined to platform of systems (e.g., Windows, Linux, etc.). Virtualization is used for hiding physical resources of system from the operating system. On the one hand, by enabling Virtualization, applications run in a safe environment and so security will be enhanced. On the other hand, Virtualization solved the problem of platform-dependent on applications. Volunteer cloud system or Clouds@home has considered as a new form of cloud computing that builds its infrastructure from volunteer computers. With mixing cloud computing and volunteer computing we can build new computing paradigm that has both commercial and volunteer viewpoints. This study introduces a mechanism that enables Virtualization for applications that run on BOINC platform. This new computing paradigm has considered as primitive steps towards deploying Clouds@home.

Keywords

Clouds@home, volunteer clouds, cloud computing, grid computing, volunteer computing, virtualization, distributed systems.

1 Introduction

Technology is the combination of knowledge and working hard. When users want to accomplish something using special technology, they don’t want to know how it works. It means that users only want to employ technology without any expert skills. So technologies are coming to solve and ease our complex problems. Computing paradigm is one of the most concerns in a complex problem. In [1] they defined computing as an affording that uses computers both hardware and software to solve many computer related purposes; processing, scientific research, gathering information to extract some beneficial information and so on. Distributed computing is one of technology that uses to solve large and complex computational problems. Itemploys Distributed Systems to addresses computational problems. Distributed Systems are collection of computers that are physically and geographically distributed and connected to each other to solve a common problem.

Grid computing technology is one of branches in Distributed computing that enables collaborating and resource sharing where resources are geographically distributed and autonomous. In [2] they highlighted that resources for sharing are not only file whereas it can be processing power, software and applications, storage capacity, data and any other possible resources. Resources that work on Grid technology benefit from its features; more fast execution speed, interoperation of software, geographically distributed resources and so on. Desktop Grids (@homes) has been emerged as a type of Grid computing where the term computing and storage are enabled by donated individuals’ computers. The idea of Desktop grids is that resources can be gain from idle desktop computers. In Desktop grids, big task is divided into small tasks and those small tasks are distributed among the worker nodes and the result ready only when all tasks have been done. The term Cloud Computing
are coming to a seat of attentions as another Distributed computing technologies. Cloud Computing are brings lots of attractive features as new and powerful technology. The NIST(National Institute of Standards and Technology) defined Cloud Computing as “a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction” [3].

Although, the state of Art Cloud Computing with its lots of features; on demand application, Elastic Resource Capacity and Pay-per-used resources has considered revolution in computing paradigm but there is a gap for scientific research which those one that need huge processing power, because of economic issues. Volunteer computing is established for open and powerful computing that can satisfy scientific project. But application that is written for o adopt volunteer computing must be overwrite and compile for each platform (Windows, Linux, MAC OS, ext) that this need huge effort. The other important issue in volunteer computing is portability. Application portability has considered a big issue in volunteer computing frameworks. with using virtualization technology we can overcome to these problems. Virtualization is a technology that defines a software abstraction layer between OS applications and hardware resources [4]. This abstraction layer called virtual machine monitor (VMM) or hypervisor. Virtualization will be clearly explained in section 2. This article introduce a mechanism to enable virtualization into a volunteer computing platform. By enabling virtualization into volunteer computing, the porting effort will be eliminated one compiled application can run on different platforms. Actually this study can be considered as a part of big project where with enabling virtualization in desktop grid’s environment we can establishing IAAS from these volunteer’s resources. Although to achieve this cloud computing service we have to overcome many issues but this study can be the starting point of that aims. So although our main focus in this research is that to develop volunteer computing framework to enable virtualization but we consider some problem of cloud computing that can be solve by use of this study.

2 Related Works

The idea of volunteer cloud or clouds@home was completlyexplained in [5]. To achieve the volunteer cloud system goals we have to overcome many problems and it has considered as a long way but achivable. The first problem that we have to done is establishing virtualization technology in volunteer computing environment. There are some efforts to establish this goal.

In [6] they described an approach that named application sanboxing that is isolated the application inside the virtual machine. They have used a wrapper for launching VM and managing applications that have run on it. Their idea has been tested by BOINC. Recently they have published [7]. In their new work they have focused on the idea of volunteer cloud. They explained what are the challenges of establishing clouds@home, and then they defined Generic BOINC Application Client (GBAC) that is mechanism to forwarding jobs from virtual organizations into a computing nodes without any porting effort. They introduced the main idea of article as “The main idea of GBAC is that instead of porting and registering applications individually a generic virtualized application (GBAC) is ported and registered in BOINC.”

CernVM [8] has considered another work on adapting virtualization in Volunteer computing frameworks. They employed BOINC VBoxWrapper tool. In CernVM solutions the virtual appliance provides job scheduler completely so in this way the BOINC server scheduling does not utilized. The main disadvantages of this approach is that its virtual appliance is too big (about 800MB) and it doesn’t welcomed by volunteers.

3 Types of computing

In this part of study we are going to explain the most important computing paradigms. We will address following computing paradigms:

- Grid computing
- Cloud computing
- Volunteer computing
3.1 Grid computing

The first type of computing that we’d like to address is grid computing. According to (Foster et al., 2001) grid is "coordinated resource sharing and problem solving in dynamic, multi-institutional virtual organizations". In the first time resources was only file but afterwards researcher expand resources to the access computer directly, data, software and other type of resources. The dream of grid computing is to commute computing as a common utility like water, electricity. Grid computing has a security concerns , heterogeneity supporting issues and portability consequently.

3.2 Cloud computing

Cloud computing is a next computing paradigm that has consider as a revolution in computing. Finding a unique definition for Cloud computing is not possible. In [3] cloud computing defined as “Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction”. Another definition for cloud computing is in [9, 10] that “Cloud computing refers to both the applications delivered as services over the Internet and the hardware and systems software in the data centers that provide those services”. According to the [3] cloud computing has lots of features that can be address the recent problem of computing. In the form of grid in [11] defined cloud as form of grid computing, that virtual resources are dynamically allocated on a pay-per-used model. Cloud computing has two important portions. Cloud computing deployment models and cloud computing services. As in [12] mentioned cloud computing has three type of models:1)private cloud,2)public cloud,3)hybrid cloud. Private clouds refer to the those clouds that data and process are managed from inside of organization . Public clouds refer to those type of clouds that cloud infrastructure is available to the public and can be accessed via web.hybrid clouds is the combination of multiple clouds(private ,public ,...) with the goal of portability with standardization technology. Cloud computing with its service-oriented architecture make things(resources) as a service. The main services that cloud provisioned is infrastructure as a service, plateform as a service and software as a service that named IAAS, PAAS and SAAS respectively. The core architecture of cloud computing is shown in figure 1.

![Figure 1. Cloud computing architecture.](image)

3.3 volunteer computing

Volunteer computing refers to enrolling users those who are connected to a network to enable sharing their computer’s idle resources to solve large computational problems [?]. The most important point is design this idea through an easy and also transparent way. There are lots of volunteer computing projects ; SETI@home [13] is one of the most popular Volunteer computing projects that was started in 1999 and since now has had over 3 million volunteers.
3.3.1 Volunteer computing frameworks

In order to compute scientific problems we have to set up frameworks for volunteer computing. The main goal of framework is to section the job and distribute them all around the world and give the results back. There is no communication between each jobs, So the program must enable parallelization. There are lots of volunteer computing frameworks that have been developed; BOINC [14], Xtremweb [15], Distributed.net, Bayanihan [17], SLINC [17]. The most well-known framework for volunteer computing is BOINC (The Berkeley Open Infrastructure for Network Computing) that design in a client-server approach in which the clients is responsible to perform the jobs and server has a manager and coordinator role in the system.). BOINC is supported by approximately 2,500,000 users that holds about 7.5 petaflops (flops means floating point operations per second)(BOINCstates, 2012).

4 Implementing virtualization for Volunteer computing

In this part of study we are going to explain adaptability of virtualization for Volunteer computing. First of all we describe the definition of virtualization, hypervisor. Thereafter we will describe some efforts for mixing virtualization and Volunteer computing.

4.1 virtualiztion

In [4] virtualization is defined as “Virtualization is commonly defined as a technology that introduces a software abstraction layer between the hardware and the operating system and applications running on top of it”. The piece of software that enable virtualization is virtual machine monitor (VMM) or hypervisor. VMM is responsible for hiding physical resources of system from operating system. In the other meaning, the hardware of the system is controlled by hypervisor. With using virtualization we can support multiple and different OSs on the same hardware paralleled. Virtual machine (VM) is referred to a logical partitioned of hardware of system. Figure 2 shows the real system without virtualization and with it.

![Diagram of System](image)

**Figure 2.** diagram of system a)without virtualization b)with virtualization.

There are lots of virtualization tools that varied by license and also technique. VirtualBox [18] is one of the most well-known hypervisor that support cross-platform virtualization application. It is freely available as open source software under the GNU General Public License (GPL). By now It can be run on Windows, Linux, MAC and OpenSolaris. It supports shared folder between host and guest.it also has a command-line front end that enables controlling VirtualBox from host OS. Another open source virtualization tool is QEMU [19] that “QEMU achieves near native performances by executing the guest code directly on the host CPU also QEMU supports virtualization when executing under the Xen hypervisor or using the KVM kernel module in Linux”.

Xen [20] is another virtual machine monitor that implements for IA-32,ARM architecture and X86_64 architectures. It is also free. Xen is “an x86 virtual machine monitor which allows multiple commodity operating systems to share conventional hardware in a safe and resource managed fashion, but without sacrificing either
performance or functionality”. In Xen there is a term that named domain0 that is actually one of the multiple virtual machines which is resposible for hosting operating system. Xen supports checkpointing and migration. Vmware [21] is the most well-known of virtualization tool that has a comprehensive of solution for different type of virtualization. Vmware can be run on windows and Linux. Unfortunately Vmware has some licensing that make it insufficient for our project.

4.2 virtualiztion and Volunteer computing

In [22] the advantages of virtualization in a grid environment is mentioned. Isolation and security is the first advantage. Due to implementing the hypervisor we have another isolation layer on top of the system. With using this layer donors can trust more the project of Volunteer computing. So if the bugs occurred it doesn’t affect the whole system. Customization has emerged as the second advantage. virtual machines are in high customized level. You can specify parameters for virtual machine, such as CPU, memory also different operating systems. It is also possible to have multiple OSes in the same server hardware. The next feature that will enable by virtualization is Legacy support which is the result of customization. If you need to deploy a legacy software you can port it into a virtual machine.

With virtualization you can have more control on resources. You can define parameters like disk storage and RAM size for your virtual machine.furthermore there is possibility to limit the amount of usability of resources by defining some scheduling policies. You can also establish dynamic resource control.

5 Cloud@home: Overview

There is another approach to build cloud computing infrastructure with low scale and lowest price is named volunteer cloud or Cloud@home. In [5] they proposed the new concept of merging cloud computing and volunteer computing as a Cloud@home. The term Cloud@home is the combination of Cloud computing and volunteer computing where we can build Computing and Storage resources from donated hosts. So instead of Building costly big data-centres for cloud computing with can deploy lower scale infrastructures with substantial decreasing investment because almost of volunteer hosts are free.

5.1 Cloud@home: features

The term cloud@home brings for us substantial benefits and features that I collect the most important of them as below [23]:

• In comparison with Cloud computing, Cloud@Home applies at lower scale. The contribution range can be from a single user who shares his/her desktops to research community or organization.
• Security: In this type of Cloud infrastructure the data and resources are protected from local computers.
• Reliability: since the Cloud@home is established of resources that are larger than cloud so its reliability can be compare to grid or volunteer computing and it is bigger than cloud.
• Reliability: Interoperability among clouds: one of the most important goal of cloud@home is interoperability between clouds.
• Active role for users: users in cloud@home can sell or buy resources and in comparison with traditional form of cloud the users are more active role.

5.2 Cloud@home: Challenges

In this part, we are going to illustrate the challenges of deploying volunteer computing frameworks and also we are going to declare what is the problem of its frameworks recently. we are addressing both volunteer computing and cloud computing drawbacks that researchers who are intend to work on cloud side follow us as well. So in this part of article we are intending to depict problems from two views: Desktop grid views and Cloud that explained respectively.
5.2.1 Desktop Grid view

The term Grid Computing refers to a collection of software and hardware infrastructure which allows users to employ from its resources in a geography distributed models (Foster, 2002). These resources can be process, storage and specific data and can be found on the network.

Foster in [24] explained three important elements that define grid systems clearly. The dream of grid technology is that computing becomes a common utility such as water and electricity. The power of computational grid attracts scientists and researchers to fulfill and implement their researches on it. One of grid branches is a desktop grid that relies on harnessing of idle PC’s resources that are connected through network to work together on a specific computational complex application. Volunteer computing is a type of Desktop grid that is established to satisfy this extraordinary growing scientific application’s demands that relies on a volunteer’s PCs.

Volunteer computing is a technology that uses the idle time of PC’s to do research and scientific projects. In fact volunteer computing employs unused CPU cycles to fulfill their workflows. Volunteer computing provides petaflops of processing power to do things like formalization of complex mathematics models or predict climate changes and so on. The most famous example of Volunteer computing project is SETI@home (Anderson, 2004). BOINC has considers as a most popular Volunteer computing platforms since now.

To develop Volunteer computing platforms we have to consider many problems. One of these issues is we have to deploy easy to understand environment. This is because of the users (donors) are in a wide technical background ranges. Our platform must attract more volunteers to have more systems and so more powerful computing power.

Due to volunteer’s natural that is wide range geography distributed, donors might own very different systems that supported by a variety of OSES with different software and applications on it. So we have to consider framework independence to avoid from compatibility issues.

One of the most important points that we have to take into account in the systems that are based on volunteers is that to attract and convince the volunteers to donate his resources. To do so we have to deploy a powerful security mechanism that make sure the users from security concerns.

Furthermore, application in Volunteer computing has some considerable drawbacks; applications in Volunteer computing lack of portability, overwrite for each platforms (Windows 32/64, MAC OS x, Ubuntu, ext.) and also there is drawbacks about fault tolerance system and week resource manager system in Volunteer computing platform.

5.2.2 Cloud view

Cloud computing is based on service-oriented-architecture that makes all resources in cloud as a service [25]. The general form of services in Cloud computing are , infrastructure as a service-IASS, platform as a service-PASS, software as a service-SAAS [3]. These levels are supported by virtualization and management tools. Cloud computing infrastructure as a service(IASS) [3] is composed of three important components; storage, servers and networks as is presented in figure 3. Network is responsible for inter-connecting entire resources, servers can be any type of servers and Storage that is attached to the servers. Amazon has been emerged as precursor since 2006 offering storage and basic processing via internet with its products Amazon Elastic Compute (Amazon EC2) and Amazon Simple Storage Service (Amazon S3).

IASS has emerge as a evolution of traditional hosting that enable on demand resources provisioning (Bhardwaj et al., 2010). Cloud computing infrastructure requires large investment to deploy datacenters [26]. Cloud computing also has a problem of data lock-in due to lake of standardization [9]. The idea of volunteer cloud or clouds@home is that deploying cloud resources on a volunteer’s PCs. In this type of cloud computing model the pay-per-use exchanges to the virtual credit system which is completely free. To do that we have to overcome a lots of problems that in this article we are going to address some of them. One of the biggest concerns to move from costly and modern cloud’s datacenters to volunteer’s resources is the volatility and availability of resources. In volunteer computing it is common that some users get out the project by either resource’s owners or some technical occurrences (e.g., system crashing or power problem). Consequently fault tolerance has considered as an important effort to establish Volunteer cloud. Another important problem that has to be considered is convincing donors to donate their resources in a more level of access than before and also resources aren’t used only for scientific problems but also used by commercial providers.
6 solution’s design

In this part of our study we are going to illustrate our design framework. The goal of this study is to enable virtualization into volunteer computing frameworks. We choose BOINC for our framework. As we explained in section 2-3-1 we need hypervisor to support virtualization. We select VirtualBox as our hypervisor because it is cross-platform and it doesn’t require any specific hardware support. Another important tool that we are using in our proposed architecture is condor as our resource manager. Condor is very powerful scheduling that provides complete resource manager tools like resourcemonitoring, scheduling policy, job queuing mechanism. So our proposed architecture is consisting of:

- BOINC framework (server and client applications)
- Condor as our resource manager
- Virtualbox as our hypervisor
- Vmluancher : is a BOINC application that import virtual appliance into a hypervisor

7 Proposed architecture

In the figure 4 our architecture is explained. The pool jobs which are submitted to the condor will go to BOINC server and BOINC list them and deletes duplicated jobs. When volunteers start BOINC client, it downloads Vmluancher and virtual appliance from BOINC server. The Vmluancher opens Virtualbox and imports virtual appliance into VirtualBox. The virtual machine will start the job form BOINC wrapper and after it finishes it will backs the result to a BOINC server. If any problem occurs the condors with using checkpointing will migrates the virtual machine transparently to another volunteer’s system and start the job from its last checkpoint.

8 Conclusion

The main aim of this research is to provide a clear definitions on establishing Clouds@home frameworks. From Cloud point of view, Cloud computing has a lake of an open, data-lock in, huge investment is required for establishing data center and also recent Cloud computing model is not fit with scientific problem which is complex and need large computational power. Moreover in the form of cloud computing the users have a passive role that mean users can only submit their demands to the Clouds and after computing their demands the result will back to users. Another important issue in Cloud computing is that we cannot create any customize form of cloud and in passing of time only few but big companies will offer Cloud computing services. From volunteer computing point of view, applications that are registered in BOINC has problem of compatibility for different platforms, so scientists who want to compute their applications into BOINC should write their applications for
each platforms (Windows, Linux, ext.) and this is very time consuming. BOINC has also portability concerns. Moreover there are some concerns in security.

The main goal of this project is to integrate VMs running on donated hosts into the resource manager’s scheduler to transform them as a computing node in a set of virtual cluster. As we mentioned in scope we choose BOINC prototype as our volunteer computing framework. To enable virtualization into BOINC we have to employ some tools and also we have to develop the BOINC both server side and client side. From server development, we have to enable BOINC to communicate with condor as our resource manager. From client side, we have to develop BOINC client that run virtual appliance and also import work units into virtual appliance to run them. The main point that we have to consider is that all these developments must be transparent from client’s view. So our mechanism must enable transparently approaches to solve the problems. So the new computing environment will support heterogeneous environment due to virtualization. So security concerns in traditional Volunteer computing environment is improved. Moreover checkpointing and migration will be enabled by using virtualization so our computing environment will be improved a fault tolerance.

References


CUDA Based Implementation of 2-D Discrete Haar Wavelet Transformation

Hovhannes Bantikyan

State Engineering University of Armenia (Polytechnic), 105 Teryan Str., Yerevan, Armenia

bantikyan@gmail.com

Abstract. The discrete wavelet transform has a huge number of applications in science, engineering, mathematics and computer science. Most notably, it is used for signal coding, to represent a discrete signal in a more redundant form, often as a preconditioning for data compression. Practical applications can also be found in signal processing of accelerations for gait analysis, in digital communications and many others. In this paper presented implementation of 2-D DWT in parallel manner on Graphics Processing Unit, using CUDA technology. Calculating DCT in parallel, using multiple threads, gives us huge improvement in calculation speed.

Keywords
Discrete Haar Wavelet Transform, Parallel computing, GPGPU, CUDA programming.

1 Introduction

The wavelet transform, originally developed as a tool for the analysis of seismic data, has been applied in areas as diverse as signal processing, video and image coding, compression, data mining and seismic analysis. The theory of wavelets bears a large similarity to Fourier analysis, where a signal is approximated by superposition of sinusoidal functions. A problem, however, is that the sinusoids have an infinite support, which makes Fourier analysis less suitable to approximate sharp transitions in the function or signal. Wavelet analysis overcomes this problem by using small waves, called wavelets, which have a compact support. One starts with a wavelet prototype function, called a basic wavelet or mother wavelet. Then a wavelet basis is constructed by translated and dilated (i.e., rescaled) versions of the basic wavelet. The fundamental idea is to decompose a signal into components with respect to this wavelet basis, and to reconstruct the original signal as a superposition of wavelet basis functions; therefore we speak a multiresolution analysis. If the shape of the wavelets resembles that of the data, the wavelet analysis results in a sparse representation of the signal, making wavelets an interesting tool for data compression. This also allows a client-server model of data exchange, where data is first decomposed into different levels of resolution on the server, then progressively transmitted to the client, where the data can be incrementally restored as it arrives (“progressive refinement”).

This entire work is aimed to develop a strategy to compute DCT more efficiently and to reduce the time it takes for calculation. In this case the Graphics Processing Unit (GPU) based algorithm can be the cost effective solution. GPU can process large volume data in parallel when working in single instruction multiple data (SIMD) mode. In November 2006, the Compute Unified Device Architecture (CUDA) which is specialized for compute intensive highly parallel computation is unveiled by NVIDIA.

2 Wavelet Transformation

The main idea of (first generation) wavelet decomposition for finite 1-D signals is to start from a signal $c^0 = (c_0^0, c_1^0, ..., c_{N-1}^0)$, with N samples (we assume that N is a power of 2). Then we apply convolution filtering of $c^0$ by a low pass analysis filter H and downsample the result by a factor of 2 to get an “approximation” signal (or
“band”) $c^1$ of length $N/2$, i.e., half the initial length. Similarly, we apply convolution filtering of $c^0$ by a high pass analysis filter $G$, followed by downsampling, to get a detail signal (or “band”) $d^1$. Then we continue with $c^1$ and repeat the same steps, to get further approximation and detail signals $c^2$ and $d^2$ of length $N/4$. This process is continued a number of times, say $J$. Here $J$ is called the number of levels or stages of the decomposition. The explicit decomposition equations for the individual signal coefficients are:

$$c_{k}^{j+1} = \sum_n h_{n-2k}c_{n}^{j}$$  \hspace{1cm} (1)$$

$$d_{k}^{j+1} = \sum_n g_{n-2k}c_{n}^{j}$$  \hspace{1cm} (2)$$

where $h_n$ and $g_n$ are the coefficients of the filters $H$ and $G$. Note that only the approximation bands are successively filtered, the detail bands are left “as is”.

This process is presented graphically in Figure 1, where the symbol $\downarrow_2$ (enclosed by a circle) indicates downsampling by a factor of 2. This means that after the decomposition the initial data vector $c^0$ is represented by one approximation band $c^J$ and $J$ detail bands $d^1, d^2, ..., d^J$. The total length of these approximation and detail bands is equal to the length of the input signal $c^0$.

![Figure 1](image1.png)

**Figure 1.** Structure of the forward wavelet transform with $J$ stages: recursively split a signal $c^0$ into approximation bands $c^j$ and detail bands $d^j$.

Signal reconstruction is performed by the inverse wavelet transform: first upsample the approximation and detail bands at the coarsest level $J$, then apply synthesis filters $\tilde{H}$ and $\tilde{G}$ to these, and add the resulting bands. (In the case of orthonormal filters, such as the Haar basis, the synthesis filters are essentially equal to the analysis filters.) Again this is done recursively. This process is presented graphically in Figure 2, where the symbol $\uparrow_2$ indicates upsampling by a factor of 2.

![Figure 2](image2.png)

**Figure 2.** Structure of the inverse wavelet transform with $J$ stages: recursively upsample, filter and add approximation signals $c^j$ and detail signals $d^j$.

### 3 NVIDIA CUDA Basics

The fact that the performance of graphic processing units (GPUs) is much bigger than the central processing units (CPUs) of nowadays is hardly surprising. GPUs were formerly focused on such limited field of computing graphic scenes. Within the course of time, GPUs became very powerful and the area of use dramatically grew. So, we can come together on the term General Purpose GPU (GPGPU) denoting modern graphic accelerators. The driving force of rapid raising of the performance are computer games and the entertainment industry that evolves economic pressure on the developers of GPUs to perform a vast number of floating-point calculations within the unit of time. The research in the field of GPGPU started in late 70’s. In the last few years, we can observe the renaissance in this area caused by rapid development of graphic accelerators. Two main players in the field of GPGPUs are AMD with their ATI Stream Technology and NVIDIA which introduced Compute Unified Device Architecture (CUDA) - the parallel computing engine accessing GPGPUs resources to software
developers. Through the frameworks extending commonly used programming and scripting languages such as C, Java, Python or MATLAB, CUDA enables easy way to make applications using NVIDIA GPUs.

### 3.1 Hardware Architecture

Present multi-core CPUs usually consist of 2-8 cores. These cores usually work asynchronously and independently. Thus, each core can execute different instructions over different data at the same time. According to the Flynn’s taxonomy, we are talking about Multiple Instruction stream, Multiple Data stream (MIMD) class of computer architectures.

On the other hand, GPUs are designed for parallel computing with an emphasis on arithmetic operations, which originate from their main purpose - to compute graphic scene which is finally displayed. Current graphic accelerators consist of several multiprocessors (up to 30). Each multiprocessor contains several (e.g., 8, 12 or 16) Arithmetic Logic Units (ALUs). Up to 480 processors is in total on the current high-end GPUs. Figure 3 shows the general overview of the CPU and GPU.

![Figure 3. Comparison CPU and GPU architectures.](image)

Also the memory hierarchy is specific in the case of graphic accelerators. Each multiprocessor has registers that are used by ALUs. Processors within a multiprocessor can access shared memory of typical size 16KB, or a main memory of the accelerator, which is not cached on the majority of present accelerators. Global memory in terminology of CUDA, is accessible from all the processors on the accelerator. In addition, there are two separate memory areas - constant memory and texture memory, also shared across the multiprocessor and both cached and read-only. When accessing some element from the texture memory, a couple of surrounding elements

![Figure 4. CUDA memory model.](image)
are also loaded. This feature is called spatial locality. One of the most limiting factors is a very small capacity of shared memory and registers. If application uses more variables per thread than available registers, they are stored in a local memory which is, in fact, the dedicated part of global memory. Accessing these variables is as time-consuming as accessing any other variable stored in the global memory. For better understanding, we can see the memory hierarchy in Figure 4.

3.2 Programming Model

A CUDA-capable GPU is referred to as a device and the CPU as a host. Thread is the finest grain unit of parallelism in CUDA.

![CUDA programming model](image)

**Figure 5.** NVIDIA CUDA programming model.

Thousands of threads are able to run concurrently on the device. Threads are grouped into the warps. Size of a warp is usually 32 threads. Warp is the smallest unit that can be processed by multiprocessors. Warps scheduled across processors of one multiprocessor are coupled into a thread blocks. Block is a unit of the resource assignment. Typical size of a thread block is 64-512 threads and depends on the particular application what is the optimal size of a thread block to ensure the best utilization of the device. Thread blocks form a grid. Grid can be viewed as a 1-dimensional, 2-dimensional or 3-dimensional array. Fig. 5 is depicting the described hierarchy.

3.3 Features and Limitations of CUDA

It is easy to learn the CUDA API, but hard to programme efficient applications which utilize the GPU’s performance. CUDA API is a set of extensions based on the standard C language. Counterweight to many features of this massively parallel architecture is that there are limitations mostly caused by HW architecture. CUDA belongs to the class of Single Instruction, Multiple Thread (SIMT) according the Flynn’s taxonomy. SIMT originates in Single Instruction Stream, Multiple Data Stream (SIMD) class known for example from the supercomputers based on vector processors (e.g., Cray-1). SIMT also implies the divergence in the program that usually leads to the serialization of the run. Recursive functions are not supported either. As introduced before, graphic accelerators were developed with the focus on computing vast amounts of arithmetic operations. Many of them are implemented directly in the hardware with a cost of units of warp-cycles. Besides arithmetic functions there is a set of bitwise operations also implemented "in hardware".
Haar transform on a little example. Suppose you are given \( N \) values scale this values. For displaying image after transformation we scale back transformed values. Let’s look at 1D for each component separately. Any component (R G B) has values from 0 to 255 to before transformation we scale this values. For displaying image after transformation we scale back transformed values. Let’s look at 1D Haar transform on a little example. Suppose you are given \( N \) values

\[
x = (x_0, x_1, \ldots, x_{N-1})
\]

where \( N \) is even. We take pair-wise average of numbers \( s_k = \frac{(x_{2k} + x_{2k+1})}{2} \) for \( k = 0, \ldots, N/2 - 1 \). For example

\[
x = (6, 12, 15, 15, 14, 12, 120, 116) \rightarrow s = (9, 15, 13, 118)
\]

We need second list of data \( d \) so that the original list \( x \) can be recovered from \( s \) and \( d \). For \( d_k \) (called directed distances) we have \( d_k = \frac{(x_{2k} - x_{2k+1})}{2} \) for \( k = 0, \ldots, N/2 - 1 \). The process is invertible since

\[
s_k + d_k = \frac{(x_{2k} + x_{2k+1})}{2} + \frac{(x_{2k} - x_{2k+1})}{2} = x_{2k}
\]

\( \frac{(x_{2k} - x_{2k+1})}{2} = x_{2k+1} \)

So we map \( x = (x_0, x_1, \ldots, x_{N-1}) \) to \( (s|d) = (s_0, \ldots, s_{N/2-1}|d_0, \ldots, d_{N/2-1}) \). This process is repeated recursively for \( s \). Using our example values we have

\[
(6, 12, 15, 15, 14, 12, 120, 116) \rightarrow (9, 15, 13, 118) \rightarrow (3, 0, 1, 2)
\]

\[
(9, 15, 13, 118) - 3, 0, 1, 2) \rightarrow (12, 65.5) - 3, -52.5) - 3, 0, 1, 2)
\]

\[
(12, 65.5) - 3, -52.5) - 3, 0, 1, 2) \rightarrow (38.75) - 26.75) - 3, -52.5) - 3, 0, 1, 2)
\]

So final result is \((38.75) - 26.75) - 3, -52.5) - 3, 0, 1, 2)\). Why might people prefer the data in this form?

- We can identify large changes in the differences portion \( d \) of the transform,
- it is easier to quantize the data in this form,
- the transform concentrates the information (energy) in the signal in fewer values,
- and the obvious answer: fewer digits.

To implement optimal algorithm for GPU we have to consider image sizes and GPU limitations. One of the most important parameter is \( \text{threadsPerBlock} \). This shows number of threads in a single block, which can cooperate together using shared memory. Threads from different blocks can’t cooperate. To transform one row we need \( N = (\text{width}/2 \times \text{threadsPerBlock}) \) working threads. We assume that width and height of image are power of 2. Each thread is doing calculation of \( s_i \) and \( d_i \), where \( i \) is index of thread. If image width is bigger than \( 2 \times \text{threadsPerBlock} \), than we cut image to parts and each thread works for each part. For example consider \( \text{threadsPerBlock} = 512 \) and width = 2048. In this case we cut image to two parts and \((0, 1024), (1, 1025) \ldots \) pixels will be processed by the same thread. So thread will calculate two pair of \( s \) and \( d \). After this calculations we need to synchronize threads by calling \text{syncthreads} \) CUDA function, and start second level of transform, which will need to times less threads. So we will run Kernel with
<< 3 * height, N >> parameters, where N is above mentioned threads count, and we multiple height to 3 for R G B components.

In Figure 6 we can see results of our algorithm: 2 level transformed Lena image and 1 level transformed Zelda image.

![Image of Lena and Zelda images](image_url)

**Figure 6.** 2 level FWT for Lena image and 1 level FWT Zelda image.

Experiments done on

- CPU: Intel(R) Core(TM) i3-2100 3,10GHz
- GPU: GeForce 9500 GT, Max threads per block: 512, Max blocks in kernel lunch: 65,535.

We compare the proposed CUDA version of the 2D-FWT with a sequential CPU implementation. We choose images of different sizes (512x512, 1024x1024, 2048x2048 and 4096x4096). We take into account the time needed to copy data and results to and from the GPU.

![Chart of execution time](chart_url)

**Figure 7.** Execution time using CPU and GPU.

5 Conclusion

CUDA is a new hardware and software architecture for issuing and managing computations on the GPU, without the need of mapping them to a graphics API, common to the latest NVIDIA developments. It promises
to simplify the development of applications that take full advantage of current and future powerful GPUs. In this paper we have presented and evaluated an initial implementation of the 2D fast wavelet transform for CUDA-enabled devices. A brief introduction to the wavelet transform and CUDA has been provided prior to explaining our parallelization strategy. We have compared the proposed CUDA version of the 2D-FWT with a sequential implementation. As we can see, we gain in performance using parallel GPU algorithm. Parallelizing the sequential and simple FHT algorithms will be beneficial to control code complexity and minimize execution time of the process.

6 Acknowledgments

I would like to thank Hakob Sarukhanyan for useful and pragmatic suggestions.

References


Грид-инфраструктура ФТИНТ им. Б.И. Веркина НАН Украины и ее практическое использование в научных исследованиях

Белан В.И.1, Белоус Л.Ф.1, Егупов С.А.1, Зобина В.Г.1, Иванов А.Ю.1, Карачевцев В.А.1, Косевич М.В.1, Нацк В.Д.1, Поляков В.М.2, Рубин Ю.В.1, Смирнов С.Н.1, Степаньян С.Г.1, Торгонин Е.Ю.2, Чаговец В.В.1

1Физико-технический институт низких температур им. Б.И. Веркина НАН Украины, пр. Ленина, 47, Харьков, Украина
2Белгородский государственный технологический университет им. В.Г. Шухова, ул. Костюкова 46, Белгород, Россия

belan@ilt.kharkov.ua, belous@ilt.kharkov.ua, egupov@ilt.kharkov.ua, zobina@ilt.kharkov.ua, ivanov@ilt.kharkov.ua, karachevtsev@ilt.kharkov.ua, mvkosevich@ilt.kharkov.ua, p_v_m@mail.ru, natsik@ilt.kharkov.ua, rubin@ilt.kharkov.ua, smirnov@ilt.kharkov.ua, stepanian@ilt.kharkov.ua, e.yudek@gmail.com, vvchagovets@gmail.com

Аннотация. Суперкомпьютерные вычисления (HPC – High-Perfomance Computing) с неограниченным территориальным доступом являются одним из основных преимуществ грид-инфраструктуры. Научные расчеты с использованием HPC проводятся в Физико-техническом институте низких температур (ФТИНТ) им. Б.И. Веркина Национальной академии наук Украины в традиционных для института направлениях исследований: физике высокотемпературной сверхпроводимости, физике нанокристаллов, теоретической физике, молекулярной биофизике. В докладе кратко изложена история развития грид-инфраструктуры ФТИНТ и представлены несколько проектов, выполняемых во ФТИНТ в настоящее время с помощью HPC.

Ключевые слова

EGI, грид, квантовая химия, молекулярная динамика, молекулярная биофизика, нанокристаллы, углеродные нанотрубки, органические наночастицы, галогенпроизводные урацила, ИК-Фурье спектры.

1 Введение

Использование суперкомпьютерных мощностей сотрудниками Физико-технического института низких температур (ФТИНТ) им. Б.И. Веркина Национальной академии наук Украины началось довольно давно, когда в 1996 году Ю.В. Рубин и С.Г. Степаньян получили возможность доступа к вычислительным ресурсам в Компьютерном центре государственного университета г. Джексон (штат Миссисипи, США), а также в Компьютерном центре Университета Аризоны (штат Аризоны, США). Работы, связанные с грид-технологиями, были начаты во ФТИНТ в 2007 году, когда в институте за собственные средства был создан мини-кластер (грид-платформа) и осуществлено его подключение к украинскому, тогда еще академическому, гриду (УАГ).


Подробное описание развития грид-технологий в Украине приведено в [1].
2 Основная часть
ФТИНТ принимал активное участие в обеих государственных программах создания и развития грид-технологий и в настоящее время входит в NGL_UA в качестве ресурсного центра UA_ILTPE_ARC. Из-за явной недостаточности финансирования этих программ ресурсы украинского грид выглядят весьма скудно, но открывшиеся возможности международного сотрудничества на уровне виртуальных организаций (ВО) в большой степени нивелируют этот недостаток. В частности, после подписания Меморандума о взаимопонимании между польским и украинским грид (http://inrastructure.kiev.ua/news/114/), открыт доступ к ресурсам польского грид (PL-Grid) – виртуальной организации Gaussian на базе современной грид-технологии qcg-computing [2]. Зарегистрированные члены ВО Gaussian получают доступ к последней версии коммерческого пакета с одним названием, который является одним из самых популярных инструментов для квантово-химических расчетов. Что касается мощности вычислительных ресурсов PL-Grid, то 145 место в TOP500 для наиболее мощной машины Zeus, с производительностью порядка 0.3 Pflops, говорит само за себя.

Проблема понятного и удобного пользователю программного интерфейса является актуальной и важной для всех видов программного обеспечения. Особенно это касается сферы высокоопроизводительных вычислений, где традиционные интерфейсы пользователей для доступа пользователей к HPC-ресурсам очень специфичные и требуют дополнительных технических знаний. Развитие грид-технологий, а также недавно возникшей технологии облачных вычислений (Cloud computing), никак не уменьшило проблему создания дружественного интерфейса. Ведь работа в гриде – это еще один дополнительный уровень сложности, который требует знания грид-инструментов командной строки, нового синтаксиса запуска задач, межкластерной совместимости программных окружений и т.д. Во ФТИНТ и ряде других академических учреждений Украины и России применяется специальная система управления суперкомпьютером (SCMS) [3] с удобным пользовательским web-интерфейсом (Рис. 1), которая является успешной попыткой уменьшить этот языковый разрыв и предоставить ученому возможность работать только в своей предметной области, не отвлекаясь на изучение множества технических деталей машин и их операционного окружения.

Важно отметить инициативу Института Теоретической Физики им. Н.Н. Боголюбова НАН Украины и НТУУ КПИ по подготовке молодых специалистов в области грид-технологий, по созданию учебных центров, учебных программ и изданию ряда учебных пособий [4]. ФТИНТ также участвует в этом процессе и входит в состав международной учебно-исследовательской грид-инфраструктуры t-infrastructure gLite|EMI|UMD, созданной и курируемой Объединенным Институтом Ядерных Исследований (ОИЯИ, Дубна, Россия). Предоставляемые для обучения ресурсы доступны членам виртуальных организаций BITPEDU, KPIEDU, EDU в режиме 7/24/365.

В дальнейшем приводятся примеры научных исследований, использующих возможности так называемой «виртуальной лаборатории e-science», которая подробно описана в [5].
2.1 Компьютерное моделирование краудионов в двумерных (2D) кристаллах методами молекулярной динамики

В настоящее время в физике конденсированного состояния вещества значительный интерес вызывают низкоразмерные наносистемы, в том числе двумерные (2D) кристаллы ограниченных размеров. Физико-механические свойства таких кристаллов сильно зависят от их дефектной структуры, в частности, от наличия собственных дефектов, не связанных с внедрением в кристалл чужеродных структурных элементов. В семействе собственных дефектов важную роль играют краудионы — незавершенные (не прошедшие через весь кристалл) пластические сдвиги фрагментов плотноупакованных атомных рядов на один параметр решетки, образующие внутри кристалла деликатизованный атом внедрения.

Рассмотрены 2D кристаллы ограниченных размеров с гексагональной решеткой, в которой межатомное взаимодействие описывается парным центрально-симметричным потенциалом Леннарда-Джонса \[ 6 \], при этом учитывалось взаимодействие каждого атома со всеми остальными атомами кристалла без "обрезки" потенциала на больших расстояниях. Движение атомов описывалось уравнениями классической механики. Моделирование краудионов в 2D кристаллах выполнено методами молекулярной динамики с применением модифицированного скоростного алгоритма Верле \[ 7 \]. Для увеличения точности вычисления сомн большого множества чисел с плавающей точкой, которые существенно отличаются по величине, применялся алгоритм Кэхэна (компенсационное суммирование) \[ 8 \]. Параллельные вычисления выполнялись с помощью технологии MPI \[ 9 \].

В качестве начальной атомной конфигурации взят фрагмент идеальной плоской гексагональной решетки, вписанной в круг заданного радиуса \( R \), с центром в одном из узлов решетки. Величина \( R \) задавалась кратной величине параметра решетки \( a \). Краудион вводился в центр кристалла путем незавершенного сдвига атомов центрального плотноупакованного ряда. После этого выполнялся расчет атомной релаксации при нулевой температуре и отсутствии внешних сил.

Получены атомные конфигурации кристаллов с краудионом (Рис. 2) и вычислены значения его собственной энергии \( U \) в кристаллах различных размеров \( R \). Установлено влияние величины \( R \) на энергию краудиона. Энергия \( U \) возрастает при увеличении \( R \) и при \( R \geq 50a \) приближается к величине \( U \approx 3,4U_a \), где \( U_a \) — энергия на один атом в бездефектном бесконечно протяженном кристалле. Проведен анализ структурных искажений, обусловленных краудионами (Рис. 3), и описаны неоднородные относительные деформации. Координатные зависимости компонент тензора дисторсии \( w_{i,j}(x,y) \) показаны на Рис. 4. Поле упругих дисторсий, обусловленное краудионом, на расстояниях от его центра превышающих \( \approx 10a \) убывает по степенным законам, что соответствует описанию краудиона в рамках линейной теории упругости.

Полученные результаты важны для развития физики дефектов в низкоразмерных нанокристаллах.

Рис. 2. Центральный фрагмент гексагонального 2D кристалла с краудионом, его эффективное ядро условно выделено прямоугольником.

Рис. 3. Зависимость смещения атома \( n \) от его номера \( n \) в центральном ряду.

Рис. 4. Зависимость от координаты \( x \) компонент тензора дисторсии \( w_{i,j}(x,0) \), \( w_{i,j}(x,a) \) и \( w_{i,j}(x,a) \): ось \( x \) совпадает с осью центрального ряда, \( x_0 \) — координата центра краудиона.
2.2 Компьютерное моделирование комплексов углеродных нанотрубок с олигомерами полиоксиметилена

Актуальной задачей для совершенствования методов создания новых наноматериалов является проведение компьютерных экспериментов с использованием возможностей методов молекулярной динамики и квантовой химии для моделирования самоорганизации наночастиц. В рамках данного проекта изучена самоорганизация комплексов, формирующихся из олигомеров органического полимера полиоксиметилена (ПОМ) и одностенных углеродных нанотрубок (УНТ). Предполагается, что такие комплексы образуются при изготовлении нанокомпозита УНТ-ПОМ путем ультразвуковой обработки суспензии УНТ в формалине с последующей криотехнологической обработкой, включающей низкотемпературную полимеризацию и сублимационную сушку.

Методом молекулярной динамики были исследованы системы, состоящие из фрагментов УНТ zigzag(10,0) разной длины и олигомеров ПОМ, содержащих разное число мономеров n, в условиях вакуума и водного окружения. При моделировании системы УНТ-ПОМ в вакууме наблюдалось скольжение полимерной цепочки по поверхности нанотрубки; отрыв цепочки от поверхности не наблюдался.

При моделировании комплекса ПОМ20 и небольшого фрагмента УНТ, термированного атомами водорода, в водном окружении обнаружился новый результат. А именно, в процессе моделирования наблюдалось постепенное затягивание цепочки олигомера в середину нанотрубки (Рис. 5). При достаточно длительном времени расчетов вся цепочка ПОМ20 втягивалась внутрь трубы.

![Рис. 5. Структура комплекса УНТ zigzag(10,0) с олигомером ПОМ20 в водной среде, рассчитанная методом молекулярной динамики (молекулы воды не показаны).](image)

Обнаруженная в компьютерном эксперименте нековалентная функционализация нанотрубок олигомерами ПОМ может дать объяснение эффекта стабилизации водной суспензии УНТ с ПОМ: наличие гидрофильного полимера в комплексе с гидрофобной УНТ повышает гидрофильность и растворимость комплекса в целом. Этот результат представляет практический интерес для специалистов, занимающихся созданием нанокомпозитов на основе УНТ и органических полимеров.

Далее, в комплексе УНТ-ПОМ, олигомер ПОМ принимает специфическую конформацию (Рис. 6, а): в то время, как в свободной цепочке ПОМ атомы кислорода располагаются по спирали, в цепочке, которая находится в середине нанотрубки, атомы кислорода развернуты в одну сторону. Методом HF/6-31++G(d,p) были рассчитаны и построены карты распределения электростатического потенциала вокруг олигомера ПОМ, выделенного из его комплекса с УНТ (Рис. 6, б), а также вокруг всего комплекса УНТ-ПОМ (Рис. 6, в).

![Рис. 6. Конформация олигомера ПОМ20, выделенного из его «эндо»-комплекса с УНТ, (а) и распределение электростатического потенциала вокруг такого олигомера (б) и комплекса УНТ-ПОМ (в).](image)

Из Рис. 6, б следует, что у ряда атомов кислорода в середине цепи ПОМ создается протяженная зона отрицательного потенциала, а из Рис. 6, в – что внутри нанотрубки образуется своего рода диполь.

Таким образом, полученные в компьютерном эксперименте результаты позволили выявить неожиданные и необычные свойства комплексов УНТ-ПОМ, которые заслуживают дальнейшего исследования.
2.3 Квантово-механические расчеты таутомерии и колебательных спектров галогенпроизводных молекул урацила, изолированных в инертных матрицах


Галогенпроизводные основания РНК урацила являются важным классом биологически активных соединений. Их используют в фундаментальных исследованиях эволюции живых организмов [13], процессов мутагенеза [13, 14], создания новых лекарственных препаратов, в том числе противовирусных. Компьютерное моделирование показало возможность существования енольных таутомеров 5-бромурацила в кластерах воды [14]. Однако дикето-форма является доминирующей для молекул урацила и его 5-галогензамещенных, находящихся в изолированном состоянии.

В нашей работе были получены ИК-Фурье спектры молекул 5Br-Ur и 5J-Ur, изолированных в инертных матрицах. Квантово-механические расчеты молекулярных структур и относительных энергий таутомеров были проведены для всего ряда галогенпроизводных урацила: 5FUr, 5ClUr, 5BrUr и 5JUr. Для расчета энергий использовались методы DFT (функционал B3LYP), MP2 и CCSD(T) с базисами aug-cc-pVDZ, 6-311++G** и aug-cc-pVTZ.

Расчеты относительных энергий показали преобладание кето-формы и последовательное уменьшение ее стабильности в ряду: Ur, 5JUr, 5BrUr, 5ClUr, 5FUr. Согласно расчетам суммарная заселенность редких таутомерных форм не должна превышать 0.1% в диапазоне температур испарения 420-480К, что хорошо коррелирует с экспериментальной оценкой (0.2%) [15].

Для оптимизации поиска характеристических частот редких таутомеров был выполнен анализ результатов статистических расхождений между экспериментальными и расчетными частотами колебательных спектров. Было установлено, что метод DFT/B3LYP с базисом 6-311++G(df,pd) демонстрирует лучшую статистику, чем этот же метод с базисом aug-cc-pVTZ [15]. Кроме того, использование базиса 6-311++G(df,pd) в четыре раза уменьшает время компьютерного расчета колебательного спектра такой многоатомной системы.

На рисунке 7 представлены фрагменты экспериментального и рассчитанного ИК спектров в области валентных и плоскостных деформационных колебаний кольца молекул 5BrUr (матрица Ar, T=10 K). 1 – эксперимент, 2 – результаты расчета методом DFT/B3LYP/6-311++G(df,pd) с коррекцией частот.

Расчет колебательных спектров 5-бромурацила в составе кластеров с 32 атомами инертного газа (Ne, Ar, Kr) методом DFT/M06-2x/6-31++G(d,p) показал качественное согласие с экспериментом.

Для дальнейшего улучшения согласия расчетов и эксперимента, впервые предложено масштабирование расчетных частот методом наименьших квадратов с использованием полинома [15]. Показано, что применение полинома второго (или третьего) порядка для корректировки частот, полученных методом DFT/B3LYP, уменьшает среднеквадратичную ошибку между расчетом и экспериментом до 4–5 см⁻¹ [15].
2.4 Изучение с помощью методов DFT, MP2 и CCSD конформационной структуры, относительной стабильности и колебательных спектров простейшего пептида NAG

Даже небольшие биологические молекулы, такие как аминокислоты и короткие пептиды, обладают высокой конформационной лабильностью из-за наличия в их структуре нескольких вращательных степеней свободы. Идентификация всех возможных конформеров этих молекул является очень важной для правильного предсказания их наиболее стабильных форм, а также для определения заселенности конформеров. При этом требуется проведение полного сканирования многомерной поверхности потенциальной энергии с использованием точных квантово-механических методов. Недавно мы показали, что методы DFT/B3LYP и MP2 могут эффективно применяться для изучения конформационной структуры лабильных молекул β-аланина и лейцина [16, 17] и предсказывать колебательные спектры этих соединений, которые были использованы для анализа экспериментальных ИК-Фурье спектров молекул, изолированных в низкотемпературных инертных матрицах.

В данном проекте мы использовали теорию функционала плотности (DFT) с функционалом B3LYP, теорию возмущений второго порядка Мюллера-Плессета (MP2) и метод спаренных кластеров (CCSD) для изучения конформационной структуры и колебательных спектров простейшего пептида N-ацетилглицина (NAG). Шесть 3D поверхностей потенциальной энергии NAG были рассчитаны с помощью методов DFT/B3LYP/aug-cc-pVDZ и MP2/aug-cc-pVDZ. В целом были выполнены около 10000 оптимизации геометрии NAG для различных значений двугранных углов. Было использовано современное распараллеленное программное обеспечение, реализованное в пакетах программ GAUSSIAN09 [10], PC GAMESS [11] и Firefly [12]. Примеры рассчитанных поверхностей потенциальной энергии приведены на Рис. 8. Минимумы на поверхности соответствуют устойчивым конформерам NAG.

![Рис. 8. 2D проекции трехмерной поверхности потенциальной энергии молекулы NAG, рассчитанные методами B3LYP/aug-cc-pVDZ (слева) и MP2/aug-cc-pVDZ (справа) (шкала уровней энергии в кДж/моль).](image)

Метод DFT продемонстрировал лучшую эффективность загрузки процессоров при увеличении числа ядер в расчетах > 98% в конфигурации 16-ядер/два SMP узла (Firefly) и > 95% в конфигурации 32-ядра/два SMP узла (Gaussian09). В то же время метод MP2 с алгоритмом semidirect, который требует чрезвычайно большого внешнего дискового пространства для временных файлов, имеет низкую эффективность (около 60 %) в конфигурации 16-ядер/один узел (Gaussian09). С другой стороны, алгоритм fulldirect метода MP2, при использовании которого временные данные хранятся в оперативной памяти, был гораздо более эффективным (около 90 %), но требовал выделения значительного количества оперативной памяти (96 ГБ для 16 ядерного узла). Загрузка процессоров при использовании метода CCSD значительно уменьшается и составляет менее 50% из-за огромного внешнего дискового пространства, требуемого для хранения временных данных, которые являются обязательными для этого метода.

3 Заключение

В работах использовались следующие пакеты прикладных программ: Gaussian, NWChem, NAMD, Firefly (PC-GAMESS), Gromacs, доступные либо в локальном режиме, либо через виртуальные организации EGI: MolDynGrid (UA-Grid), CompuChemGridUA (UA-Grid), Gaussian (PL-Grid, qcg-icon).

По материалам исследований с использованием HPC-технологий за последние три года сотрудниками ФТИНТ опубликовано более 30 статей в международных журналах.
4 Благодарности

Авторы выражают искреннюю благодарность Координационному Комитету государственной целевой программы UA-Grid и руководству академических институтов за возможность использования грид-кластеров:

- Физико-технического института низких температур им. Б.И. Веркина НАН Украины;
- Института кибернетики им. В.М. Глушкова НАН Украины;
- Института теоретической физики им. Н.Н. Ботюковб НАН Украины;
- Института молекулярной биологии и генетики НАН Украины;
- Института синтетических материалов НАН Украины.

Авторы надеются на дальнейшее успешное международное сотрудничество с польским грид и благодарят его руководство за возможность использования ресурсов виртуальной организации Gaussian.

Часть работ выполнена с использованием вычислительных ресурсов Компьютерного центра Университета Аризоны (штат Аризона, США) и суперкомпьютерного комплекса МГУ имени М.В. Ломоносова [18] (Москва, Россия). Авторы выражают признательность за возможность использования этих ресурсов.

Литература


Grid Infrastructure of B. Verkin ILTPE NASU and its practical use in scientific research

Belan V.I.¹, Belous L.F.¹, Egupov S.A.¹, Zobnina V.G.¹, Ivanov A.Yu.¹, Karachevtsev V.A.¹, Kosevich M.V.¹, Natsik V.D.¹, Polyakov V.M.², Rubin Yu.V.¹, Smirnov S.N.¹, Stepanian S.G.¹, Torgonin E.Yu.², Chagovets V.V.¹

¹ B. Verkin Institute for Low Temperature Physics and Engineering of National Academy of Sciences of Ukraine, 47 Lenin Ave., Kharkov, Ukraine
² V. Shukhov Belgorod State Technological University, 46 Kostyukov str., Belgorod, Russia

belan@ilt.kharkov.ua, belous@ilt.kharkov.ua, egupov@ilt.kharkov.ua, zobnina@ilt.kharkov.ua, ivanov@ilt.kharkov.ua, karachevtsev@ilt.kharkov.ua, mvkosevich@ilt.kharkov.ua, p_v_m@mail.ru, natsik@ilt.kharkov.ua, rubin@ilt.kharkov.ua, smirnov@ilt.kharkov.ua, stepanian@ilt.kharkov.ua, e.yudek@gmail.com, vvchagovets@gmail.com

Abstract. Supercomputing (HPC – High-Performance Computing) with unlimited territorial access is one of basic advantages of the grid infrastructure. Scientific computation with use of HPC is being performed at B. Verkin Institute for Low Temperature Physics and Engineering (ILTPE) of the National Academy of Sciences of Ukraine in the traditional areas of research for the institute: physics of high-temperature superconductivity, physics of nanocrystals, theoretical physics and molecular biophysics. This report summarizes the history of the grid infrastructure development at ILTPE and presents several projects being performed at ILTPE nowadays by means of HPC.

Keywords

EGI, grid, quantum chemistry, molecular dynamics, molecular biophysics, nanocrystals, carbon nanotubes, organic nanoparticles, halogenated uracil, FTIR spectra.

1 Introduction

Researchers of B. Verkin Institute for Low Temperature Physics and Engineering (ILTPE) of the National Academy of Sciences of Ukraine had begun to use supercomputing facilities long time ago, when in 1996 Yu.V. Rubin and S.G. Stepanian had got the access to computing resources at the Computer Center of Jackson State University (Mississippi, USA) and at the Computer Center of the University of Arizona (Arizona, USA). Work related to grid technologies had begun at ILTPE in 2007, when the institute created mini-cluster (grid platform) by means of its own funds, and connected to the first Ukrainian grid segment, just academic at that time.

In 2008, as the results of execution of program «Implementation of grid technologies and the creation of clusters in the NASU» the grid node was installed at the institute with modern enough for that time, 88 core cluster as one of three branches of distributed Northeast Grid Coordinating Center. Further development of grid technologies in Ukraine continued due to the second State organized scientific and technical program «Implementation and application of grid technology in 2009-2013» (UA-Grid). Successful implementation of these two state programs ultimately had led to the creation of the National Grid Initiative of Ukraine (NGI UA). In mid-July 2012 NGI UA had completed the certification process for major grid services, and had become a full member of European Grid Infrastructure (EGI).

Detailed description of the grid technologies development in Ukraine is given in [1].
2 Main sections

ILTPE took an active part in both government programs on creation and development of grid technology, and it’s now included in NGI-UA as the EGI resource center UA_ILTPE_ARC. Because of an apparent lack of funding of these programs the resources of Ukrainian grid look very modest, but new opportunities for international cooperation at the level of virtual organizations (VO) negate this disadvantage to large an extent. In particular, after the signing of the Memorandum of Understanding between the Polish and Ukrainian grids (http://infrastructure.kiev.ua/news/114/), open access to the resources of the Polish grid (PL-Grid) – VO Gaussian based on the modern grid technologies qcrg-computing [2]. VO Gaussian registered members have access to the latest version of the commercial package with the same name, which is one of the most popular tools for quantum chemical calculations. As for the power of PL-Grid computing resources, then 145-th rank on the list TOP500 for most powerful machine Zeus with a performance about 0.3 Pflops, speaks for itself.

The problem of intuitive and user-friendly software interface is relevant and important for all kinds of software. This is especially true for high-performance computing, where traditional interfaces for user access to HPC resources are very specific and require additional technical knowledge. The development of grid technology, as well as the newly emerging technology of «cloud computing», does not reduce the problem of creating a user-friendly interface. Because working in the grid - this is another level of complexity, which requires competence in the grid command-line tools, new syntax for running tasks, inter-cluster compatibility, software environments, etc. At ILTPE and several other academic institutions of Ukraine and Russia the special SuperComputer Management System (SCMS) is used [3] with user-friendly web interface (Fig. 1), which is a successful attempt to reduce this language gap and provide for scientists the opportunity to work only in their application area that doesn’t require to be focused on the study of many technical details of computers and their operating environment.

It should be noted the important initiative of Bogolyubov Institute for Theoretical Physics of National Academy of Sciences of Ukraine and National Technical University of Ukraine «Kyiv Polytechnic Institute» related to training of young specialists in the field of grid technologies, to creation of training centers, training programs and to publication of a number of textbooks [4]. ILTPE also is involved in this process and as UA-ILTPE resource it is joined to the international educational and research grid gLite|EMI|UMD t-infrastructure created and supervised by Joint Institute for Nuclear Research (JINR, Dubna, Russia). These training resources are available to members of virtual organizations BITPEDU, KPIEDU, EDU in 7/24/365 mode.

The following are examples of research projects using the so-called «virtual laboratory e-science» opportunities, described in detail in [5].

![Fig. 1. The web-portal of SuperComputer Management System SCMS-ILTPE.](http://ds0.ilt.kharkov.ua/index.php?lang_id=1&menu_id=45)
2.1 Molecular dynamics computer simulation of crowdions in two-dimensional (2D) crystals

Low-dimensional nanosystems, including two-dimensional (2D) crystals of finite size, are currently of considerable interest in condensed matter physics. Physical-mechanical properties of such crystals depend strongly on their defect structure, particularly on the presence of native defects, which are not related to implantation of foreign structural elements into a crystal. Crowdions play an important role in a native defects family. Crowdion is an uncompleted (not passed through the whole crystal) plastic shift of close-packed atomic row fragments on one lattice parameter, which forms a delocalized interstitial atom in a crystal.

Finite size 2D crystals with a hexagonal lattice were considered. Interatomic interaction in the lattice was described with the Lennard-Jones centrally symmetric pair potential [6]. Interaction of each atom with all other atoms of crystal without potential «cut-off» at large distances was taken into account. Movement of atoms was described by the Newtonian mechanics equations. Simulation of crowdions in 2D crystals was carried out using methods of molecular dynamics and the modified high-speed Verlet algorithm [7]. To improve the accuracy of calculation of sums of a lot of numbers with floating point which differ significantly in magnitude, Kahan algorithm (compensated summation) [8] was applied. Parallel computations were performed using MPI technology [9].

As an initial atomic configuration of the crystal we considered a fragment of defect-free flat hexagonal lattice, inscribed in a circle of specified radius \( R \) with a center in one of the lattice sites. Value \( R \) was multiple of lattice parameter \( a \). The crowdion was created in the crystal center by incomplete shift of central close-packed row atoms. Thereafter the calculation of atomic relaxation was performed in conditions of zero temperature and external forces absence.

Atomic configurations of crystals with crowdion were obtained (Fig. 2) and its self-energy values \( U \) in crystals of various sizes \( R \) were calculated. Influence of value \( R \) on crowdion energy was ascertained. Energy \( U \) increases with \( R \) and at \( R \geq 50a \) it approaches to \( U \approx 3.4U_a \), where \( U_a \) – average value of energy per one atom in defect-free infinitely stretched crystal. The analysis of structural distortions caused by crowdions was carried out (Fig. 3) and anisotropic relative deformations were described. Fig. 4 shows coordinate dependencies of distortion tensor components \( w_{ij}(x,y) \). The elastic distortions fields caused by crowdion decrease under power laws when distance from crowdion center exceeds \( \approx 10a \), which corresponds to crowdion description within the linear elasticity theory.

The obtained results are important for development of physics of defects in low-dimensional nanocrystals.
2.2 Computer modeling of carbon nanotube complexes with polyoxymethylene oligomers

Urgent task to improve methods of new nanomaterials creation is conducting computer simulations using the capabilities of molecular dynamics and quantum chemistry to simulate self-assembly of nanoparticles. In this project we study self-organization of complexes formed by oligomers of organic polymer polyoxymethylene (POM) and single-walled carbon nanotubes (CNT). It is assumed that such complexes are formed during the manufacture of CNT-POM nanocomposite by sonication of CNT suspension in formalin followed by cryotechnological treatment consisting in a low-temperature polymerization and freeze drying.

Systems consisting of zigzag(10,0) CNT fragments of different lengths and POM$_n$ oligomers containing different number of monomers $n$ were investigated by molecular dynamics under vacuum and aqueous environment conditions. During the simulation procedure of the CNT-POM$_n$ system in vacuum the sliding of the polymeric chain over the nanotube surface was observed; separation of the chain from the surface did not happen.

The simulation of POM$_{20}$ complex with a small fragment of CNT terminated by hydrogen atoms in aqueous environment produced a new result. Namely, during the simulation a gradual penetration of the oligomer chain into the interior of the nanotube was observed (Fig. 5). The whole POM$_{20}$ chain was pulled inside the tube when the calculation time was long enough.

Discovered in the computer experiment noncovalent functionalization of the nanotubes by POM oligomers can explain the stabilization of CNT-POM water suspension: the presence of the hydrophilic polymer in the complex with hydrophobic CNT increases the hydrophilicity and solubility of the complex as a whole. This result is of practical interest for professionals involved in creation of nanocomposites based on CNTs and organic polymers.

Further, in the complex of CNT with POM$_n$, the POM$_n$ oligomer gains a specific conformation (Fig. 6, a): while in a single POM$_n$ chain oxygen atoms are arranged along a spiral, in the chain located inside the nanotube the oxygen atoms are deployed in one direction. The electrostatic potential distribution around the POM oligomer derived from its complex with the CNT (Fig. 6, b), as well as around the entire CNT-POM complex (Fig. 6, c), was mapped basing on HF/6-31++G (d,p) calculations.

Fig. 6, b shows that the row of oxygen atoms in the middle of the POM chain creates an extended zone of negative potential, and from Fig. 6, c one can see that inside the nanotube a kind of a dipole is formed.

Thus, the results obtained in the computer experiment have revealed some unexpected and unusual properties of CNT-POM complexes, which merit further investigations.
2.3 Quantum-mechanical calculations of tautomers and vibrational spectra of the halogenated uracil molecules isolated in the inert matrices

Quantum-mechanical calculations have played a prominent role in modern studies of the structure and physicochemical properties of the molecules that form the basis of living matter. Modern SMP-based clusters make possible calculations of the energy characteristics and vibrational spectra of a polyatomic bioorganic molecules using the different quantum mechanical program (eg: GAUSSIAN [10], PC GAMESS [11] or Firefly [12]) and DFT, MP2, CCSD, etc methods.

Halogenated uracil RNA bases are an important class of the biologically active compounds. They are used in basic research of the evolution of living organisms [13], the processes of mutagenesis [13, 14], the creation of new drugs, including anticancer drugs. Computer simulations have shown the possibility of the existence of 5-bromouracil as enol tautomer in the water clusters [14]. However, the diketo-form is dominant for molecules of uracil and 5-halo uracil located in an isolated state.

In our work were obtained FTIR spectra of molecules 5Br-Ur and 5J-Ur isolated in the inert matrices. Quantum-mechanical calculations of molecular structures and relative energies of the tautomers were carried out for the entire series of halogenated uracil: 5FUr, 5ClUr, 5BrUr and 5JUr. The DFT/B3LYP, MP2 and CCSD (T) methods with aug-cc-pVDZ, 6-311++G** and aug-cc-pVTZ. basis sets were used to calculate the energies.

Calculations of the relative energies showed the predominance of the keto form and the consequent reduction of its stability in the row: Ur, 5JUr, 5BrUr, 5ClUr, 5FUr. According to calculations, the total population of rare tautomeric forms should not exceed 0.1 % in the evaporation temperature range 420-480K, which correlates well with the experimental evaluation (0.2 %) [15].

The statistical analysis of differences between the experimental and calculated frequencies in the vibrational spectra was performed to optimize the quest of the characteristic frequencies of the rare tautomers. It was found that DFT/B3LYP method with 6-311++G(df, pd) basis set shows better statistic than the same method with aug-cc-pVTZ basis set [15]. Furthermore, the using of the 6-311++G(df, pd) basis set reduces the computation time of the vibrational spectra by a factor four.

Figure 7 shows fragments of experimental and calculated IR spectra in the region of stretching and deformation vibrations of the planar ring of 5-bromouracil molecule in the Ar matrix.

![Fig. 7. The region of stretching and deformation vibrations of the planar ring of 5BrUr molecules: 1 – FTIR spectra (Ar matrix, T=10 K), 2 – calculation spectra by using DFT/B3LYP/6-311++G(df, pd) method with frequency correction.](image_url)

Calculation of the vibrational spectra of 5-bromouracil in the clusters with 32 inert gas atoms (Ne, Ar, Kr) by using DFT/M06-2x/6-31++G(d, p) method showed a qualitative agreement with experiment.

For the first time, least square method with the using of polynomial for corrective scaling of calculated frequencies of vibrations was proposed to further improve the agreement of calculations and experiment [15]. It was shown that the correction of calculated frequencies with the polynomial of degree two (or three) permits to reduce the root-mean-square discrepancy between the calculated and experimental ones to 4–5 cm\(^{-1}\) [15].
2.4 DFT, MP2 and CCSD study of conformational structure, relative stabilities and vibrational spectra of matrix isolated simplest NAG peptide

Even small biological molecules, such as amino acids and short peptide, are highly flexible due to presence in their structure of multiple rotational degrees of freedom. Location of all possible conformers of these molecules is very important for correct prediction of their most stable forms as well for prediction of populations of the conformers. It requires exhaustive scan of the multidimensional potential energy surface using precise quantum mechanical methods. Recently we demonstrated than DFT/B3LYP and MP2 methods may be effectively applied to study conformational structure of flexible molecules of β-alanine and leucine [16, 17] and to predict their vibrational spectra of these compounds which were used to analyze experimental FTIR spectra of the molecules isolated in low temperature inert gas matrices.

In this project we used density functional theory (DFT) with functional B3LYP, Møller-Plesset second-order perturbation theory (MP2) and coupled cluster theory (CCSD) to study conformational structure and vibrational spectra of simplest peptide N-acetylglycine (NAG). Six 3D potential energy surfaces of NAG were calculated at the DFT/B3LYP/aug-cc-pVZD and MP2/aug-cc-pVZD levels of theory. Totally about 10000 geometry optimizations were performed. Modern parallel software implemented in the GAUSSIAN09 [10], PC GAMESS [11] and Firefly [12] program packages was used. Examples of the calculated potential energy surfaces are shown in Fig. 8. Minima on the surfaces correspond to stable NAG conformers.

![Fig. 8. 2D projections of 3D potential energy surfaces of NAG calculated at the B3LYP/aug-cc-pVZD (left) and MP2/aug-cc-pVZD (right) levels of theory (energy scales are in kJ/mol).](image)

The DFT method demonstrated best CPU efficiency with increase number of cores in the calculations: > 98% in the configuration of 16-cores/two SMP nodes (Firefly) and > 95% in the configuration 32-cores/two SMP nodes (Gaussian09). At the same time the MP2 method with semidirect algorithm which requires external disk storage of extremely large temporary files has poor efficiency (about 60%) in the configuration 16-cores/one node (Gaussian09). On other side fully direct MP2 algorithm which stores temporary data in memory was much more effective (CPU efficiency was about 90%) but required allocation of significant amount of RAM (96Gb per 16 core node). The CPU utilization by the CCSD method decreases significantly (<50%) due to huge external disk storage which cannot be avoided with this method.

3 Conclusion

The software packages have been used in the projects are the following: Gaussian, NWChem, NAMD, Firefly (PC-GAMESS), Gromacs, all being available either in local mode, or through virtual organizations of EGI: MolDynGrid (UA-Grid), CompuChemGridUA (UA-Grid), Gaussian (PL-Grid, qcg-icon).

On materials of researches with use of HPC-technologies for the last three years ILTPE stuff published more than 30 articles in international magazines.
4 Acknowledgments

The authors express their sincere gratitude to the Coordinating Committee of the UA-Grid program and managers of academic institutions for the use of grid clusters:

- B. Verkin Institute for Low Temperature Physics and Engineering of NASU;
- V. Glushkov Institute of Cybernetics of NASU;
- N. Bogolyubov Institute for Theoretical Physics of NASU;
- Institute of Molecular Biology and Genetics of NASU;
- Institute for Scintillation Materials of NASU.

The authors hope for further successful international cooperation with the Polish grid and thank its managers for the opportunity to use the virtual organization Gaussian resources.

Some of the researches were performed with using resources of the Computer center Arizona University (Arizona, USA) and of the Supercomputing Center of Lomonosov Moscow State University [18] (Moscow, Russia). The authors are grateful for the opportunity to use the resources.

References

Применение гетерогенных вычислительных систем для описания процессов в системах виртуальной реальности

Белан В.И. 2, Белоус Л.Ф. 2, Поляков В.М. 1, Торгонин Е.Ю. 1, Хутайфа А. 1

1 Белгородский государственный технологический университет им. В.Г. Шухова, ул. Костюкова 46, Белгород, Россия
2 Физико-технический институт низких температур им. Б.И. Веркина НАН Украины, пр. Ленина, 47, Харьков, Украина

belan@ilt.kharkov.ua, belous@ilt.kharkov.ua, p_v_m@mail.ru, e.yudek@gmail.com, huthaifa1984@gmail.com

Аннотация. Большинство современных вычислительных систем, будь то персональные компьютеры или массивные вычислительные системы, являются гетерогенными. Применение подобных систем даёт множество преимуществ в таких областях как создание систем виртуальной реальности, но в то же время усложняет процесс проектирования программного обеспечения. Исследованиям по созданию эффективных решений для таких систем уделяется большое внимание в Белгородском государственном технологическом университете им. В.Г. Шухова. Использование HPC систем при создании приложений реального времени позволяет существенно увеличить возможности и эффективность программных решений. В данной работе представлены направления и результаты последних исследований по реализации проектов с использованием HPC, выполняемых в БГТУ им. В.Г. Шухова.

Ключевые слова
Системы виртуальной реальности, гетерогенные вычислительные системы, моделирование и визуализация жидкостей, PBR.

1 Введение

Реализация систем виртуальной реальности требует от разработчиков применения наиболее простых и быстрых алгоритмов описания объектов и процессов, а также методов их визуализации. В то же время аппаратные средства вычислительных систем постоянно развиваются, центральные процессоры и графические ускорители становятся все быстрее и быстрее, что позволяет программистам реализовать все более сложные и физически корректные модели описания поведения объектов реального мира и их визуализации.

В настоящее время мощность многих вычислительных систем позволяет реализовать модели PBR (Physically Based Rendering) для использования в системах виртуальной реальности [1]. Реализация реалистичной визуализации всегда подразумевает использование некоторых базовых физических моделей описания процессов или явлений реального мира. Термин «Physically Based» указывает на использование только физически корректных моделей или приближений, которые являются широко известными и апробированными во многих областях науки и техники. Использование таких методов становится всё более популярным при создании систем виртуальной реальности.

В тоже время физически корректные модели описания многих явлений являются сложными для реализации в системах реального времени, поэтому в целях соблюдения некоторой точности и практичности, создаются гибридные модели, сочетающие различные методы.

2 Основная часть

Описанные в данной работе задачи решаются с использованием различных гетерогенных вычислительных систем, различной конфигурации и мощности, что позволяет полноценно оценить эффективность предлагаемых решений.
Для создания фотореалистичных изображений, важное значение имеет реализация непрямого освещения, но его вычисление является дорогим с вычислительной точки зрения и очень сильно зависит от сложности сцены. В то время как его вычисление в режиме off-line и хранение в виде текстур является неприемлемым для большинства систем виртуальной реальности. Такие приложения, как тренажеры и симуляторы требуют реального времени обработки информации или интерактивных подходов для уче́та непрямого освещения в трёхмерной сцене.

На данный момент в БГТУ им. В.Г. Шухова проводятся исследования по возможности создания новых методов и алгоритмов вычисления непрямого освещения в реальном времени, позволяющих избежать использования ресурсоёмких методов предварительной обработки информации. Наиболее перспективным можно назвать направление по адаптации метода Voxel Cone Tracing, описанного в работе [2], для применения в различных областях науки и техники. Основной целью является необходимость в представлении и обработке информации об объектах в трёхмерной сцене при освещении, как объектах обладающих большинством свойств материалов реального мира.

На данный момент реализована система виртуальной реальности, реализующая PBR и представляющая собой морской симулятор, который использует в своей работе гибридный метод представления моря на основе гидродинамической модели [3]. Как показали исследования, предложенный метод может применяться в системах реального времени за счёт использования средств гетерогенной среды. Данный метод позволяет учесть, что волны различной частоты имеют различную скорость распространения и могут иметь различную природу возникновения, поэтому задачу представления и визуализации моря можно разделить на процессы и выполнять вычисления параллельно. В рамках предложенного метода, поверхностные волны могут быть представлены в виде суммы 3 компонент, предусматривающих обработку различной информации:

$$f(X, t) = C_0 \cdot \text{Ext(Flow)} + B_s \sum_{i=1}^{n} R(k, t) e^{-i \omega t} + A_s \sum_{i=0}^{M} a_i \cdot f_{\text{noise}}(2^i, X, t),$$ (1)

где $X$ - горизонтальная позиция точки $(x,y)$, высоту которой мы оцениваем; $\text{Ext(Flow)}$ - высота изо-постоянной в точке $X$, построенная в результате моделирования жидкости методом SPH; $R(k, t)$ - комплексное число, представляющее в амплитуду и фазу волн, и амплитуду и фазу волны $k$ во время $t$; $n$ - количество гармоник выбираемых из спектра описывающего ветровое волнение, $M$ - количество октав шума применяемого для имитации рябь; $\alpha$ - параметр стойкости, определяющий весовой коэффициент с которым $i$-ая октава шума влияет на итоговый результат; параметры $A_s, B_s, C_s$ - некоторые коэффициенты масштабирования, позволяющие учитывать степень вклада в итог визуализации, соответственно SPH метода описания среды, статистического метода описания ветрового волнения поверхности среды и шумовой функции, имитирующей высокоочастотные волны, такие как рябь.

Результаты работы разработанного симулятора, использующего данную модель, показаны на рисунке 1. Для реализации отражений при освещении реализована модель IBL [4, 5] т.е. освещения на основе карт окружаения. Которая в свою очередь является частью используемой PBR модели.

Рис. 1. Примеры визуализации моря в разработанном морском симуляторе.

Производительность разработанного программного комплекса зависит от двух основных факторов: эффективность реализованных алгоритмов и время затраченное на планирование задачи в гетерогенной среде.

Разработанные алгоритмы были протестированы на двух тестовых вычислительных системах. Первая система состоит из CPU 2.4GHz Intel Core 2 Quad Q6600 - четырёхъядерный процессор c четырьмя гигабайтами RAM, системной шиной пропускной способностью ~18 GB/s, и графическим ускорителем AMD HD 4850 (~1 TFLOPS в операциях с 32 битными числами).

Вторая система состоит из CPU 3.3GHz Intel Core i5-3550 - четырёхъядерный процессор с 8 гигабайтами RAM, системная шина ~22 GB/s, и AMD HD7850 2Gb VRAM (1,76 TFLOPS). Тестовые системы похожи, но
отличаются производительностью отдельных компонент, что позволяет оценить эффективность как масштабирования приложения, так и использования отдельных вычислительных устройств в системе.

Три алгоритма построения и визуализации поверхности моря сравнивались между собой в вычислениях с одинарной точностью с эталонным четырёх поточным приложением на CPU Intel Core 2 Quad Q6600, что отражено на рисунке 2. Большой перерасход памяти в гетерогенной среде (процессор Intel Core i5 и видеокарта HD7850) вызван большим количеством доступных вычислительных блоков и необходимостью дублировать большее количество информации, но оправдан и позволяет получить существенный прирост производительности за счёт экономии времени на работу средств синхронизации. Таким образом, предлагаемый подход хорошо масштабируется, эффективен в гетерогенной среде, позволяет использовать произвольное количество потоков исполнения.

Рис. 2. Тестирование масштабируемости разработанного метода

Использование в качестве дополнительного исполнителя GPU позволяет существенно увеличить производительность по сравнению с системой, использующей множество исполнителей на базе CPU. Это происходит даже в случаях, когда передаются данные между системной памятью и памятью GPU при каждом вызове функции визуализации.

В таблице 1 отражено количество кадров в секунду выполняемых при визуализации жидкости (поверхности океана) с геометрической сеткой размерностью 256х256 вершин и набором 100 000 частиц для метода SPH. Эксперимент проведён на второй, более мощной тестовой системе, где исполняется разработанный гибридный метод (Hb) описания и визуализации жидкой среды. Для сравнения приведены широко известные FFT-метод [6] и SPH [3], использующий 1 миллион частиц.

<table>
<thead>
<tr>
<th>Общий размер карт поверхности</th>
<th>Объёмный интеграл</th>
<th>CPU</th>
<th>GPU</th>
<th>CPU+GPU</th>
</tr>
</thead>
<tbody>
<tr>
<td>8192х8192</td>
<td>~3</td>
<td>~4</td>
<td>~5</td>
<td>~0</td>
</tr>
<tr>
<td>4096х4096</td>
<td>~7</td>
<td>~9</td>
<td>~11</td>
<td>~0</td>
</tr>
<tr>
<td>2048х2048</td>
<td>~17</td>
<td>~30</td>
<td>~10</td>
<td>~1</td>
</tr>
<tr>
<td>1024х1024</td>
<td>~41</td>
<td>~57</td>
<td>~64</td>
<td>~3</td>
</tr>
<tr>
<td>512х512</td>
<td>~93</td>
<td>~120</td>
<td>~145</td>
<td>~7</td>
</tr>
</tbody>
</table>

Как видно из таблицы, разработанный метод даёт возможность используя преимущества методов, основанных на моделях вычислительной гидродинамики, в режиме реального времени реалистично визуализировать жидкие среды, за счёт разработанного метода синтеза поверхностного волнения.

Использование в системе только CPU устройств имеющих доступ к общей памяти практически не накладывает дополнительных расходов при синхронизации. Использование только GPU устройства для большинства расчётов накладывает ряд издержек, не позволяющих максимально эффективно использовать все доступные ресурсы. Увеличение издержек связано с невозможностью переупорядочивания задач средствами GPU, с издержками передачи данных через шину PCI-E, а также особенностями доступа к памяти GPU. Совместное использование CPU и GPU вычислительных устройств позволяет несколько минимизировать данные издержки и лучшим образом распределять нагрузку.

Но как показал эксперимент, использование гетерогенной вычислительной системы с GPU и CPU исполнителями, так же не может идеально масштабироваться. Заметен относительно небольшой рост скорости
выполнения по сравнению с использованием только GPU в качестве исполнителя. Причина этого кроется в невозможности, в некоторых случаях, разделить связанные данные в отдельные кластеры, что вызывает рост объёмов передаваемых по PCI-E шине данных и соответственно увеличивает рост затрат времени на планирование и подготовку данных.

Рисунок 3 показывает результат проверки издержек, возникающих при распараллеливании вычислений, в системе с задействованными от одного до восьми CPU исполнителей (потоков) и двух GPU исполнителей. Проверялась эффективность масштабирования спектрального метода Тессендорфа (FFT), разработанного гибридного метода и SPH-подхода с числом частиц в 1,5 раза большим, чем в гибридном. Выявлено, что среднее время, потраченное на планирование задачи не значительно, разработанный программный комплекс для параллельного выполнения задачи описания динамики и визуализации поверхности жидкой среды позволяет налагать лишь небольшие издержки на планирование и распределение ресурсов.

Рис. 3. Среднее время планирования единственной задачи в зависимости от количества исполнителей

Увеличение издержек при использовании GPU в качестве девятого и десятого исполнителей связано с издержками передачи данных через шину в локальную память GPU, к тому же метод SPH не очень хорошо масштабируется в гетерогенной среде, когда требуется перемещать большие объёмы взаимосвязанной информации в память различных вычислительных узлов.

3 Заключение

Выполненное тестирование и анализ работы разработанного программного комплекса позволяет сделать вывод о том, что создание гибридных, хорошо распараллеливаемых методов реализации освещённости и описания процессов реального мира, на примере жидких сред, является эффективным подходом создания систем виртуальной реальности в настоящее время.

Разработанный подход хорошо масштабируется и ориентирован на работу в гетерогенных вычислительных системах. Используя принцип разбиения задачи по процессам в зависимости от их природы явления, удалось достичь высокой эффективности при обработке информации.

Учитывая полученные результаты, в БГТУ им. В.Г. Шухова запланированы дальнейшие работы по созданию PBR системы для гетерогенных систем виртуальной реальности, позволяющей увеличить качество визуализации с целью увеличения эффекта погружения человека в виртуальную среду.

Список литературы

Molecular dynamics study of irradiation damage in LaPO$_4$ and YbPO$_4$

Grechanovsky A.E., Ogar T.V.

M.P. Semenenko Institute of Geochemistry, Mineralogy and Ore Formation of NAS of Ukraine, 34 Palladina av., Kiev, Ukraine
grechanovsky@gmail.com

Abstract. The radiation resistance of the monazite LaPO$_4$ and the compound YbPO$_4$ (zircon structure type) has been investigated using the computer simulation. The number of Frenkel pairs, which are formed in the structure of these minerals after the passage of a primary knock-on thorium atom with an energy of 30 keV, has been calculated by the molecular dynamics method. The formation of Frenkel pairs and their recombination in the motion of recoil nuclei in the structure of the studied minerals have been discussed. It has been shown that the probability of the “survival” of Frenkel pairs in the LaPO$_4$ monazite is significantly lower than in the YbPO$_4$ compound. The tendency of these minerals toward amorphization under radiation damage has been described numerically. The obtained results have demonstrated that one of the main factors determining the radiation resistance of orthophosphates LnPO$_4$ is the type of crystal structure, and the compounds with the monazite structure are more radiation resistant than the compounds with the zircon structure.

Keywords
Radiation resistance of minerals, semiempirical interatomic potential method, grid-calculations, computer simulation, method of molecular dynamics

1 Introduction

Over recent decades, in a number of countries there has been traced a tendency to increase the use of electricity generated by nuclear power plants. In particular, according to the International Atomic Energy Agency (IAEA) data, in 2012 the share of electricity generated by nuclear power plants is 75% in France, 46% in Ukraine, 19% in the United States, and 18% in the Russian Federation [1]. On the other hand, the prospects for the development of nuclear power engineering are associated with the effective management of nuclear waste.

The development of nuclear power engineering raises a number of problems relating to the disposal of long-lived radioactive waste and plutonium. One of the main problems in this respect is the choice of radiation resistant matrices, which, in contact with long-lived high-level radioactive waste, for a long time will not change their physical and chemical properties. At present, alumino-phosphate or borosilicate glasses have been used as matrices for spent fuel. However, high-level radioactive waste can be stored in these matrices for a time of no longer than 30–40 years. This is the reason that the search for matrices with efficient performance characteristics has been actively continued. It has been found that crystalline ceramic materials are significantly better suited for the utilization of high-level radioactive waste. To date, a number of ceramic materials have been developed for the disposal of high-level radioactive waste and plutonium. Extensive studies have been performed on materials such as zircon ZrSiO$_4$, pyrochlores Gd$_2$Ti$_2$O$_7$ and Gd$_2$Zr$_2$O$_7$, monazites (La,Ce,Nd)PO$_4$, zirconolite CaZrTi$_2$O$_7$, perovskite CaTiO$_3$, and other complex oxides, as well as rutile TiO$_2$ and baddeleyite ZrO$_2$.

Many researchers have considered zircon as a promising matrix for the disposal of nuclear fuel and weapon-grade plutonium [2–5]. However, over geological time, the alpha decay of uranium and thorium atoms leads to the damage of the structure of zircon and its transition from the crystalline state to the X-ray amorphous (metamict) state. Each act of alpha decay results in the formation of an alpha particle and a heavy recoil atom [5]. Alpha particles with an energy of 4.2–5.5 MeV, as was noted in [4], displace approximately 100 atoms in the end of the path with a length of 10–20 μm, whereas heavy recoil atoms with an energy of 70–90 keV displace several thousand atoms within an interval of 20 nm.

A promising alternative for zircon can be natural orthophosphates LnPO$_4$ (Ln is a lanthanide) with the structures of zircon and monazite, as well as their artificial analogues. In contrast to zircon, these compounds are very rare in the
metamict state despite the significant amount of uranium and thorium atoms [6]. The orthophosphates containing heavy rare-earth elements (Tb, Tm, Yb, Lu) crystallize in the tetragonal zircon structure (I4_1/amd) [7]. The monazites containing lighter and larger rare-earth elements (from La to Dy) are characterized by the monoclinic monazite structure (P2_1/n) [8]. Earlier, the radiation resistance of these compounds was experimentally investigated by Meldrum et al. [6, 9]. The results of these studies showed that orthophosphates with the monazite structure have a higher radiation resistance than the orthophosphate containing heavy rare-earth elements. In particular, the critical temperature of amorphization (the temperature above which the material does not transform into an amorphous state under irradiation with a beam of heavy particles) is $T_c \approx 1000$ K for zircon [9], $570$ K for the YbPO$_4$ compound with the zircon structure, and $333$ K for the LaPO$_4$ monazite [6]. The processes of forming and annealing a damaged region, resulting from the alpha decay, usually occur for a few tens of picoseconds. Therefore, in order to investigate the damage of minerals due to the alpha decay, in addition to experimental studies it is extremely useful to carry out computer simulation experiments.

The purpose of this work was to perform a computer simulation for investigating the mechanisms of formation of a displacement cascade and the subsequent relaxation processes in the structure of the LaPO$_4$ monazite and in the YbPO$_4$ compound with the zircon structure as a result of the alpha decay due to the recoil of the nucleus.

2 Simulation technique

The molecular dynamics (MD) method consists in calculating trajectories of the motion of all atoms involved in a system on the basis of Newton’s second law. The initial data are taken as the initial coordinates and velocities of all the atoms and the interatomic interaction potentials. In addition to the Coulomb interactions of all electrostatic charges between themselves, the interatomic interaction potential takes into account the repulsion of electron shells of the atoms and the dipole–dipole interaction between the atoms in terms of the short-range interaction Buckingham potential:

$$V(r) = A \cdot \exp(-r/\rho) - C \cdot r^{-6},$$  (1)

where $r$ is the distance between two atoms (Å), $A$ is the pre-exponential factor for the term characterizing the repulsion (eV), $\rho$ is the stiffness parameter (Å), and $C$ is the force parameter of the van der Waals interaction (eV·Å$^6$).

Parameters specified in (1) were taken from work [10]. Optimization of these structures was made using experimental values of unit-cell parameters, atom coordinates, elastic constants, and thermodynamic properties.

In the structure of the monazite, we chose a fragment containing approximately 5 million atoms. One of the lanthanum atoms was replaced by a thorium atom. At small interatomic distances (less than 1 Å), we used the internuclear repulsion potential ZBL, which was introduced to correctly take into account the strong internuclear repulsion [11]. The simulation time step was $0.5$ fs.

As a result of a critical consideration of various programs, we dwelt on the DL_POLY program complex [12], elaborated for simulation of structural fragments of minerals, macromolecules, polymers, and ion systems. This program complex gives an opportunity to perform computer simulations (investigation of structures due to alpha decay of actinides), study of processes in minerals, study of forming migrations of point defects in these minerals. For the realization of calculations the web-sites of uagrid.org.ua and grid.inpracom.kiev.ua were used. All calculations were executed in the virtual organization «GEOPARD», organized by Glushkov Institute of Cybernetic of NAS of Ukraine, M.P. Semenenko Institute of Geochemistry, Mineralogy and Ore Formation of NAS of Ukraine and S.I. Subbotin Institute of Geophysics of NAS of Ukraine.

3 Results and discussion

The motion of a primary knock-on atom leads to its collision with other atoms of the system. These atoms are displaced from their equilibrium positions, begin to move, and, in turn, displace other atoms. This stage can be referred to as ballistic. This process results in the creation of an amorphous zone surrounded by relatively undistorted regions (point defects). A substantial fraction of displaced atoms returns to their original positions during a period of several picoseconds. Other atoms form a displacement cascade (Fig. 1).

For a more detailed investigation of the processes occurring in a cascade of displaced atoms, we examined the differences in the kinetics of accumulation and recombination of defects for the two minerals.

The estimated probability of the “survival” of Frenkel pairs of the oxygen atoms (FP(O)) for the studied structures depends on the displacement of the oxygen atoms: with an increase in the distance between the vacancy and the displaced atom (dV1), the probability of the recombination of Frenkel pairs decreases (Fig. 2). These data also indicate...
that the probability of the “survival” of Frenkel pairs is significantly lower in the structure of the LaPO₄ monazite than in the YbPO₄ compound.

\[
\delta = \frac{1}{E_{PKA}} (E_{FP}(\text{Ln}) \cdot N_f(\text{Ln}) + E_{FP}(\text{P}) \cdot N_f(\text{P}) + E_{FP}(\text{O}) \cdot N_f(\text{O})) ,
\]

where \(E_{FP}(\text{Ln}), E_{FP}(\text{P}), \) and \(E_{FP}(\text{O})\) are the energies of Frenkel pairs of the atoms Ln, P, and O, respectively; \(N_f(\text{Ln}), N_f(\text{P}), \) and \(N_f(\text{O})\) are the numbers of Frenkel pairs of the atoms Ln, P, and O, respectively, at the end of the simulation; and \(E_{PKA}\) is the energy of the knock-on atom.

The calculations have demonstrated that the parameter \(\delta\) is equal to 0.39 for the LaPO₄ monazite, whereas for the YbPO₄ orthophosphate, it is approximately equal to unity. These values are consistent with the experimental data: the critical temperature of amorphization is equal to 570 K for the YbPO₄ compound and 333 K for the LaPO₄ monazite [6].

\[\text{Figure 1. Radiation damage, produced by 30 keV Th recoil in monazite at the peak of the damage (a) and after structure relaxation (b).}\]

\[\text{Figure 2. Relative number of } FP(O) \text{ on the distance between the vacancy and the displaced atom for LaPO}_4 \text{ and YbPO}_4.\]

\[\text{In [13], we proposed the parameter } \delta, \text{ which reflects the tendency of minerals toward amorphization under radiation damage. This parameter characterizes a part of the energy of a knock-on atom, which is consumed for the formation of Frenkel pairs in a cascade of displaced atoms, and can be calculated according to the formula}\]

\[\text{4 Conclusions}\]

The mechanisms of radiation-induced damages in the LaPO₄ monazite and in the YbPO₄ compound with the zircon structure as a result of the alpha decay due to the recoil of the nucleus have been investigated using the molecular dynamics computer simulation. The kinetics of accumulation and recombination of Frenkel pairs in the cascade of displaced atoms after the passage of the primary knock-on thorium atom with an energy of 30 keV in the monazite structure were calculated.
We introduced a parameter that reflects the tendency toward amorphization of the minerals under radiation damage. This parameter characterizes a part of the energy of a knock-on atom, which is consumed for forming Frenkel pairs in a cascade of displaced atoms. The results have shown that this parameter for the LaPO$_4$ monazite is equal to 0.39, and for the YbPO$_4$ orthophosphate, it is approximately equal to unity. This is in agreement with the experimental data.

The obtained results have also demonstrated that one of the main factors determining the radiation resistance of the orthophosphates LnPO$_4$ is the type of structure. Compounds with the monazite structure are more radiation resistant than compounds with the zircon structure. The higher radiation resistance of the LaPO$_4$ monazite as compared to the YbPO$_4$ compound is most likely also associated with the longer relaxation of the monazite structure.

The results of this investigation can be used to solve the fundamental and applied problems associated with the isolation and disposal of high-level radioactive waste and, in particular, to evaluate the radiation resistance of orthophosphate matrices proposed for waste disposal. The computer simulation performed in this work can be helpful in analyzing and predicting the behavior of matrices under radiation effects. The obtained results can also be useful in saving time and financial resources and, eventually, in choosing the most suitable matrices.

5 Acknowledgments

The research has been performed within the framework of project of research works of young scientists of National Academy of Sciences of Ukraine (research work «Research of the radiation-enhanced processes in zircon in connection with the decision of geochronological problems and searching of rare-earth elements»).

References

Study of pattern identification processes in highly productive human synaptic networks

Khodakovskyi N. I., Zolot A. I., Merjvinsk yi P. A.

Institute of Cybernetics by Glushkov NAS of Ukraine,
03687, Av. Glushkov, 40, Kyiv-187, Ukraine

nhodak@ukr.net

1 Introduction

When using distributed computing technologies to solve large computationally complicated problems, a number of issues, related to the coordination of a huge number of interacting computers and data flow between them, arise. One solution to these problems is the system of meta-computing as a software tool, designed to organize distributed heterogeneous computing environments and perform calculations in such environments [1].

The basic components of the system are the task server and the client. Task server is responsible for splitting the particular application task into independent computing portions, their distribution into computational nodes and combining the results.

Due to the high-speed nature of computations, similar to the ones produced by brain, it is possible to use a highly parallel architecture for artificial highly productive synaptic networks using neuromorphic chips [2].

2 Use of neuromorphic chips in bionic highly productive synaptic networks

Silicon chips, whose work can be controlled as well as the vital activity of neurons, can be used as neuromorphic chips. Such chips, consisting of individual neuro modules, have their own memory and other characteristics of living neurons. The need for such chips arose from the fact that modern microprocessors with Boolean logic mishandle emulation of “neurons” and require resources of a supercomputer to simulate operation of the brain of bio systems at the level of insects.

On the crystal of neuromorphic chip electronic analogs of neurons are implemented. Using all available designs of artificial "neurons", we managed to build subdivided neural networks, which for their work require not only computational abilities, but also two types of memory, long-term and short-term [2]. Such neuromorphic approach allowed to give the chip analytical capabilities and implement truly intelligent algorithms of decision making, that cannot be or are very difficult to obtain, using the principles of linear logic upon which all programs for microprocessors are based.

Let us consider the possibilities of using the considered chips in personal computers of sixth generation, which constitute an integrated system, the hardware components of which are completely based on artificial neural networks (ANN).

Accelerators of computation here contain standard reprogrammable processors, improved performance of which is provided by paralleling of computing repetitive operations with help of ANN. Required training for this is a process of self-organization of distributed computing environment - "neural" ensembles. In distributed "neural" networks parallel information processing takes place, which is accompanied by constant training, driven by the results of this processing.

The main thing that unites the brain and neurocomputers - focus on processing images. Let us consider biological prototypes and focus on the basic principles of distributed data processing.

Let us consider what the effect of back propagation of error in training of bionic neural networks gives. It is known that the basis for training of neural networks is the gradient optimization method - the iterative change of synaptic weights, gradually lowering error of processing of training examples by the neural network [3].

Weight changes occur due to local gradient of the error function. Effective method of finding the gradient is the algorithm of back propagation of error. Differences of the neural network responses from the correct responses are determined on the output layer of "neurons" and are distributed across the network towards the stream of signals.

Massive parallelism of neurocomputing, required for effective image processing, is provided by locality of information processing in neural networks. Each "neuron" responds only to information, supplied to it by the associated neurons, without an appeal to the general plan of calculations for conventional ECM [4]. Therefore neural network algorithms are local, and "neurons" can operate parallely.

Absence of a global plan of computations in neural networks involves special type of their programming. Each "neuron" changes its adjustable parameters in the form of synaptic weights according to the incoming local information. This information is determined by the outputs of the network and reflects the effectiveness of its work as a whole. It
extends from the outputs to the inputs, towards the stream of input signals. Therefore, the basic algorithm of training networks is called back propagation of errors. Since the error passes through the same synaptic links between "neurons", the greatest signal about error is obtained by "neurons", which gave the greatest contribution to the wrong answer.

As a result, such least trained "neurons" are trained the most. This is a very simple and effective training principle. Therefore, in the process of self-teaching, biological neural networks use such effective sensory processing algorithms [5].

3 The role of software agents of synaptic human networks in the formation of molecular storage objects in the brain cortex

Let us consider memorizing process that occurs in nerve cells by increasing effectiveness of connections, called synapses. In the case of short-term memory effect only lasts for minutes or hours. In the case of long-term memory, synaptic connection is increased for a long time.

Memory is formed as a consequence of the signals passing through the synapses. In the synapse, which got enough stimulation, molecules of signal substance are produced. Once the synapse efficiency is improved, it can maintain the memory for a certain time until the signal substance is located on the way to the nucleus of the nerve cell.

This substance activates certain genes there, which are necessary for the synthesis of proteins that strengthen synaptic connection for a long time. Thus the transcription factor CREB was discovered and described, as this signal substance described above, that performs the key role in converting short-term memory into long-term [3].

From the viewpoint of molecular cybernetics, such transcription factor is a software agent in implementing the "software" of the neuron cell and synaptic network in processing external information by the latter.

Software agents are essentially complexes of sequences of the DNA genetic information, connected with specific proteins. Such complexes are able to manage processes in the cell nucleus when searching specific DNA sequences, and connect to them. In practice, they are switches that control the transcription of genes. Therefore, activation of such software agents in a neuron leads to gene activation that leads to the production of proteins that intensify synaptic connection, and conversion of short-term memory into long-term memory.

In this work we show the role of software agent in the processing of information in the synaptic human networks. Thus, when fixing the input signal at the input membrane of the synapse, software agent passes synapse, output synapse membrane, and is transferred into the nucleus of the neuron.

After receiving information from DNA structures in the nucleus of the neuron, the software agent is combined with the template RNA and is transferred to the ribosome of the neuron, where the synthesis of proteins, transferred into the synapse to intensify synaptic connections, is provided. And, accordingly, this completes the creation and preservation of the storage molecular object in this synaptic network.

4 Conclusion

In this study realization principles of highly productive processes of pattern identification in human synaptic networks are analyzed. It is shown that such processes are implemented using different software agents and methods to provide information processing with a high degree of paralleling.

On basis of bionic approach to explaining the formation of molecular memory in synaptic human networks, we can suggest ways to use software agents in artificial neural networks of new generations of neurocomputers. For work and life using the Internet, a new approach to software development is required.

Personal assistants to help organize personal information filtering are required. In place of the passive object interface, you can use the active agent's interface. The distinguishing feature of such software agents, as well as in natural synaptic networks – the urge to better understand what is required of them. Therefore neural networks, capable to generalize examples by training discussed above are completely natural component of software agents.

Very high-performance computers, on which it will be possible to implement high-tech artificial intelligence systems, and that will continue to work even in case of failure of some components of the computer, can be the result of the appliance of neuromorphic chips, as well as in the human brain, which functions, losing approximately one million neurons every day.

Literature

Суперкомп’ютери: огляд нових рейтингів та ефективних прискорювачів

Комухаєв Е.Г., Черепинець В.В.

Інститут кібернетики ім. В.М. Глушкова НАН України, просп. Глушкова, 40, Київ, Україна

Анотація. Підвищення показників сучасних суперкомп’ютерів забезпечило збільшення кількості ядер, ефективних прискорювачій, нові технології обміну з пам’ятю, адаптацію до пріоритетних задач.

Ключові слова
Гібридизация, прискорювачі, масштабування, потокові технології, реконфігурація, СКІТ-4, ПФлопс, МФлопс/Вт, GTEPS.

1 Оновлені списки рейтингів


Виділяється новачок в лідируючий десятку (шосте місце) списку Top500 – суперкомп’ютерна система Швейцарського національного центру CSCS, яку вже називають «зеленим лідером світу». Ця система має високий показник енергозбереження в новому списку рейтингу Green500 – 3185,91 МФлопс/Вт при дуже високому рівні загальної потужності (1753,66 кВт).

Новий список рейтингу Green500 [2] кардинально оновився, фіксуючи значний розвиток енергозберігаючих систем. Найвищий показник – 4,5 ГФлопс/Вт тепер має невелика токійська система Tsubame-KFC.

В табл. 1 представлено чільну десятку систем, всі вони використовують прискорювачі NVidia.

<table>
<thead>
<tr>
<th>Позиція в Green500</th>
<th>МФлопс/Вт</th>
<th>Організація</th>
<th>Комп’ютер</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4503.17</td>
<td>Центр GSIC, Токійський технологічний інститут</td>
<td>TSUBAME-KFC - LX 1U-4GPU/104Re-1G Cluster, Intel Xeon E5-2620v2 6C 2.100GHz, Infiniband FDR, NVIDIA K20x</td>
</tr>
<tr>
<td>2</td>
<td>3631.86</td>
<td>Кембриджський університет</td>
<td>Wilkes - Dell T620 Cluster, Intel Xeon E5-2630v2 6C 2.600GHz, Infiniband FDR, NVIDIA K20</td>
</tr>
<tr>
<td>3</td>
<td>3517.84</td>
<td>Центр обчислювальних наук, Університет Цукуба</td>
<td>HA-PACS TCA - Cray 3623G4-SM Cluster, Intel Xeon E5-2680v2 10C 2.800GHz, Infiniband QDR, NVIDIA K20x</td>
</tr>
<tr>
<td>4</td>
<td>3185.91</td>
<td>Швейцарський національний суперкомп’ютерний центр (CSCS)</td>
<td>Piz Daint - Cray XC30, Xeon E5-2670 8C 2.600GHz, Aries interconnect, NVIDIA K20x</td>
</tr>
<tr>
<td>5</td>
<td>3130.95</td>
<td>РОМЕО НРС Центр, Шампань, Ардена</td>
<td>Romeo - Bull R421-E3 Cluster, Intel Xeon E5-2650v2 8C 2.600GHz, Infiniband FDR, NVIDIA K20x</td>
</tr>
<tr>
<td>Позиція в Green500</td>
<td>МФлопс / Вт</td>
<td>Організація</td>
<td>Комп’ютер</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------</td>
<td>-------------</td>
<td>------------</td>
</tr>
<tr>
<td>6</td>
<td>3068.71</td>
<td>Центр GSIC, Токійський технологічний інститут</td>
<td>TSUBAME 2.5 - Cluster Platform SL390s G7, Xeon X5670 6C 2.930GHz, Infiniband QDR, NVIDIA K20x</td>
</tr>
<tr>
<td>7</td>
<td>2702.16</td>
<td>Університет Аризони</td>
<td>iDataPlex DX360M4, Intel Xeon E5-2650v2 8C 2.600GHz, Infiniband FDR14, NVIDIA K20x</td>
</tr>
<tr>
<td>8</td>
<td>2629.10</td>
<td>Max-Planck-Gesellschaft MPI/IPP</td>
<td>iDataPlex DX360M4, Intel Xeon E5-2680v2 10C 2.800GHz, Infiniband, NVIDIA K20x</td>
</tr>
<tr>
<td>9</td>
<td>2629.10</td>
<td>Фінансова установа</td>
<td>iDataPlex DX360M4, Intel Xeon E5-2680v2 10C 2.800GHz, Infiniband, NVIDIA K20x</td>
</tr>
<tr>
<td>10</td>
<td>2358.69</td>
<td>CSIRO</td>
<td>CSIRO GPU Cluster - Nitro G16 3GPU, Xeon E5-2650 8C 2.000GHz, Infiniband FDR, NVIDIA K20m</td>
</tr>
</tbody>
</table>

Найбільший суперкомп’ютер України СКІТ-4 Інституту кібернетики ім. В.М. Глушкова НАНУ зараз модернізується, поки що має такі енергопоказники: 720 МФлопс/Вт, рівень енергоспоживання 25 кВт [3].

42-й список основного суперкомп’ютерного рейтингу Top500 у листопаді 2013 року незначно оновився в чільних рядах. В табл. 2 наведено показники чільної десятки систем нової редакції Top500 [1].

Таблиця 2. Перші 10 позицій рейтингу Top500 за листопад 2013 р.

<table>
<thead>
<tr>
<th>Позиція</th>
<th>Організація</th>
<th>Комп’ютер</th>
<th>Ядер</th>
<th>Макс. продуктивність, ТФлопс</th>
<th>Пікова продуктивність, ТФлопс</th>
<th>Потужність, кВт</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Національний суперкомп’ютерний центр в Гуангжоу, Китай</td>
<td>Tianhe-2 (MilkyWay-2) - TH-IVB-FEP Cluster, Intel Xeon E5-2692 12C 2.200GHz, TH Express-2, Intel Xeon Phi 31S1P NUDT</td>
<td>3120000</td>
<td>33862.7</td>
<td>54902.4</td>
<td>17808</td>
</tr>
<tr>
<td>2</td>
<td>DOE/SC/ Національна лабораторія Оак Ридж, США</td>
<td>Titan - Cray XK7, Opteron 6274 16C 2.200GHz, Cray Gemini interconnect, NVIDIA K20x Cray Inc.</td>
<td>560640</td>
<td>17590.0</td>
<td>27112.5</td>
<td>8209</td>
</tr>
<tr>
<td>3</td>
<td>DOE/NNSA/ LLNL, США</td>
<td>Sequoia - BlueGene/Q, Power BQC 16C 1.60 GHz, Custom IBM</td>
<td>1572864</td>
<td>17173.2</td>
<td>20132.7</td>
<td>7890</td>
</tr>
<tr>
<td>4</td>
<td>RIKEN Інститут обчислювальних наук (AICS), Японія</td>
<td>K computer, SPARC64 VIIIfx 2.0GHz, Tofu interconnect Fujitsu</td>
<td>705024</td>
<td>10510.0</td>
<td>11280.4</td>
<td>12660</td>
</tr>
<tr>
<td>5</td>
<td>DOE/SC/ Аргонська національна лабораторія, США</td>
<td>Mira - BlueGene/Q, Power BQC 16C 1.60GHz, Custom IBM</td>
<td>786432</td>
<td>8586.6</td>
<td>10066.3</td>
<td>3945</td>
</tr>
</tbody>
</table>
Відмітимо появу «новачка» на позиції №6 – систему Piz Daint швейцарського центру з високим показником реальної продуктивності 6271 ТФлопс.

В цій десятці шість систем не використовують прискорювачі (Sequoia, K-Computer, Mira, Jukebox, Vulcan, SuperMUC).

Лідер рейтингу Tianhe-2 (33,86 ПФлопс) та №7 Stampede використовують прискорювачі Intel Xeon Phi. В двох системах використані прискорювачі NVidia K20x: №2 Titan Cray XK7 та №6 Piz Daint Cray XC30.

Російський лідер «Ломоносов» (на позиції №37) з реальною продуктивністю 902 ТФлопс використовує два види прискорювачів: NVidia 2070 GPU та IBM PowerXCell 8i.

Японська система HA8000-tcH7210 (на позиції №36) з реальною продуктивністю 905,4 ТФлопс використовує два види прискорювачів: NVidia K20/K20x та Intel Xeon Phi 5110p.

Китайська система Inspur (№204) з реальною продуктивністю 196 ТФлопс використовує прискорювачі NVidia K20M та Intel Xeon Phi 5110p.

Активно розширюється список рейтингу Graph500 [3], який показує рівень придатності систем для розв’язання задач з високою інтенсивністю обробки даних. Чільну десятку цього рейтингу представляє табл.3.

<table>
<thead>
<tr>
<th>Позиція</th>
<th>Організація</th>
<th>Комп’ютер</th>
<th>Ядер</th>
<th>Макс. продуктивність, ТФлопс</th>
<th>Піковая продуктивність, ТФлопс</th>
<th>Потужність, кВт</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>Швейцарський національний суперкомп’ютерний центр (CSCS), Швейцарія</td>
<td>Piz Daint - Cray XC30, Xeon E5-2670 8C 2.600GHz, Aries interconnect, NVIDIA K20x Cray Inc.</td>
<td>115984</td>
<td>6271.0</td>
<td>7788.9</td>
<td>1753.66</td>
</tr>
<tr>
<td>7</td>
<td>Технічний обчислювальний центр/ Університет Техасу, США</td>
<td>Stampede - PowerEdge C8220, Xeon E5-2680 8C 2.700GHz, Infiniband FDR, Intel Xeon Phi SE10P Dell</td>
<td>462462</td>
<td>5168.1</td>
<td>8520.1</td>
<td>4510</td>
</tr>
<tr>
<td>8</td>
<td>Центр Юліх (FZJ), Німеччина</td>
<td>JUQUEEN - BlueGene/Q, Power BQC 16C 1.600GHz, Custom Interconnect IBM</td>
<td>458752</td>
<td>5008.9</td>
<td>5872.0</td>
<td>2301</td>
</tr>
<tr>
<td>9</td>
<td>DOE/NNSA/LLNL, США</td>
<td>Vulcan - BlueGene/Q, Power BQC 16C 1.600GHz, Custom Interconnect IBM</td>
<td>393216</td>
<td>4293.3</td>
<td>5033.2</td>
<td>1972</td>
</tr>
<tr>
<td>10</td>
<td>Лейбниц Центр, Німеччина</td>
<td>SuperMUC - iDataPlex DX360M4, Xeon E5-2680 8C 2.700GHz, Infiniband FDR IBM</td>
<td>147456</td>
<td>2897.0</td>
<td>3185.1</td>
<td>3423</td>
</tr>
</tbody>
</table>
Таблиця 3. Перші 10 позицій рейтингу Graph500 за листопад 2013 р.

<table>
<thead>
<tr>
<th>№</th>
<th>Комп'ютер</th>
<th>Організація</th>
<th>Кількість вузлів</th>
<th>Кількість ядер</th>
<th>GTEPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Sequoia (IBM - BlueGene/Q, Power BQC 16C 1.60 GHz)</td>
<td>Національна лабораторія Лоуренса Лівермора</td>
<td>65536</td>
<td>1048576</td>
<td>15363</td>
</tr>
<tr>
<td>2</td>
<td>Mira (IBM - BlueGene/Q, Power BQC 16C 1.60 GHz)</td>
<td>Аргонська національна лабораторія</td>
<td>49152</td>
<td>786432</td>
<td>14328</td>
</tr>
<tr>
<td>3</td>
<td>JUQUEEN (IBM - BlueGene/Q, Power BQC 16C 1.60 GHz)</td>
<td>Німецький Центр Юліх (FZJ)</td>
<td>16384</td>
<td>262144</td>
<td>5848</td>
</tr>
<tr>
<td>4</td>
<td>K computer</td>
<td>RIKEN Інститут обчислювальних наук (AICS)</td>
<td>65536</td>
<td>524288</td>
<td>5524.12</td>
</tr>
<tr>
<td>5</td>
<td>Fermi (IBM - BlueGene/Q, Power BQC 16C 1.60 GHz)</td>
<td>CINECA</td>
<td>8192</td>
<td>131072</td>
<td>2567</td>
</tr>
<tr>
<td>6</td>
<td>Tianhe-2 (MilkyWay-2)</td>
<td>Національний університет оборонних технологій Китаю</td>
<td>8192</td>
<td>196608</td>
<td>2061.48</td>
</tr>
<tr>
<td>7</td>
<td>Turing (IBM - BlueGene/Q, Power BQC 16C 1.60GHz)</td>
<td>CNRS/IDRIS-GENCI</td>
<td>4096</td>
<td>65536</td>
<td>1427</td>
</tr>
<tr>
<td>8</td>
<td>Blue Joule (IBM - BlueGene/Q, Power BQC 16C 1.60 GHz)</td>
<td>Рада з питань науки та технологій – Лабораторія Даресбюри</td>
<td>4096</td>
<td>65536</td>
<td>1427</td>
</tr>
<tr>
<td>9</td>
<td>DIRAC (IBM - BlueGene/Q, Power BQC 16C 1.60 GHz)</td>
<td>Університет Единбургу</td>
<td>4096</td>
<td>65536</td>
<td>1427</td>
</tr>
<tr>
<td>10</td>
<td>Zumbrota (IBM - BlueGene/Q, Power BQC 16C 1.60 GHz)</td>
<td>EDF R&amp;D</td>
<td>4096</td>
<td>65536</td>
<td>1427</td>
</tr>
</tbody>
</table>

Системи IBM BlueGene/Q з різною кількістю вузлів посідають три перші позиції у Graph500: Sequoia у Ліверморській лабораторії (65536 вузлів, 1048576 ядер, 15363 GTEps), Mira в Аргонській лабораторії (49152 вузла, 786432 ядра, 14328 GTEps), JUQUEEN в німецькому центрі Юліх (16384 вузла, 262144 ядра, 5848 GTEps).

Відмітимо, що Аргонська національна лабораторія для розв’язання графових задач, крім великої системи Mira (позиція №2), використовує спеціалізовані пристрої (міні-суперкомп’ютер) Celero на основі реконфігурованих прискорювачів HC-2 компанії Convey, який у рейтингу Graph500 посідає позицію №64, маючи показник 11,448 GTEps. Аналогічні прискорювачі посідають у рейтингу Graph500 позиції №63, 65, 66, 78, 82, обслуговуючи запити клінік, Інституту біоінформатики Вірджинії та інших користувачів.

Включення автономнých реконфігурованих прискорювачів в рядки рейтингу Graph500 свідчить про їх перспективність для розв’язання графових задач. Більш детально реконфігуровані прискорювачі розглянемо у наступному розділі, присвяченому актуальним прискорювачам.

В Інституті кібернетики ім. В.М. Глушкова НАН України продовжується модернізація кластера СКІТ-4, який посідає позиції №38 і 50 в рейтингу Green500 у вимірі енергозберігання. Встановлена продуктивність в 25 кВт.

Перспективним і доступним для значного підвищення продуктивності може бути присвоєння в якості нових прискорювачів IBM PowerXCell 8i. Ці прискорювачі мають доступну низьку ціну, забезпечені багатьма засобами ПЗ, більш детально про них нижче при огляді чотирьох різновидів прискорювачів.
2 Ефективні засоби прискорення

У вузлах сучасних гібридних суперкомп’ютерів основні багатоядерні процесори найчастіше взаємодіють з одним чи двома прискорювачами наступних видів:

- Intel Xeon Phi
- графічні сопроцесори компаній NVidia або AMD
- реконфігуровні сопроцесори на FPGA компаній Xilinx або Altera
- синергетичні процесори IBM PowerXCell 8i

В якості ілюстрації ефективності застосувань цих прискорювачів можуть служити такі сучасні гібридні системи: китайська Tianhe-2 (Intel Xeon Phi, 33,86 ПФлопс), американська Titan Cray XK7 (NVidia K 20, 17,59 ПФлопс), японська Atacama (FPGA, 120 ТФлопс), російський «Ломоносов» (IBM PowerXCell 8i, NVidia Tesla, 902 ТФлопс).

2.1 Прискорювачі Intel Xeon Phi

Прискорювачі Intel Xeon Phi у поєднанні у вузлах з процесорами Intel Xeon E5 кардинально підвищують швидкість розв’язання багатопрофільних задач. Intel підтримує таку взаємодію розширеними засобами ПЗ. У 2013 році були доступні три сімейства Intel Xeon Phi (табл. 4).

<table>
<thead>
<tr>
<th>№</th>
<th>Назва моделі, параметри</th>
<th>Адаптованість до задач</th>
<th>Максимальні енерговитрати</th>
<th>Ціна</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3120p, (6 ГБ, 1,1 GHz, 57 ядер)</td>
<td>Монте Карло, Black-Scholes, HP1 тощо</td>
<td>300 Вт</td>
<td>$1695</td>
</tr>
<tr>
<td>2</td>
<td>5100d, (8 ГБ, 1,053 GHz, 60 ядер)</td>
<td>З підвищеннями вимогами до об’єму, пропускної здатності пам’яті</td>
<td>245, 225 Вт</td>
<td>$2759, $2649</td>
</tr>
<tr>
<td>3</td>
<td>7100x, (16 ГБ, 1,238 GHz, 61 ядро)</td>
<td>Вимоги максимальної продуктивності, пам’яті</td>
<td>300 Вт</td>
<td>$4129</td>
</tr>
</tbody>
</table>

Старші моделі Xeon Phi 7120p та 7120x оснащені буферами карти пам’яті GDDR5 ємністю 16 ГБ з пропускною здатністю 352 ГБ/с. Вони забезпечують продуктивність 1,2 ТФлопс з подвійною точністю. Молодші моделі Xeon Phi 3120 оснащаються 6 ГБ GDDR5 з пропускною здатністю 240 ГБ/с, забезпечують продуктивність 1 ТФлопс з подвійною точністю.

На рис. 1 представлено блок-схему сопроцесора Intel Xeon Phi [5].
Анонсовано нове покоління Xeon Phi, яке випускатиметься у 2015 році по технології 14 нм з інтеграцією сопроцессора, оперативної пам’яті. Ця інтеграція дозволить значно підвищити пропускну здатність шини, яка поєднує сопроцессор з оперативною пам’ятю.

2.2 Графічні прискорювачі

Основним ринковим конкурентом прискорювачів Intel Xeon Phi стали прискорювачі NVidia K20, а незабаром будуть доступні й NVidia K40.

Багато суперкомп’ютерів нового списку рейтингу найбільших продуктивних систем Top500 використовують прискорювачі NVidia K20.

Серед них система Titan Cray XK7 (США), яка посідає другу східку з Linpack-продуктивністю 17590 ТФлопс та енергоспоживанням 8209 кВт.

Особливо широко прискорювачі NVidia K20 представлені у новому списку рейтингу Green500, в якому, зокрема 10 найбільш «зелених» систем використовують NVidia K20 (див. табл. 1).

Переваги NVidia K20 у підвищенні продуктивності та енергозбереженні дещо зменшують дві обставини:

- програмування графічних прискорювачів є одним з найбільш складних;
- по застосуванню нових норм техпроцесу NVidia трохи відстає, що відповідно збільшує ціну, габарити, вагу, енерговитрати.

Компанія NVidia не має своїх технологічних ліній, як Intel, IBM тощо, та найчастіше домовляється про випуск чипів з тайванською TSMC. Компанії NVidia та AMD тепер переводять випуск своїх чипів на норму 28 нм після норми 40 нм.

На рис. 2 представлено спрощену блок-схему NVidia Tesla K20x.

Рис. 2. Спрощена блок-схема NVidia Tesla K20x
Нешодавно анонсовано співробітництво компаній IBM та NVidia з випуску вузлів з об’єднанням нових процесорів IBM Power 8 та нових прискорювачів NVidia K40. Такі вузли планують застосувати у 2014-2015 рр. при створенні нового покоління багатопетафлопсних суперкомп’ютерів.

Новий 12-ядерний чип Power 8 у 2-3 рази швидший за чип Power 7 за рахунок переходу на техпроцес 22 нм, збільшення розміру кеша 12х95 МБ другого рівня та 96 МБ роздільного кеша третього рівня, тактової частоти до 4 ГГц, підвищення вдвічі швидкості схем вводу-виводу.

Новий NVidia K40 містить 2880 ядер CUDA, інтегровану оперативну пам’ять об’ємом 8-16 ГБ з пропускною здатністю до 500 ГГі/. Виводиться, що Tesla K40, досягає продуктивності 4,29 ТФлопс для обчислень з одинарною точністю та 1,43 ТФлопс для обчислень з подвійною точністю. Використано інтерфейс PCle третього покоління, який двічі продуктивніший за PCIe 2.

2.3 Реконфігуровальні прискорювачі на FPGA


При розв’язанні спеціалізованих задач плати на FPGA дозволяють підвищувати продуктивність на 1-2 порядки у порівнянні з використанням засобів загального призначення.

Так, у 2013 році в Чилі запрацювала суперкомп’ютерна система Atacama Compact Array Correlator, створена компанією Fujitsu та Національною астрономічною лабораторією Японії, яка включає 35 серверів Fujitsu Primergy та спеціалізований обчислювальний модуль на 4096 FPGA, об’єднаних 1024 оптоволоконними зв’язками. Ця система обробляє 512 млрд. відліків радіосигналів за секунду, одержаних з 16 радіоантен, що відповідає продуктивності 120 ТФлопс.

Ефективні блоки HC-1, HC-2 на FPGA для розв’язання складних задач обробки даних біоінформатики виготовляє компанія Convey Computer. Її міні-суперкомп’ютер HC-2 включає ряд Xilinx Virtex 6 FPGA, ряд процесорів Intel Xeon E5 із загальним енергопотреблінням 1670 Вт. Зокрема, в новому списку рейтингу Graph500 система Convey HC-1 посіла 82-ту позицію з показником 6 GTepr. Вона успішно розв’язує задачі в Інституті біоінформатики Вірджинії. Структуру HC-1 представлена на рис. 3.

В поточний список Graph500 включено ще 5 міні-суперкомп’ютерів Convey на позиціях 63-66 та 78. Система Celero на позиції №64 з показником 11,4 GTepr належить Аргонській національній лабораторії, в якій працює IBM BlueGene/Q з показником 14328 GTepr (позиція №2 нового списку Graph500).

В поточний список Graph500 включено ще 5 міні-суперкомп’ютерів Convey на позиціях 63-66 та 78. Система Celero на позиції №64 з показником 11,4 GTepr належить Аргонській національній лабораторії, в якій працює IBM BlueGene/Q з показником 14328 GTepr (позиція №2 нового списку Graph500).

Рис. 3. Структура HC-1
2.4 Синергетичний процесор IBM PowerXCell 8i

У червні 2008 року IBM вперше у світі подолала петафлопсний бар’єр продуктивності суперкомп’ютерів (система IBM RoadRunner, 1,026 ПФлопс), застосувавши об’єднання 6562 двоядерних чипів AMD Opteron з 12240 дев’ятиядерними чипами IBM PowerXCell 8i (згідно списку №31 рейтингу Top500).

У 2014 році IBM перейшла на петафлопсні системи нового покоління на базі нового процесора IBM Power 8, поєднаного з новими графічними прискорювачами NVidia K 40, відмовившись від застосування раніше анонсованого переходу на нові технічні характеристики процесора IBM PowerXCell 8i.

Навіть одна проведена IBM модернізація архітектури з переходом від норм 90 нм на норми 65 нм підвищила у п’ять разів продуктивність IBM PowerXCell 8i, знизила енерговитрати, значно прискорила обмін з системною пам’яттю.

Архітектура PowerXCell 8i представлена на рис. 4.

Рис. 4. 9-ти ядерний Cell

Дев’ятиядерний Cell-прискорювач є мікросхемою з 64-бітовим процесорним елементом PPE, восьми спеціалізованими сопроцесорами SPE на базі SIMD архітектури, зі спеціалізованою швидкісною шиною EIB, контролерами пам’яті та вводу-виводу. PPE розподіляє задачі на SPE, контролює системні операції вводу-виводу, при цьому SPE виконують математичні операції.

У PowerXCell 8i значно прискорено обмін з системною пам’яттю за рахунок введення в кожний SPE до 256 кб локальної пам’яті зі швидкісним доступом. З неї дані можуть переміщуватись в основну пам’ять та назад, навіть з випередженням, без зупинки обчислювального процесу в SPE. Всі SPE та PPE мають прискорений доступ до основної пам’яті через контролер сумісної пам’яті та внутрішню магістраль.

Основним інструментом програмування XCell є IBM SDK for Multicore Acceleration, у складі якого компілятори, відладчики, бібліотеки BLAS, FFT, засоби генерації випадкових чисел тощо. IBM XCell має чимало спільних рис з процесором IBM Power BQC, який застосовувався у складі суперкомп’ютера Sequoia.

IBM розробила блейд сервер QS22 на двох процесорах IBM PowerXCell 8i, який забезпечує, зокрема, високу ефективність розв’язання задач для нафтогазової індустрії, фінансової галузі тощо. IBM XCell має чимало спільних рис з процесором IBM Power BQC, який застосовувався у складі суперкомп’ютера Sequoia.

Продовжує використовувати чипи IBM PowerXCell 8i і компанія «Т-Платформи», розробник найшвидшого суперкомп’ютера Росії «Ломоносов». У список №42 Top500 «Ломоносов» посідає 37-ме місце, використовуючи 6 різновидів чипів, серед них 60 чипів IBM PowerXCell 8i.

Компанія «Т-Платформи» розробила лінійку продуктів PeakCell на базі чипів IBM PowerXCell 8i, які впроваджує в ряді країн. «Т-Платформи» створила новий дистрибутив «Т-Linux» для «революційної» синергетичної архітектури PowerXCell 8i [8].

Для робочих місць програмістів «Т-Платформи» розробила міні-кластер PeakCell GPS з піковою продуктивністю 8,19 TФлопс на чотирьох процесорах IBM PowerXCell 8i з підключенням оперативної пам’яті
до 64 ГБ, до чотирьох жорстких дисків. Також пропонується малогабаритний сервер, робоча станція на двох PowerXCell 8i [8].

3 Висновки


2. В новому списку «зеленого» рейтингу Green500 перша десятка систем використовує прискорювачи NVidia, найвищий показник енергозбереження має японська система Tsubame KFC (4,5 ГФлопс/Вт).

3. Зросла кількість систем, номінованих у список рейтингу Graph500, який оцінює рівень придатності систем для розв’язання задач з високою інтенсивністю обробки даних. Системи IBM BlueGene/Q з різною кількістю вузлів посідають три перших позиції, перша з них Sequoia забезпечує показник 15363 GТеpс. В новий список на позиціях 63-66, 78, 82 включені невеликі системи на базі реконфігуровних прискорювачів компанії Convey Computer. Наприклад, Аргонська національна лабораторія використовує для графових задач великий суперкомп’ютер Mira на 49152 вузла з показником 14328 GТеpс та автономний прискорювач Celero на одному вузлі з 8 ядрами з показником 11,448 GТеpс.

4. Модернізований кластер СКІТ-4 Інституту кібернетики ім. В.М. Глушкова НАН України включає 28 вузлів, кожен з процесором Intel Xeon E5-2670 та 36 прискорювачів NVidia Tesla M2075, забезпечує реальну продуктивність 18 ТФлопс при рівні енергоспоживання 25 кВт. Ці показники достатні для включення у рейтинг Top50 країн СНД та у першу половину списку «зеленого» рейтингу Green500 з рівнем 0,72 ГФлопс/Вт. Поки оцінки СКІТ-4 на придатність для розв’язання графових задач не зроблені.

5. СКІТ-4 модернізується головним чином для розв’язування багатопрофільних задач на базі Intel E5 та NVidia Tesla. Значення резерву модернізації для розв’язування ряду пріоритетних задач залишається введення вузлів на IBM PowerXCell 8i, використовуючи гібридні системи на базі Intel Xeon Phi, що забезпечує реальну продуктивність 18 ТФлопс при рівні енергоспоживання 25 кВт. Ці показники достатні для включення у рейтинг Top50 країн СНД та у першу половину списку «зеленого» рейтингу Green500 з рівнем 0,72 ГФлопс/Вт. Поки оцінки СКІТ-4 на придатність для розв’язання графових задач не зроблені.

Література

The Adaptive Routing Algorithm Taking Into Account the Trust Level to the Remote Nodes

G.M. Loutsky, V.Ye. Mukhin

National Technical University of Ukraine "Kiev Polytechnic Institute"

lutsky@comsys.kpi.ua, v_mukhin@mail.ru

Abstract. An algorithm of the adaptive routing, which implies the data transfer on a communication channel, formed by the criterion of the maximum security of data transfers for each constituent sub-channels between nodes. The safety criterion is defined as the probability of threats realization to data safety on a particular sub-channel.

Keywords

Adaptive routing, safety criterion, distributed computer systems

1 Introduction

The distributed computer system (DCS) consists of nodes including: servers, nodes, switches and communication channels, and the switches have a very important role there. In the DCS and switched networks the most of nodes-sources and nodes-receivers are not connected directly, and when the data are transferred between them, they must pass through routers and switches. The data flow may be forwarded (switched) from the source node to any receiver node. The principle of the switching, i.e. the routing algorithm and data packets buffering strategy determines which channels from the network topology can be used for maximum efficiency of the data transfers in the network. To form the directed secured communication channel we need to apply the special routing mechanisms that allow the direct data flow through the predefined routers, with respect that the communication channels between them are less prone to potential threats from the intrusions agents.

Routing mechanisms in the switched computer networks support the communications between the subjects or nodes in the network. In particular, these mechanisms determine the principle of the allocation, filling and releasing of the buffers and ports in the routers and also the time for data transfers to the next router along the path of the destination. In addition, the routing mechanisms determine the time intervals at which the resources are allocated in the routing process: the buffers and switch ports of the router and the parameters of the routing protocols for different network topologies [1]. Thus, these mechanisms have a key role for the conflict and deadlock situations resolution in the implementation of the routing protocols. It should be noted that significant influence on the time delay of the data transferred and on the network bandwidth has the mechanisms for the data processing and the characteristics of the network traffic. [2]

2 Adaptive routing algorithm on the safe route based on the criterion of trust level to the nodes of the DCS

The implementation of the adaptive routing algorithm by the safe route requires the additional parameters: weight of the data channel (or communication channel) in terms of its safety as a value that is proportional to the trust level for the corresponding DCS node, and is inversely proportional to the probability of the security threats realization on this channel and node. Each node (router) has its own the vector of weights of the communications channels, which is periodically updated by data from the other nodes, then also the routing tables are updated. Thus, each node supports three vectors.

So, each node $n$ support $SW_n$ – the vector of the safety of the communication channels:

$$SW_n = (SW(n,1),...,SW(n,M))$$

(1)

where $M$ – the number of networks to which the node $n$ is connected directly, $SW(n, i)$ – the weight of the channel from the viewpoint of its safety.

Further two vectors are forming: $NL_n$ – the distances vector for a node $n$:  

- 67 -
\[ NL_n = (NL(n,1),..., NL(n,N)) \]  

where \( NL(n, j) \) – the current estimate of the weight of the safety of the channel from node \( n \) to the network \( j \); \( N \) – the number of networks in the configuration; \( NT_n \) – the vector of the next hops for the node \( n \):

\[ NT_n = (NT(n,1),..., NT(n,N)) \]  

where \( NT(n, j) \) – the next router in the current route with a maximum weight of a secure communication channel from node \( n \) to the network \( j \).

Periodically, after a short time period, each node receives the weights of the safety of the channels from all neighboring nodes. Basing on the all received vectors of the weights of the safety of the channels the node \( x \) updates all these vectors as follows:

\[ NL(n,j) = \min (NL (n, j)), \text{при SW}(n, j) \geq TL_{crit} \]  

where \( NL(n, j) = y \), \( y \) – the router, wherein the \( NL (n, j) \) has the minimum value, \( TL_{crit} \) – the minimum value of the trust level to the node, which allows that the data can be transmitted to this node.

During the routing process the addresses of the routers on the packets route are added in the packet header field. The principle of the generation of the routers address is crucial, for example, for the directed routing it is necessary to ensure the packets transmission by the chain from the trusted routers only. Thus, the format of the packet, based on criteria of the nodes reliability \( (NR) \) and trust level to the nodes \( (TL) \) is the next:

<table>
<thead>
<tr>
<th>comments</th>
<th>version</th>
<th>IP-1</th>
<th>NL-1</th>
<th>NR-1</th>
<th>TL-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP-2</td>
<td>NL-2</td>
<td>NR-2</td>
<td>TL-2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IP-n</td>
<td>NL-n</td>
<td>NR-n</td>
<td>TL-n</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Fig.1. Packet format according to the criteria of the nodes reliability \( (NR) \) and the trust level to nodes \( (TL) \)

3 The realization of the adaptive routing by the safe route

The routing algorithms are differ, in particular, by the point of the decision-making about the route: if for this goal is used the special central controller, such decisions defines centralized routing; next there is an variant of the source routing, when the decisions is making before packets are injected to the network, and the next variant: routing is performed in a distributed way, when the decisions are made during the packets transfers through the network – the distributed routing \([3,4]\). Also, the combined solutions are possible. The one of such solution is based on a multiphase routing, when the source node defines a set of the destination nodes, and the route between them is determined in a distributed way. Let us consider in detail the features of the multiphase routing algorithm.

The multiphase routing algorithm is a combined mechanism. In this algorithm, the source node determines the node receiver or the some subset of the nodes-receivers, and the route between them is determined in a distributed way and consistently. The data transmission from the node is realized by the several neighboring nodes simultaneously, but only those nodes are used, that meet the criteria of possibility to transfer the data by them, for example, the trust level to these nodes. In fact, among all the neighboring nodes we need to choose the most trusted and reliable, and the data are transferred to these nodes. Further, the process of route constructing continues until the data will reach the destination node. This approach also improves the reliability of the data transmission due to their actual duplication and simultaneous transmission by the multiple channels.

The problem of implementation of this routing algorithm is a need for continuous and rapid exchange of metrics \((NL, NR, TL)\) between all the DCS nodes, as well as the redundant data channels, which may reduce the performance of the DSC and the network bandwidth of the user data transmission.

Fig. 8 shows the variant of the multiphase routing realization for computer system with a single-channel links. To improve the efficiency of the routing mechanism there is used the approach based on the implementation of multi-channel links (Fig. 9).

In this case, the redundant transmission channels are realized simultaneously and without mutual interlocking, for example, in Time Division Multiple Access (TDMA) mode, and the transfers of the vector of the weights of the
communication channels \((NL, NR, TL)\) is also simultaneously transmitted with these data that allow increase the data throughput.

In fact, in this case we have the combination of physical and topological multiple channels, which allows to change fast the data transmission route according to the current metrics of the nodes and communication channels and to provide sufficiently reliable data transmission from the node source to the receiver.

\[
NL_i, NR_i \quad NL_j, NR_j \quad NL_k, NR_k
\]

**Fig. 2.** The multiphase routing for the DCS with single-channel links

\[
NL_i, NR_i \quad NL_j, NR_j \quad NL_k, NR_k
\]

**Fig. 3.** The multiphase routing for the DCS with multi-channel links
4 Conclusion

The multiphase routing algorithm is the one of the most efficient for the directed data routing in the scalable computer systems. Additional efficiency is achieved through the use of computer systems with multi-channel communications, as in this case there is a combination of technologies and physical topology of multi-channel data that allows quickly re-route data according to the current parameters of the nodes and communication channels, and provide sufficiently reliable data transfer from the sources to the receivers.

References


Компонентная модель системы управления обучением с открытой формой представления тестового задания на основе грид технологий
Маслянко П.П., Руденко П.А.
Национальный технический университет Украины «Киевский политехнический институт», Киев, Украина
mppdom@i.ua, fr@bekker.ua

Аннотация. В данной работе приводится исследование использования грид-технологий для построения распределенных систем обучения с удаленным доступом. Построение на базе выбранной технологии компонентной модели системы управления обучением. Описываются подходы к созданию согласованной, открытой и стандартизованной технологической среды, обеспечивающей гибкое, безопасное, скоординированное совместное использование вычислительных ресурсов глобальной сети для решения сложных и ресурсоемких задач в важнейших областях современной науки и техники, таких как E-Learning. Приводится описание компонентной модели позволяющей интегрировать в систему управления обучением удаленного пользователя и преподавателя, отдельные локальные сети, удаленные филиалы и сетевые ресурсы учебного заведения.

Ключевые слова
Электронное обучение, E-Learning, грид-инфраструктура, грид-технологии.

1 Введение
В настоящее время происходит стремительное развитие технологий грид с целью создания согласованной, открытой и стандартизованной технологической среды, обеспечивающей гибкое, безопасное, скоординированное совместное использование вычислительных ресурсов глобальной сети для решения сложных и ресурсоемких задач в важнейших областях современной науки и техники.
Существует огромное количество грид-инфраструктур, которые отличаются по своему функциональному назначению. В частности большой класс грид-технологий составляют вычислительные гриды, ориентированные на распределенные вычисления с целью образования «виртуального суперкомпьютера» из большого числа компьютеров, связанных друг с другом посредством сети и работающих совместно при решении сложных задач, требующих значительных вычислительных и информационных ресурсов. Так, например, в E-learning все более широкое применение находят информационные гриды, обеспечивающие доступ к неоднородным, распределенным репозиториям данных большого объема наряду с разделяемым доступом к другим видам ресурсов, включая вычислительные.
Удаленный доступ и сетевые сервисы дают возможность использовать вычислительные гриды для построения распределенных обучающих систем различного назначения и предоставлять широчайший спектр образовательных услуг.

2 Постановка задачи
Одной из главных особенностей дистанционной формы обучения является то, что одновременно с обучающей системой могут работать тысячи, десятки и даже сотни тысяч пользователей. Так, например, известный в России интернет-университет информационных технологий (ИНТУИТ) почти на пять лет опередил знаменитый Массачусетский технологический институт, который только недавно предложил в открытом доступе электронные версии всех своих учебных курсов. По данным ИНТУИТ, сегодня в интернет-университете учится около 600 тысяч человек, ежедневно на сайте регистрируется приблизительно 300-400 новых пользователей. Для того чтобы одновременно обслужить такое большое количество учащихся, остро стал вопрос создания серверных систем на базе множества территориально распределенных серверов. В свою очередь, создание информационной системы для управления обучением на базе такой сети обусловлено несколькими причинами - обеспечение устойчивой работы, повышение производительности, возможность получения хорошей отказоустойчивой системы на случай возникновения сбоев в оборудовании, а также, сравнительно экономически дешевое решение.
Таким образом, цель этой статьи является исследование возможности использования грид-технологий для построения распределенных систем обучения с удаленным доступом и построение компонентной модели системы управления обучением.

3 Актуальность

В настоящее время происходит стремительное развитие технологий грид с целью создания согласованной, открытой и стандартизованной технологической среды, обеспечивающей гибкое, безопасное, скоординированное совместное использование вычислительных ресурсов глобальной сети для решения сложных и ресурсоемких задач в важнейших областях современной науки и техники, такой как, например: E-Learning.

Существующие грид-инфраструктуры делятся на следующие категории [1-10]:

- Computational Grid - грид для распределенных вычислений
- Data Grid - грид для обработки больших потоков данных
- Informational Grid - грид для интеграции крупных распределенных хранилищ (OGSA-DAI), в подобных архитектурах используется централизованный реестр, хранящий метаданные всех сервисов и распределенных хранилищ
- Hybrid Grid - грид сочетающий в себе как Computational/Data Grid так и Informational Grid
- Semantic Grid - это любой, из описанных типов грид-архитектур в котором описывается семантика ресурсов (интерфейсы, характеристики производительности, особенности безопасности). Позволяет оперировать данными расположенными в различных базах данных.

Как уже было отмечено, одной из главных особенностей дистанционной формы обучения является то, что одновременно с системой могут работать десятки тысяч пользователей. В результате чего особенно остро стает вопрос об использовании высокопроизводительных вычислений и массовой обработки информации. Поэтому внедрение грид-технологий при построении современных систем дистанционного обучения является актуальной научной и практической задачей.

4 Выбор грид-технологии при построении современной обучающей системы

Важным вопросом создания информационной системы управления обучением в области грид и распределенного компьютера является выбор грид-технологий. Одним из основных требований к оснащению гридами учебной базы - это обеспечение возможности для учащихся работать с гридами разных классов, в частности, как с вычислительными, так и информационными.

На сегодняшний день успешно развиваются и используются два основных подхода к построению вычислительных гридов [1-10].

ПЕРВЫЙ ПОДХОД, так называемый сервисный грид (Service Grid), предполагает развертывание распределенной сервис-ориентированной инфраструктуры, обеспечивающей унифицированный удаленный доступ к выделенным ресурсам уровня кластеров или суперкомпьютеров. Поставщиками ресурсов в таких системах являются достаточно крупные организации, обладающие ресурсами указанного уровня. Как правило, включаемые в сервисный грид ресурсы являются гомогенными, то есть работающие под управлением одной версии операционной системы и предоставляют одинаковое окружение для запускаемых задач. Количество пользователей таких гридов гораздо больше, чем число поставщиков ресурсов. При этом каждый пользователь может использовать ресурсы грид для запуска своих приложений. Примерами сервисных гридов являются EGEE, NorduGrid, TeraGrid. Базовым промежуточным ПО подобных систем служат технологии Globus Toolkit, gLite, ARC, UNICORE. Недостаток сервисных гридов является высокая сложность установки и администрирования указанного ПО, что ограничивает круг потенциальных поставщиков ресурсов.

ВТОРОЙ ПОДХОД, так называемый грид рабочих станций (Desktop Grid), предполагает использование ресурсов большого количества простаивающих персональных компьютеров, подключенных к сети. Поставщиками ресурсов в подобных системах являются обычные рядовые пользователи. При таком подходе подключаемые в грид ресурсы рабочих станций являются гетерогенными по своей архитектуре и программному обеспечению. Данные ресурсы, при этом, как правило, доступны не постоянно, а только в моменты их простоя. Это означает, что в отличие от сервисных гридов, в состав ресурсов грид рабочих станций является гораздо более динамичным. В подобных системах число поставщиков ресурсов обычно гораздо больше числа пользователей, использующих ресурсы грид для запуска приложений. Примерами технологов для организации грид рабочих станций являются BOINC, Condor, XtremWeb. В отличие от использования технологий сервисных гридов, данные технологии позволяют быстро и легко подключать к системе новые ресурсы [9].
В данной работе для создания информационной системы управления обучением был выбран подход на основе Desktop Grid. Для этих целей использовались следующие программные средства:
- программное обеспечение gLite, как наиболее распространенный способ организации грид-вычислений в Европе;
- грид рабочих станций (desktop grids) BOINC (Berkley Open Infrastructure for Network Computing), в качестве вычислительных ресурсов которого планируется использовать мощности компьютерных классов и выделенные ресурсы организации EGEE, подключенные к гриду рабочих станций при помощи технологии EDGeS [8-10].

На базе выбранных технологий на кафедре прикладной математики НТУУ «КПИ» была создана распределенная информационная система управлением обучением, общая структура которого представлена на рис.1

Рис. 1. Структурная схема системы управления обучением для тестов с открытой формой тестового задания

Компонентная модель системы управления обучением для тестов с открытой формой тестового задания представлена на рис.2.
Рис. 2. Компонентная модель системы управления обучением для тестов с открытой формой тестового задания. Диаграмма Компонентов в нотации UML.

Системная спецификация компонентной модели системы управления обучением показана в табл. 1-6.
Таблица 1. Сущность ListEtalonTree для хранения эталонного ответа

<table>
<thead>
<tr>
<th>Id (PK)</th>
<th>Уникальный идентификатор записи</th>
</tr>
</thead>
<tbody>
<tr>
<td>id_konceptul (FK)</td>
<td>Ссылка на идентификатор концептулы</td>
</tr>
<tr>
<td>id_parent</td>
<td>Ссылка на идентификатор родительской концептулы</td>
</tr>
<tr>
<td>id_link_search (FK)</td>
<td>Ссылка на идентификатор связи для поиска</td>
</tr>
<tr>
<td>id_answer_expert (FK)</td>
<td>Ссылка на идентификатор ответа преподавателя</td>
</tr>
<tr>
<td>flag_main_konceptul</td>
<td>Флаг главной концептулы</td>
</tr>
<tr>
<td>flag_key_konceptul</td>
<td>Флаг ключевой концептулы</td>
</tr>
<tr>
<td>id_mandatory (FK)</td>
<td>Ссылка на идентификатор важности концептулы</td>
</tr>
<tr>
<td>id_link_mark (FK)</td>
<td>Ссылка на идентификатор типа связи для нахождения оценки</td>
</tr>
</tbody>
</table>

Таблица 2. Сущность ListKonceptul для хранения имен концептул

| Id konceptul (FK) | Идентификатор концептулы |
| text_konceptul | Название концептулы |

Таблица 3. Сущность ListValueKonceptul возможных вариантов значений концептул

<table>
<thead>
<tr>
<th>Id (PK)</th>
<th>Уникальный идентификатор записи</th>
</tr>
</thead>
<tbody>
<tr>
<td>text_value_konceptul</td>
<td>Текстовое представление возможного варианта для концептулы эталонного ответа</td>
</tr>
<tr>
<td>id_konceptul (FK)</td>
<td>Ссылка на идентификатор концептулы, для которой определяются возможные варианты значений</td>
</tr>
<tr>
<td>id_sootv (FK)</td>
<td>Ссылка на идентификатор степени соответствия для концептулы</td>
</tr>
</tbody>
</table>

Таблица 4. Сущность SootvLingvoVariable степеней соответствий для концептул, определенных через лингвистическую переменную

<table>
<thead>
<tr>
<th>Id (PK)</th>
<th>Уникальный идентификатор записи</th>
</tr>
</thead>
<tbody>
<tr>
<td>name_lingvo_sootv</td>
<td>Текстовое представление термов лингвистической переменной «степень соответствия»</td>
</tr>
<tr>
<td>koef_sootv</td>
<td>Численное значение степени соответствия</td>
</tr>
</tbody>
</table>

Таблица 5. Сущность MandatoryLingvoVariable степеней соответствий для концептул, определенных через лингвистическую переменную

<table>
<thead>
<tr>
<th>Id (PK)</th>
<th>Уникальный идентификатор записи</th>
</tr>
</thead>
<tbody>
<tr>
<td>name_lingvo_mandatory</td>
<td>Текстовое представление термов лингвистической переменной «важность концептулы»</td>
</tr>
<tr>
<td>koef_mandatory</td>
<td>Численное значение важности</td>
</tr>
</tbody>
</table>
Таблица 6. Сущность ListUserTree степеней соответствий для концептуал, определенных через лингвистическую переменную

<table>
<thead>
<tr>
<th>Столбец</th>
<th>Описание</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id (PK)</td>
<td>Уникальный идентификатор записи</td>
</tr>
<tr>
<td>id_konceptual (FK)</td>
<td>Ссылка на идентификатор концептуалы которую определил пользователь</td>
</tr>
<tr>
<td>id_parent</td>
<td>Ссылка на идентификатор родительской концептуалы</td>
</tr>
<tr>
<td>id_answer_user (FK)</td>
<td>Ссылка на идентификатор ответа пользователя</td>
</tr>
<tr>
<td>id_mandatory_user (FK)</td>
<td>Ссылка на идентификатор важности концептуалы</td>
</tr>
<tr>
<td>id_undef (FK)</td>
<td>Ссылка на идентификатор недопустимой концептуалы</td>
</tr>
<tr>
<td>id_user (FK)</td>
<td>Ссылка на идентификатор пользователя</td>
</tr>
<tr>
<td>id_answer_expert (FK)</td>
<td>Ссылка на идентификатор ответа преподавателя</td>
</tr>
</tbody>
</table>

5 Заключение

Создание распределенной многосерверной информационной системы обучения на основе грид технологий с равномерной загрузкой вычислительных ресурсов открывает большие возможности для предоставления образовательных услуг. Применение компонентной модели системы управления обучением для тестов с открытой формой тестового задания в университетах Украины, позволяет проектировать высокопроизводительные системы дистанционной формы обучения.

Компонентная модель позволяет интегрировать в систему управления обучением удаленного пользователя и преподавателя, отдельные локальные сети, удаленные филиалы и сетевые ресурсы учебного заведения.

Список литературы

Динамические и статические методы планирования пакетов заданий в распределенных вычислительных системах

С.В. Минухин

Харьковский национальный экономический университет, 9-а, пр. Ленина, Харьков, Украина

ms_vl@mail.ru

Аннотация. Рассмотрены вопросы планирования заданий в распределенных вычислительных системах на основе метода о покрытии. Разработан сценарий проведения экспериментальных исследований и сравнительного анализа различных методов решения задачи о покрытии и метода FCFS. Проведены исследования оценки эффективности исследуемых методов на основе времени выполнения и коэффициента использования ресурсов. Приведен сравнительный анализ полученных результатов экспериментов для динамического и статического методов пакетного планирования. Сформулированы требования к выбору параметров компонента используемого механизма планирования, позволяющие повысить производительность распределенных вычислительных систем и сформулировать предпочтения использования динамического и статического методов планирования.

Ключевые слова

Распределенная вычислительная система, задание, планирование, метод покрытия, время выполнения, коэффициент использования, динамический метод, статический метод.

1 Введение

Одной из важнейших задач повышения производительности и эффективности работы распределенных вычислительных систем (мультикластерных, вычислительных Грид-систем) является выбор метода планирования, режима работы с заданиями входной очереди (пакетный, интерактивный), а также оценка эффективности на основе определённых критериев (целевых функций). В качестве таких критериев используются такие, как время разрешения очереди (время завершения выполнения последнего задания очереди, makespan), время ответа (или среднее время ответа – среднее время выполнения одного задания с учетом времени ожидания до обработки, времени, затрачиваемого на его планирование (совокупное время, называемое ожиданием), и времени решения задания на ресурсе), а также коэффициент использования ресурсов (коэффициент загрузки). При этом основной задачей является определение таких параметров планирования и параметров инфраструктуры распределенной системы, при которых при обработке потока входных заданий, оптимизируются приведенные целевые функции, а также обеспечивается требуемый (допустимый) уровень качества обслуживания пользователей QoS (Quality of Service).

Целевые функции отражают различные стратегии для оценки эффективности работы распределенной системы: коэффициент использования является системным (системно ориентированным) показателем, характеризующим загрузку ресурсов и основным для владельцев ресурсов; временные критерии отражают стратегии, ориентированные на пользователей (как одиночных, так и в составе виртуальных организаций). Таким образом, состав пользователей и решаемые ими задачи (заявки, запросы) определяют и выбор режима работы с заданиями входной очереди, характеризующие приоритеты одиночных пользователей или участников виртуальных организаций. Для первой группы более предпочтительным является, как правило, интерактивный режим, для второй группы – пакетные режимы обработки задач, получившие в настоящее время значительное развитие [1–6]. В данном исследовании развивается метод, в котором используется математическая модель задачи о наименьшем покрытии, основные результаты использования которого получены в работах [1–3, 5, 6].
Целью настоящего исследования является исследование и анализ основанных на методе покрытия методов планирования пакетов заданий с использованием статического и динамического режимов, определяемых изменением параметров системы планирования и инфраструктуры распределенных вычислений в зависимости от изменения параметров потоков входных заданий и распределенной вычислительной системы, а также оценка эффективности их использования на основе выбранных критериев функционирования распределённой системы.

2 Организация проведения исследования методов планирования пакетов заданий

В исследовании проведен анализ эффективности методов планирования на основе плана экспериментов, который представлен на рис. 1. Для проведения анализа были выбраны 3 метода, относящиеся к методам решения задачи о покрытии – точный метод решения задачи о покрытии, жадный алгоритм решения задачи о покрытии, эвристический метод MC (minimal cover) [1], а также, с целью проведения сравнительного анализа, метод FCFS.

Предварительные исследования [2, 4, 5] показали, что для всех этих методов наиболее важным, с точки зрения повышения эффективности их использования, является величина средней относительной сложности заданий (среднего времени выполнения заданий входной очереди на ресурс), рассчитываемая как отношение средней сложности заданий (среднего времени выполнения) к средней производительности ресурсов системы. Для оценки влияния фактора сложности задания на критерии эффективности функционирования системы в данной работе используется следующая классификация: в качестве малой относительной сложности заданий было выбрано значение 10, в качестве большей – 1000 (как правило, дифференциация заданий по сложности (времени выполнения) определяется для такой классификации несколькими порядками). Таким образом, характеристика задания – «относительная сложность задания» – выбрана в качестве корневого узла предлагаемого дерева последовательности проведения экспериментов (см. рис. 1), а основные узлы определяют направления исследования, соответствующие заданиям с малой (левая ветвь) и большой (правая ветвь) относительной сложностью (рис. 1).


Рис. 1. Дерево сценария исследования и оценки эффективности методов планирования для заданий с различными значениями параметров их характеристик
Среднее время освобождения ресурсов системы [4, 5] является ключевым показателем для выбора величины периодичности планирования, так как он определяет среднюю продолжительность работы ресурса для решения задачи соответствующей сложности, и, следовательно, позволяет определить периодичность планирования (интервал планирования) в качестве некоторой границы, при превышении которой ресурсы, с большой вероятностью, будут проставать, т.е. не будут загружены заданиями вследствие того, что планирование не проводилось. Распределение относительной сложности заданий определяется отношением используемых в моделировании двух нормально распределенных величин, и в случаях, когда среднее значение времени выполнения заданий превышает среднюю величину производительности ресурсов системы, оно также имеет нормальное распределение.

В исследовании использованы следующие значения периодичности: для задач малой относительной сложности периодичность планирования равна 1 такту [2] (минимальной периодичности в использованной имитационной модели механизма планирования); для задач большой относительной сложности равно (-3σ) относительно среднего времени освобождения ресурса. В данном случае использование правила 3σ гарантирует, что периодичность планирования будет определяться минимально возможным временем, требуемым для освобождения ресурса. Так, например, если выбрать величину периодичности равной среднему времени освобождения ресурса +2σ, это приведет к тому, что задания, длительность которых равна среднему времени ±2σ будут проставать, так как периодичность планирования будет больше времени, требуемого на выполнение этого задания. Таким образом, выбор экстремальных значений для определения периодичности планирования, распределенной по случайному закону, позволяет определить условия достижимости максимальной эффективности планирования для заданных параметров выполняемых заданий и имеющихся ресурсов.

При проведении экспериментов использовано небольшое количество ресурсов (в качестве их могут выступать процессоры или ядра процессора) – в диапазоне от 10 до 30. Это связано с тем, что точный метод решения задачи о покрытии не позволяет проводить моделирование с требуемым количеством тестов (например, 40 наблюдений, позволяющих получать статистически достоверные результаты) и количества ресурсов более 30 вследствие экспоненциальной сложности точного метода решения задачи о покрытии. Выбираемая универсальность задачи (отношение количества ресурсов, на которых может быть решено задание вхождение очереди, к общему количеству ресурсов распределенной системы) определяет граничные ситуации для планирования заданий на ресурсы: 10% - задания уникальные, 50% - задания могут быть решены в 50% от имеющихся ресурсов, 90% - задания могут быть решены на 90% от имеющихся ресурсов. Как показали исследования [4, 5], универсальность заданий также является важным фактором, влияющим на эффективность работы методов планирования.

Далее рассмотрен динамический метод пакетного планирования, при котором исследуется влияние интенсивности задач вхождения потока и количества ресурсов системы на величину пул и периодичность планирования для повышения эффективности функционирования распределенной вычислительной системы.

3 Исследование и анализ динамических методов пакетного планирования заданий

В соответствии со сценарием проведения исследований рассмотрим влияние заданий малой сложности на результаты работы алгоритмов.

В качестве настроек динамической системы использованы следующие: средняя сложность заданий – 100, СКО – 10 (нормальный закон распределения); средняя интенсивность заданий входного потока – 50, СКО – 5 (нормальный закон распределения); универсальность заданий – 10, 50 и 90%; размер пакета заданий – 1%; величина пул – 10, 50; производительность ресурса – 10, СКО – 1 (нормальный закон распределения). Количество ресурсов изменялось в диапазоне от 10 до 30 с шагом 10; периодичность планирования – 1 такт [2, 4, 5].

На рис. 2 и 3 показаны результаты планирования заданий с заданными параметрами для точного метода, жадного алгоритма, метода FCFS для пула 10 и универсальности заданий 10%. Следует отметить, что результаты планирования для точного метода, жадного алгоритма и метода FCFS практически идентичны, как для времени выполнения, так и для коэффициента использования ресурсов. При увеличении количества ресурсов имеет место нелинейное уменьшение среднего времени выполнения задания (рис. 2): при увеличении количества ресурсов до 20 (прирост – 10 ресурсов) уменьшение составляет 110 тактов, а при увеличении количества ресурсов до 30 (прирост – 10 ресурсов) уменьшение составило только 30 тактов. Такой эффект проявляется вследствие уменьшения очередей заданий на соответствующие ресурсы. Увеличение количества ресурсов повышает вероятность нахождения подходящего ресурса для задач низкой универсальности, и, таким образом, меньше времени затрачивается на нахождение задания в очереди на ресурс. Увеличение количества ресурсов в данном случае приводит к увеличению их общей загрузки и положительно влияет на балансировку загрузки ресурсов (рис. 3). Использование метода FCFS приводит к значительно худшим результатам, чем точный метод, жадный алгоритм и эвристический метод MC, в первую очередь, потому, что
FCFS загружает первый доступный ресурс в списке свободных и подходящих (First Fit), а в условиях низкой универсальности заданий эта процедура становится неэффективной в случае нескольких конкурирующих между собой заданий, претендующих на один и тот же ресурс.

На рис. 4 и 5 показаны результаты планирования для точного метода, жадного алгоритма, метода МС и FCFS для величины пула 10 и универсальности заданий равной 50 %. Особенностью этих результатов является практически линейный тренд уменьшения среднего времени выполнения для точного метода, жадного алгоритма и МС при увеличении количества ресурсов. Среднее время выполнения для метода FCFS, точного и эвристических методов (жадного алгоритма и МС) для количества ресурсов равного 10 становится минимальным, а коэффициент использования для количества ресурсов 10 одинаков для всех методов. Увеличение количества ресурсов практически не улучшает результаты планирования при использовании метода FCFS, более того, при увеличении количества ресурсов до 20 наблюдается резкое уменьшение коэффициента использования (рис. 5).

Рис. 2. Зависимость среднего времени выполнения заданий для точного метода, жадного метода, МС и FCFS для величины пула 10, значения универсальности, равной 10 % от количества ресурсов

Рис. 3. Зависимость среднего значения коэффициента использования ресурсов для точного метода, жадного, МС и FCFS для величины пула 10, универсальности, равной 10 % от количества ресурсов
Анализ результатов планирования для точного метода, жадного алгоритма, МС и FCFS при величине пула 10 и универсальности заданий равной 90% показан на рис. 6, 7. При количестве ресурсов равным 10 среднее время выполнения и коэффициент использования ресурсов у всех методов одинаковы, а при увеличении количества ресурсов до 20 эффективность метода FCFS снижается: уменьшается среднее значение коэффициента использования (рис. 6). Это свидетельствует о неэффективности планирования методом FCFS в условиях тенденции увеличения (или большого количества) ресурсов для заданий с высокой универсальностью. В случае практической реализации это соответствует ситуации, при которой вычислительная архитектура имеет низкую гетерогенность (фактически является гомогенной).
Для универсальности 90%

Рис. 7. Зависимость среднего значения коэффициента использования ресурсов для точного метода, жадного, MC и FCFS для величины пула 10 и универсальности заданий, равной 90 % от количества ресурсов

Таким образом, при небольшом количестве ресурсов и с увеличением универсальности поступающих на обработку в систему заданий все методы, основанные на использовании метода покрытия, позволяют получить существенно лучшие результаты по сравнению с наиболее используемым в настоящее время в промышленных системах методом FCFS.

Сравнительный анализ особенностей влияния величины пула на эффективность использования методов планирования для универсальности заданий 50 % приведен на рис. 8–13. На рис. 8, 9 показаны результаты планирования для точного метода, которые подтверждают влияние величины пула на эффективность планирования. В данном случае небольшой пул обеспечивает лучшие результаты планирования – времени выполнения и коэффициента использования ресурсов.

Для универсальности 50%

Рис. 8. Зависимость среднего времени выполнения заданий для точного метода для величины пула 10 и 50 при универсальности заданий, равной 50 % от количества ресурсов

Для универсальности 50%

Рис. 9. Зависимость среднего значения коэффициента использования ресурсов для точного метода для величины пула 10, 50 при универсальности заданий, равной 50 % от количества ресурсов
Таким образом, величина пула, даже при небольшом количестве ресурсов, влияет на производительность распределенной вычислительной системы, и оптимизация ее работы может быть осуществлена на основе выбора его величины.

В работе приведены результаты, обосновывающие влияние периодичности планирования на эффективность функционирования распределенной вычислительной системы.

Для универсальности 50%

Рис. 10. Зависимость среднего времени выполнения для метода МС для величины пула 10, 50 при универсальности заданий, равной 50 % от количества ресурсов

Рис. 11. Зависимость среднего значения коэффициента использования ресурсов для метода МС для величины пула 10, 50 при универсальности заданий, равной 50 % от количества ресурсов

4 Исследование и анализ статических методов пакетного планирования заданий

Использование статического метода планирования предполагает отсутствие динамически настраиваемых параметров модели, как было рассмотрено в п. 3 данного исследования, в случае динамически изменяющихся параметров заданий входного потока и распределенной вычислительной системы (количества ресурсов).

Для проведения экспериментов сгенерировано 1000 заданий большой относительной сложности при фиксированной интенсивности поступления заданий – 1000 и пуле, равным 1000 (т.е. все задания сразу же попадают в пул). Результаты планирования получены для универсальности заданий 10%, 50% и 90%.

Результаты планирования для различных критериев эффективности работы системы для универсальности заданий 10% показаны на рис. 12, 13.
Для универсальности 10%  

Рис. 12. Зависимость среднего времени выполнения заданий для FCFS, жадного алгоритма и метода MC для величины пула 1000 при универсальности заданий, равной 10 % от количества ресурсов (большое количество)

Полученные результаты планирования для универсальности заданий 50% и 90% свидетельствуют о том, что при увеличении универсальности заданий при использовании метода MC среднее время выполнения заданий увеличивается. При универсальности заданий 90% и количестве ресурсов 1000 время выполнения заданий при статическом планировании для MC значительно увеличивается (100 000 тактов), чем при динамическом (16000 тактов). При этом коэффициент использования ресурсов для MC при динамическом планировании составил 0.52, а при статическом планировании – 0.08. Таким образом, при увеличении универсальности заданий эффективность статического планирования для метода MC уменьшается. Это объясняется тем, что при использовании только одной процедуры планирования количество решений задачи о покрытии в процессе перепланирования резко увеличивается вследствие того, что задания, находящиеся в очереди, с увеличением универсальности практически все решаются на одних и тех же ресурсах, что, в свою очередь, приводит к увеличению времени нахождения заданий в очередях на ресурсы.

Следует отметить, что при статическом планировании коэффициент использования FCFS для 100 ресурсов является лучшим среди исследуемых методов, однако для 500 ресурсов, в отличие от метода MC, коэффициент использования для метода FCFS стремится к 0 (см. рис. 13).

Следует отметить, что при статическом планировании коэффициент использования FCFS для 100 ресурсов является лучшим среди исследуемых методов, однако для 500 ресурсов, в отличие от метода MC, коэффициент использования для метода FCFS стремится к 0 (см. рис. 13).

Полученные результаты планирования для универсальности заданий 50% и 90% свидетельствуют о том, что при увеличении универсальности заданий при использовании метода MC среднее время выполнения заданий увеличивается. При универсальности заданий 90% и количестве ресурсов 1000 время выполнения заданий при статическом планировании для MC значительно увеличивается (100 000 тактов), чем при динамическом (16000 тактов). При этом коэффициент использования ресурсов для MC при динамическом планировании составил 0.52, а при статическом планировании – 0.08. Таким образом, при увеличении универсальности заданий эффективность статического планирования для метода MC уменьшается. Это объясняется тем, что при использовании только одной процедуры планирования количество решений задачи о покрытии в процессе перепланирования резко увеличивается вследствие того, что задания, находящиеся в очереди, с увеличением универсальности практически все решаются на одних и тех же ресурсах, что, в свою очередь, приводит к увеличению времени нахождения заданий в очередях на ресурсы.

5 Выводы

Проведенные исследования показали, что при выборе стратегий планирования выполнения заданий в распределенных вычислительных системах необходимо учитывать факторы, влияющие на компоненты механизма планирования в зависимости от изменения характеристик поступающих на обработку задач и распределенной инфраструктуры. Для используемой модели планирования [1, 2] пакетов заданий такими компонентами являются величина пула, универсальность заданий и периодичность планирования.

В целом, полученные результаты позволяют сформулировать следующее:
для заданий большой относительной сложности ключевым фактором является периодичность планирования. Увеличение периодичности планирования приводит к ухудшению эффективности методов решения задач о покрытии. Однако, в тех случаях, когда периодичность планирования меньше величины (-3\(\sigma\)) относительно среднего времени освобождения ресурса, эффективность планирования не изменяется; универсальность заданий существенно влияет на результаты планирования. При увеличении универсальности заданий эффект использования методов планирования на основе метода покрытия уменьшается. При этом при универсальности заданий 90% для количества ресурсов 20, 30 результаты планирования при использовании этих алгоритмов практически идентичны; для заданий большой сложности использование большого пулла является нецелесообразным, так как за время решения заданий можно выполнить неоднократное перепланирование заданий на освободившиеся ресурсы, и, следовательно, принципиальным является не величина пулла, а периодичность планирования. При этом пул рекомендуется выбирать равным (или меньшим) количеству ресурсов (в исследовании рекомендуется выбирать величину пулла равной 10); применение статического метода пакетного планирования приводит к ухудшению результатов по сравнению с динамическим. Вместе с тем, при статическом планировании при низкой универсальности заданий метод МС позволяет получить лучшие результаты по времени выполнения заданий по отношению к динамическому при сохранении величины коэффициента использования ресурсов. Статическое планирование целесообразно использовать для заданий с большой сложностью, высокой интенсивностью входного потока заданий и большого количества ресурсов.

Литература

Structural Organization of the Router for the Multi-channel Computer Systems

V. Ye. Mukhin, Ya. I. Kornaga

National Technical University of Ukraine “Kiev Polytechnical Institute”, Pr. Pobedy, 37, Kiev, Ukraine

v_mukhin@mail.ru, slovyan_k@ukr.net

Abstract. In this paper we suggest a specialized router for distributed computer systems, which has the internal multichannel links. This router allows to eliminate the deadlocks in the routed data flows for the various routing mechanisms that supports a variety of simultaneous secured communication channels and allow to improve the functioning of the computer systems and security of the processed data there.

Keywords

Data flow, router, multichannel links.

1 Introduction. The statement of the problem

There is an effective mechanism to improve the data security in the distributed computer systems (DCS) – the directed routing, which provides the transfer of data on a communication channel with the criterion of the maximum safety for the transmitted data by each constituent sub-channels between the nodes. The safety criterion is defined as the probability of threats realization for the transmitted data security on a particular sub-channel. This probability is functionally related to the trust level to the node actions in the DCS.

Let us consider the case where the trust level parameter to nodes is varied in the range from 0 to 1. Let us show the certain topology of the distributed computer system with the predefined trust levels for the nodes action in DSC (Fig. 1).

Fig. 1. The topology of the DCS with the predefined trust levels for nodes actions

When the data is transmitted from the node A to the node M then there multiple paths are available. Thus, the data transmission is safe in case only if the trust level to the all nodes on the packets route must not fall lower than 0.6, then the data can be transmitted over a shorter path 1. In case if the requirement is that the trust level must not fall lower than 0.55, then the data transmission must pass through path 2, which is not the most optimal in terms of distance of the data
transfers. Thus, there is the need to develop a special mechanism for the data routing along a predetermined path that will allow to transmit the data with the criterion of trust level to the nodes actions in the DCS, which is functionally linked to the safety of the data transmission.

When we are implementing the directed routing there is raised another related problem. The directed routing means that the data are transmitted by a certain communication channel, which is formed by the criterion of the maximum data transfers safety. Let us want to send a message from the node A to the node M by the safest route: A – F – G – L – M (Fig. 2).

Fig. 2. The multiple data flows, which are intersecting in the topology of a distributed computer system

As a result there is formed a temporary channel A – F – G – L – M, which for certain period closes the possibility of transfer data by the other communication channels in a given topology, such as the channels D – C – G – L – K or E – F – G – H. In fact, there is a classical data interlocking that significantly increases the time of the data transmission. Furthermore, the heavy traffic recall the high probability that a packet will be blocked in the buffer, taking a number of buffers on his route, which in turn causes the increasing the time of the data packages transmission. The problem of channels interlocking also arises when there is implemented the routing mechanism based on the virtual channels. Thus, the actual problem is the implementation of a specialized router for distributed computer systems, allowing serve the multiple concurrent data flows.

2 The analysis of recent results and publications

The theory of the routers design is sufficiently developed. There are known a number of publications in this field, in particular [1,2].

The generalized structure of classical router for DCS and switched networks is shown in Fig. 3.

Fig. 3. The generalized structure of a router for the distributed computer systems and switched networks

This router performs the data processing and transmitting in 4 phases [3,4]:
- the input buffering: the data arrives in the input buffer;
- the forming of the routing paths: based on the destination address is determined the output port of the router;
- the bits transmission through the router: the data passes through the router to the output buffer;
- the bits transmission over the communication channel (CC): the data are transmitted over the communication channel to the next router.
OS - optical splitter

Fig. 4. The structure of a router with internal multi-channel communications
3 The router for distributed computer systems with the internal multi-channel links

To solve the problems discussed above on the data flows interlocking we propose to apply the mechanism of the multi-channel communications. Instead a single channel with the buffer in router we suggest to set a number of buffers which are operating with the parallel multi-channel links, so there the data are split for a certain number of sub-channels, and these sub-channels are independent and the data transmission on them can be performed in parallel, thereby eliminating the data transfers delays, associated with interlocking, which is caused by the capture of the packet buffers or the link.

The structure of the router with internal multi-channel communications is shown on Fig. 4.

For the channels multiplexing for the data transfers are may be used the protocols of the Time Division Multiple Access (TDMA) or the Frequency Division Multiple Access (FDMA). At the inputs of the router there is a buffer memory for the intermediate data storage. The choice of data to send is implemented in accordance with the routing algorithm. In the multi-channel communications there is a division of data between the multiple sources. To support the multi-channel mechanism are used the special data transfer protocols with the fixed or random access, which are realizing the certain scheme for channels division.

One of the most effective protocols to support multi-channel communication is ALOHA, which enables the communication between the source and the receiver in the multi-channel environment. The classic one, the ALOHA protocol without the time slots is rather difficult to manage, as packets can be transmitted at any time and are independent from each other.

The modifying one is a protocol Slotted ALOHA (S-ALOHA) which implements the traffic division by the slots and all packets are sent entirely within the slot, and this allows increased the throughput in comparing to the classical ALOHA protocol by reducing the number of conflicting packages due to the prohibiting the data transfer in next slot in case if there is a conflict and there is no partial overlap of conflicting packages.

It should be noted that the number of simultaneously used channels on the buffer level can be different, but a fixed number of channels is simplified the control mechanism for the multi-channeled transmission. The number of channels is determined by the traffic intensity, which in turn determines the number of required virtual channels, which are formed simultaneously. In the practice, the actual number of channels should be equal to the number of source nodes which are generate and transmit packets with a high intensity. Also, in case if the traffic intensity is increased the data interlocking may occur, but the number of the data locking cases is reduced substantially.

Thus, the suggested multi-channel mechanism allows to solve the problem of the channels interlocking and the buffers blocking, which allow increase the efficiency of routing mechanism in distributed computer systems. As a result, we may realize a number of the simultaneously operating secured sub-channel for the critical data transmission and the open communication channels for the open data transfers without any interlocking, which increases the data transfer efficiency and their security in the distributed computer systems.

4 Conclusion

There is suggested the internal multi-channel links mechanism in the routers to solve the problem of mutual blocking of data flows. The main idea is that instead of a single-channel communication buffer in the router are installed a number of parallel operating buffers and multi-links, which are divided the data channel on a certain number of sub-channels. Sub-channels are independent between themselves and the data on them is transferred in parallel, thereby eliminating the time delays associated with the interlocking, which are caused by the capture of packet buffers or links. The result is formed a number of simultaneously operating secured channel for the critical data transmission and the open communication channels for open data transmitting without any interlocking, which increases the efficiency and security for the data processing in the distributed computer systems.

References

An Effective Cluster Score Job Scheduling Algorithm for Grid Computing

Vinodhini N., Dr. Kamalam G.K.

Department of Information Technology, Kongu Engineering College, Perundurai, Erode-638052, India
Anna University, Chennai

vinoguru053@gmail.com, gkk@kongu.ac.in

Abstract. Grid computing is the collection of computer resources from multiple locations to reach a common goal. Distributed computing supports resource sharing. Parallel computing supports computing power. Grid computing aims to harness the power of both distributed computing and parallel computing. Grid can achieve the same level of computing power as a supercomputer does, also with a lowest cost. Grid is also a heterogeneous system. Grid comprises a collection of clusters. Scheduling independent task is more complicated in grid environment. In order to utilize the power of grid computing efficiently, an Adaptive Scoring Job Scheduling Algorithm (ASJS) assigns job to the resources by calculating the cluster scores. Cluster score is defined as the weighted value of the Average Transmission Power and Average Computing Power of cluster. An ASJS Algorithm is used to reduce the completion time of the submitted job, which may compose of computing-intensive jobs and data-intensive jobs in a grid environment.

Keywords
Grid Computing, Job Scheduling, Resource Allocation, Cluster Formation

1 Introduction

The computational capability and network performance have gone to a great extent, there are still problems in the fields of science, engineering, and business, which cannot be effectively dealt by using the current generation of supercomputers. The emergence of the Internet as well as the availability of powerful computers and high-speed network technologies as low-cost commodity components is rapidly changing the computing landscape and society. These technology opportunities have led to the possibility of using wide-area distributed computers for solving large-scale problems, leading to what is popularly known as Grid computing.

Grid computing is the collection of computer resources from multiple locations to reach a common goal. Grid can achieve the same level of computing power as a supercomputer does, but at a much reduced cost. Grid is like a virtual supercomputer. Distributed computing supports resource sharing. Parallel computing supports computing power. Grid computing aims to harness the power of both distributed computing and parallel computing. The goal of grid computing is to aggregate idle resources on the Internet such as Central Processing Unit (CPU) cycles and storage spaces to facilitate utilization.

Grid computing enables the sharing, selection and aggregation of a wide variety of geographically distributed resources including supercomputers, databases, data sources and specialized devices owned by different organizations. The Grid users need not be aware of the computational resources that are used for executing their applications and storing their data.

2 Problem definition

Grid environment is heterogeneous in nature. Jobs arriving the grid are heterogeneous in nature. The scheduling of the jobs to the appropriate resources in a heterogeneous environment is complicated. Grid comprises of ‘n’ number of resources and ‘m’ number of jobs to be scheduled. The scheduling problem consists of mapping ‘m’ jobs to the ‘n’ resources.
3 Related work

Saha et al (1995) proposed the Fastest Processor to Largest Task First Scheduling Algorithm (FPLTF) is a good representative for Bag-of-Tasks applications. The strategy of the FPLTF scheduling algorithm is to schedule jobs according to the workload of jobs and computing power of resources.

Maheswaran et al (1999) proposed the Min–min scheduling algorithm, each job will be always assigned to the resource which can complete it earliest in order to spend less time completing all jobs. The Max–min scheduling algorithm is similar to Min–min scheduling algorithm. It gives the highest priority to the job with the maximum earliest completion time. Fairness is the key idea of the Round Robin (RR) scheduling algorithm.

Maheswaran at al (1999) proposed the On-line mode heuristic scheduling algorithms. Jobs are scheduled as soon as it arrives. Because a grid environment is heterogeneous with different types of resources, on-line mode heuristic scheduling algorithms are more appropriate for grid environment. Dynamic FPLTF Scheduling Algorithm (DFPLTF)is based on FPLTF scheduling algorithm and is modified to make the FPLTF scheduling algorithm more adaptive for grid environment.

Xu et al (2003)proposed the simple grid simulation architecture and modified the basic ant algorithm for job scheduling in grid. The scheduling algorithm they proposed needs some information such as the number of CPUs, Million Instructions Per Second(MIPS) of every CPU for job scheduling. A resource must submit the information mentioned above to the resource monitor.

Sheng-De Wang et al(2005) proposed the Most Fit Task First Scheduling Algorithm (MFTF) mainly attempts to assign the most suitable resource to the task by a value called fitness.

Abraham Silberschatz et al (2011) proposed the Batch mode heuristic scheduling algorithms. In that Jobs are queued and collected into a set when they arrive in the batch mode. They will be scheduled afterwards by the scheduling algorithm. Batch mode heuristic scheduling algorithms are more appropriate for the environment with the same type of resources. First-Come, First-Served Scheduling Algorithm (FCFS) is the simplest algorithm for job scheduling. Jobs are executed according to the sequence of job submitting. The second job will be executed when the first job is done, and therefore FCFS has a serious problem called convoy effect. The convoy effect will happen when there is a job with large workload in the front of the job sequence. All other small workload jobs have to wait until the big one finishes.

4 Proposed system

First initialize GridSim Toolkit with n number of users and the router is initialized. The resource is registered in GIS. Create n number of resources with n number of machines. ‘n’ number of resources will be created and ID will be randomly generated for each resource. Each resource contain ‘n’ number of machines and each machine contain random number of PEs(Processing Elements) and MIPS(Million Instruction Per Second) is assigned for each PE. Cost and baud_rate is assigned to each PE. Job is called as Gridlet. ‘n’ number of Gridlet will be created and ID will randomly generated to each job. Length of the job is assigned.

Initialize the load of each resource. The user submit computing-intensive jobs and data-intensive jobs. The computing-intensive job means that jobs need lots of computing power to complete and the data-intensive job means that the resource needs to take lots of bandwidth to transmit files.

4.1 Computing power

\[ CP_k = CPU Speed_i \times (1 - load_i) \]  
\[ CP_k \] = Available Computing Power of resource k.

4.2 Average transmitting power

\[ ATP_i = \frac{\sum_{j=1}^{m} \text{Bandwidth available}_{ij}}{m-1}, i \neq j \]  
Bandwidth available_{ij} is the available bandwidth between cluster i and cluster j, m is the number of clusters in the entire grid system.
4.3 Average computing power

$\text{ACP}_i$ means the average available CPU power cluster $i$ can supply to the job and is defined as:

$$\text{ACP}_i = \frac{\sum_{k=1}^{n} \text{CPU Speed}_k \cdot (1 - \text{load}_k)}{n}, \quad i = 0 \text{ to } 2$$

(3)

$\text{CPU Speed}_k$ is the CPU speed of resource $k$ in cluster $i$, $\text{load}_k$ is the current load of the resource $k$ in cluster $i$, $n$ is the number of resources in cluster $i$.

4.4 Cluster score

The cluster score is defined as:

$$\text{CS}_i = \alpha \cdot \text{ATP}_i + \beta \cdot \text{ACP}_i, \quad i = 0 \text{ to } 2$$

(4)

$\text{CS}_i$ is the cluster score for cluster $i$, $\alpha$ and $\beta$ are the weight value of $\text{ATP}_i$ and $\text{ACP}_i$ respectively, the sum of $a$ and $b$ is 1, $\text{ATP}_i$ and $\text{ACP}_i$ are the average transmission power and average computing power of cluster $i$ respectively.

For a Grid environment the user might require a job to be cost effective or time effective. For a time effective job, the resource with lowest time in the chosen cluster is selected. For a cost effective job, the resource with lowest cost in the chosen cluster is selected. This is repeated for $n$ number of Gridlets. Finally Makespan and processing cost is calculated.

4.5 Scheduling jobs to cluster

After calculating the cluster score of each cluster, the Job Scheduler will select the resource with the best computing power in the cluster with the highest cluster score and assign the job to the resource. For a Grid environment the user might require a job to be cost effective or time effective. Based on the type of job, it is submitted.

After a resource receives a job, it starts to execute. The status of the resource will change, and therefore local update will be applied to adjust the cluster score of the cluster containing the resource. Once a job is completed by a resource, the result will be sent back and stored in the Information Server.

5 Result analysis

There are three clusters in the grid system and each cluster has three resources. The initial status of each resource is stored in Information Server Grid user_0 Creating 10 Gridlets. Job0 arrives at Job Scheduler, Job Scheduler will identify the type of job1 as computing-intensive and set the value of $\alpha$ and $\beta$. The value of $\alpha$ and $\beta$ are 0.3 and 0.7 respectively. Then Job Scheduler will calculate the ATP, ACP and CS of each cluster and select the resource with the best CP in the cluster with the highest CS to execute job0. According to the cluster score of each cluster, job0 will be submitted to resource 0 in cluster 0. Local update will be executed and the load of resource 0 becomes 35%.

Table: Resource and Job Characteristics

<table>
<thead>
<tr>
<th>Cluster</th>
<th>Resource ID</th>
<th>CPU Speed (MHZ)</th>
<th>Load (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>2796</td>
<td>30.0</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>2661</td>
<td>31.0</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>2652</td>
<td>26.0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>2835</td>
<td>33.0</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>3041</td>
<td>31.0</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>3060</td>
<td>28.0</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>3191</td>
<td>25.0</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>3146</td>
<td>32.0</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>3205</td>
<td>32.0</td>
</tr>
</tbody>
</table>

After Job Scheduler submits job0, it continues to schedule job1, because job2 is data-intensive, the value of $\alpha$ and $\beta$ set by Job Scheduler are 0.7 and 0.3 respectively. According to the cluster score of each cluster, job1 will be submitted to resource 3 in cluster 0. Local update will be executed and the load of resource 2 becomes 31%. Job0 is completed by resource 0 after Job Scheduler submits job1. The global update will be executed. Job2 will be scheduled by the Job Scheduler similarly.
For job 0 Minimum processing time 15.1 in Resource 0. This is repeated for n number of Gridlets. Finally Makespan and processing cost is calculated.

**Table:** Calculating Time and Cost

<table>
<thead>
<tr>
<th>Resource</th>
<th>Time (sec)</th>
<th>Cost ($)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>15.1</td>
<td>45.3</td>
</tr>
<tr>
<td>3</td>
<td>16.5</td>
<td>16.5</td>
</tr>
<tr>
<td>6</td>
<td>16.8</td>
<td>151.200</td>
</tr>
</tbody>
</table>

**Figure:** Cost and Time Efficient Jobs

From this graph job0 having time and cost efficient job among all these obtained jobs.

### 6 Conclusion

An cluster scoring method is to schedule jobs in grid environment. ASJS selects the fittest resource to execute a job according to the status of resources. Local and global update rules are applied to get the newest status of each resource. Local update rule updates the status of the resource and cluster which are selected to execute the job after assigning the job and the Job Scheduler uses the newest information to assign the next job. Global update rule updates the status of each resource and cluster after a job is completed by a resource. It supplies the Job Scheduler the newest information of all resources and clusters such that the Job Scheduler can select the fittest resource for the next job. The experimental results show that ASJS is capable of decreasing completion time of jobs and cost.

In future, ASJS will be implemented in real business grid applications. ASJS algorithm focuses on job scheduling. Modified ASJS shall be considered for division of file and the replica strategy in data-intensive jobs.

### References

Abstract. In this paper we propose a new design of an Operating System Kernel especially designed for HPC (High Performance Computing) systems. As technology is reaching the limits of silicon physics, it is not possible to substantially improve the CPU core’s performance by simply increasing the frequency or through new internal structures as there are no new ideas on how to increase single threaded performance substantially. This had led to a situation in which the theoretical performance of HPC systems can only be increased by integrating more CPU cores, making it necessary for the applications to be able to instantiate more and more threads to be able to make use of the increasingly amount of resources in the system. But often the overhead and jitter of the Operating System can make the application to perform worse as the level of parallelization increases. Currently, most HPC systems use adapted general purpose Operating System kernels in the compute nodes. In this paper we will discuss what problems this carries and we will describe an especially designed kernel that addresses HPC needs arguing that it can greatly help in reducing the Operating System’s overhead and contribute to improve jitter, enabling higher levels of concurrency and improving overall application performance by allowing more heavily threaded applications.

Keywords

Operating Systems, Kernel, HPC, overhead, jitter

1 Introduction

For decades now, the computing industry has relied on increasing clock frequency and architectural changes to improve performance. The situation has changed nowadays; small architectural improvements continue, but the power wall, the memory wall and thermal issues have made the approach of increasing clock frequency unfeasible. Semiconductor manufacturers continue developing the silicon manufacturing technology and doubling the number of transistor per unit area every few months, but rather than to improving the clock speed (because the physics do not allow it any more), they increase the number of processing units (i.e. cores) in the chip die.

This trend has been an uptake by HPC (High Performance Computing), continuously multiplying the number of nodes and core count in the HPC machines as the main way to increase the (theoretical peak) performance. Creating a situation in which only embarrassing parallel applications can effectively increase the execution performance by taking advantage of the additional resources. To illustrate this trend to, for example, 20 years ago, the most powerful computer had 140 cores (Numerical Wind Tunnel), 10 years ago 51200 (BlueGene/L) and in November 2013 over 3 million cores (Thianhe-2) [14].

In order to maximize the theoretical performance of new HPC system, developers must use an extremely high level of parallelism in their programs, rather than expect higher performance from the processor updates. But exploiting parallelism in any degree can be difficult: not only that the data must be segmented, but the communication overhead can lead to reduced performance. Also, the amount of work of concurrent threads must be sufficient to compensate for the overhead of thread creation and management. Unrolling loops with

* Work funded by BMBF (Bundesministerium für Bildung und Forschung) grant 01IH13003 - MyThos Projekt
short or little compute-intensive loop bodies can be too expensive and can greatly reduce the performance of the program.

The adequate load per loop body or per thread is dependent on several factors, the first is the code and data size transmitted to the destination node. It also has to be taken into consideration the operations executed by the Operating System to instantiate and configure the thread and the additional services needed to make the execution of the threads possible. Although in some Operating System kernels implementations threads have fewer dependencies than processes, the minimum execution environment for threads is still too large and leads to cache misses, instantiation challenges and expensive context switches when a thread is rescheduled or makes a system call.

This overhead is not just a problem that the programmer should cater for, but it is also a primary obstacle that keeps parallel applications from scaling better because it prevents the effective use of concurrency. Several studies have noted that the current architecture of Operating Systems generally have a negative impact on scalability and performance, and that an Operating System redesign for HPC is urgently needed [1][8][6][2].

Current approaches to increase scalability try to improve the data segmentation, to produce better adapted thread bodies and to reduce data dependencies. However, the Operating System is part of the core of the problem, and has not yet been correctly addressed. To correctly address this issue the architecture of Operating Systems for highly scalable applications has to be redesigned to eliminate secondary functionalities and reduce as much as possible the overhead.

Most computers in the Top500 use monolithic general purpose kernels, such as Linux, that internally have feature inter-dependencies of features that are not relevant for the execution of individual threads, but that increase memory and CPU usage [14]. For example, due to the expensive system calls, currently the creation of a thread requires over 10,000 compute cycles [8][16]. Although thread pool can be used to reduce this time, it also reduces the degree of dynamicity of the system.

Our aim is to reorganize the Operating System architecture so that it is exclusively dedicated to tasks related to computing, specially multithreading, with some extra functionality necessary for creating the execution environment specially designed for our purpose, the execution of HPC applications. These non-core functionalities of the Operating System are loaded accordingly to the needs of the system or the application at each moment. With this reorganization we expect to greatly increase the performance for thread creation.

To achieve this a modular approach will be used, taking concepts from the microkernel architecture paradigm, which means that the resources will only be minimally loaded by the Operating System, leaving more free resources to the program itself as in each node there will be deployed the minimal functionality for it to work. But this deployment will be done by generating a single execution context for the application and the Operating System, reducing the Operating System overhead, by having less expensive system calls, and improving the predictability of the system, which ultimately increases the performance of distributed synchronous systems.

This approach proposes an Operating System implementation dedicated to rapid deployment and management of threads on specific hardware architectures. The modular design of the Operating System will allow programs to interchange data between threads instantiated on different system architectures, supporting communication across different system boundaries (i.e. processor core to processor nodes, etc.). We expect that our Operating System kernel design to be a suitable solution for increasing the scalability of applications in multicore processors and multiprocessor nodes.

2 Current Operating Systems in HPC

Modern conventional Operating Systems are designed to support a wide range of needs and for this they provide a large set of interdependent daemons. In the computing nodes there might also happen to be hardware that is not needed by the program that is running at that moment and that is causing hardware interruptions. These processes (daemons) and interruptions that is not relevant to attend lead to a high overhead, unpredictable amount of CPU and memory usage and context switches.

The unpredictability of the Operating System and the system daemons behaviour affect specially communications. Current HPC systems typically use Message Passing Interface (MPI) or similar mechanisms for communication, as well as synchronization across computing nodes. Developers introduce specific synchronization events in the application code but if one core takes longer to reach to that point then all the other processors must wait, increasing the overall execution time as all nodes execute at the speed of the slowest. One of the main reasons that makes a processor take longer to reach a synchronization point than the others is
the unpredictable Operating System and daemons behaviour making HPC systems particularly sensitive to the Operating System footprint and overhead.

Many HPC vendors use custom Lightweight Kernel Operating System (LWK) that try to reduce this overhead by pursuing these goals and characteristics:

- Target massively parallel environments composed of thousands of processors with distributed memory with a tightly coupled network.
- Provide support for highly scalable, performance-oriented scientific applications.
- Emphasize efficiency over functionality.
- Maximize the amount of resources (e.g. CPU, memory, and network bandwidth) allocated to the application.
- All previous characteristics have the common aim of “Minimize time to completion for the application” [4].

The implementations of LWK may vary but they all try to give applications predictable and maximum access to the CPU, memory and other system resources. They often do this by simplifying algorithms such as thread scheduling and memory management and reduce system daemons to a minimum as computing nodes do not have to cater for multiple purpose computer services, such as enabling a multiuser and multitasking environment. In LWK systems, available services, such as job launch, are constructed in a hierarchical fashion to ensure scalability to thousands of nodes. Networking protocols for communication between nodes in the system are also carefully selected and implemented to ensure scalability, reducing latency (by reducing the network stack) and maximizing bandwidth [4].

By restricting services to only those that are absolutely necessary and by streamlining those that are provided, the jitter of the LWK Operating Systems is reduced, allowing a significant and predictable amount of the processor cycles to be given to the parallel application. Since the application can make consistent forward progress on each processor they will reach their synchronization points with better synchronization, as result, the node’s waiting time is reduced [12].

While computing nodes run the application and execute the calculations, the service nodes take care of the extra functionality. These are usually a small set of nodes running full service-like Operating System to offload services from the computing nodes (login access, compilation, job submission/launching, file I/O, etc.). All these functionality is needed for allowing users to run their application and interact with the system while providing security for the system and the user’s data.

Some good examples of systems that use custom LWK Operating Systems in the computing nodes are:

- Compute Node Linux (CNL): is a runtime environment based on the Linux kernel for the Cray XT3, Cray XT4, Cray XT5, Cray XT6, Cray XE6 and Cray XK6 supercomputer systems based on SUSE Linux Enterprise Server. It is part of the Cray Linux Environment (CLE) [18].
- The IBM Blue Gene supercomputers run various versions of Compute Node Kernel (CNK) Operating System based on Linux [7].
- Sandia National Laboratories has an almost two-decade commitment to Lightweight Kernels on its high-end HPC systems. It includes Operating Systems such as Catamount that was deployed on ASCI Red, and continues its work in LWKs with the Kitten OS [11].

Most of current LWKs are based on Unix-like Operating Systems, mainly using Linux, which is a monolithic general purpose kernel. In chapter 2.3 we will analyze that many of the general purpose Operating System kernel features are not needed in HPC computing nodes and can affect negatively the performance of the applications.

2.1 HPC System Architecture

In current HPC systems the activities of the computing nodes are orchestrated by “clustering middleware”, a software layer that sits on top of the nodes and allows the users to treat the cluster as one large cohesive computing unit (via a single system image concept).

The service nodes are the single point of management and job scheduling for the HPC, providing control and access to the computing resources and queuing the user’s jobs (programs they want to run) and their associated
tasks; allocates resources to these jobs, initializes the tasks on the computing nodes and reports the status of jobs, tasks, and compute nodes if necessary.

The most typical way of using this kind of systems consist in the user accessing the service node, sending their application source code and data, then compiling the application source code into a binary using an optimized compiler designed for the specifics of the system and its architecture and linking it to an also a set of optimized mathematical and communication libraries (such as BLAS or MPI). Then the user commits the compiled application with its data to the execution queue, that will decide when and on which specific computing nodes to execute the application. Once the execution is finished the user can retrieve the processed data from the system and run the application again with new data if desired.

In this paper, when we refer to a HPC Operating System, we are exclusively referring to the Operating System that is running in the computing nodes and not to the service or I/O nodes. The computing nodes are in most computers exclusively running one application of one user at a given time (although different nodes may execute different applications simultaneously), in other words, computing nodes are monouser and monotask in their nature.

The computing nodes work in an isolated environment, in which they do not have direct access to the “external world”, and the only communication point that they have is through the service nodes. That means that the computing nodes cannot suffer direct attacks from the external world. The advantage of this design is that there is no need to introduce security mechanisms in the computing nodes as all the possible attacks have to go through the service nodes, which should provide the convenient security mechanisms. The service node handle the user requests and determine the amount of computer nodes that each application and each user has available at each moment.

2.2 Current Issues

Even if they reduce the Operating System jitter, current node LWK Operating Systems are still far from an ideal situation as usually they still have these problems [15]:

- Context Switching overhead: Changing from the CPU’s user mode to kernel mode is in current systems very expensive. It has been measured, on the basic request “getpid”, to cost 1000-1500 cycles on most machines [9]. Of these just around 100 are for the actual switch (70 from user to kernel space, and 40 back), the rest is kernel overhead. In modern microkernels, such as the L3 microkernel, the minimization of this overhead reduced the overall cost to around 150 cycles [5]. But for other complex tasks typical microkernels are slower; microkernel Operating Systems attempt to minimize the amount of code running in privileged mode, for purposes of security and elegance, but ultimately sacrificing performance.

  - No real need for CPU execution modes: the purpose of distinct operating modes for the CPU is to provide hardware protection against accidental or deliberate corruption of the system environment (and corresponding breaches of system security) by software. Only trusted portions of system software are allowed to execute in the unrestricted environment (also known as “kernel mode”). All other software executes in one or more user modes. If a processor generates a fault or exception condition in a user mode, in most cases system stability is unaffected; if a processor generates a fault or exception condition in kernel mode, most Operating Systems will halt the system with an unrecoverable error. When a hierarchy of execution modes exists (ring-base security) faults and exceptions at one privilege level may destabilize only the higher-numbered privilege levels. Thus, a fault in ring 0 (the kernel mode with the highest privilege) will crash the entire system, but a fault in ring 2 will only affect rings 3 and beyond and ring 2 itself, at most.

  This feature is not strictly needed in HPC Computing Nodes as they only hold code and data of the application that is being executed in that right moment, so in case a user wanted to execute malicious code it could only attack itself. While at the same time it would not reduce functionality as faults and exceptions can still be handled in the most privileged mode.

- Cache Misses: Operating System kernel is too big, even when configured correctly the Operating System Kernel (normally Linux, 96.4% by November 2013 according to Top500 [10]) uses a notably large amount of memory. The impact on RAM usage is not noticeable (very few MiB vs many GiB), but the impact on processor cache is considerable. General purpose kernel’s data structures are very big, as they have to contain extra information for features not needed in computing nodes, and are not designed to make correct usage of cache lines as they are not specifically adapted for the processor on which they are running and compilers are not perfect in transforming them, the data structures, when generating the binary code. Also the data and code in memory should be organized so that they take advantages of the processor’s
data and instructions prefetch algorithms. The main idea is that the kernel should use as little as possible of the processor cache to evict the minimum of data / code belonging to the application [17].

• Operating System jitter: these systems still have many daemons running. In this kind of systems most of the daemons needed for a general purpose system has been removed, but some are still running. They provoke context switches, increasing overhead and evicting program’s data and code from the processor’s cache. They also reduce the predictability of the system worsening the inter-node synchronization times.

By improving the cache miss rate, eliminating context switches and improving the system’s predictability we can expect applications to create smaller body threads, increase resource usage and improve scalability, leading to a better performance and smaller time to completion.

2.3 General purpose Operating System features that HPC does not need

As explained in chapter 2.1, HPC systems are by nature, at least at computing node level, single user, and single task systems. For this reason there is no need for the Operating System to offer multiuser and multitasking services provided by general purpose Operating Systems, making, in principle, possible to get rid of many of the general purpose Operating System’s functionality:

• daemons: we should try reduce even further the amount of daemons running in the system (the ones left are monitoring, distributed workload management...) and if possible eliminate them completely. For this we need to find another solution to provide those services without using the standard daemon design. A solution is proposed in Chapter 3.

• big monolithic kernel: reduce even more the kernel size to minimize the disruption of the cache, and use simpler and faster system calls. The design is based on a minimalistic flexible microkernel that loads on demand the needed functionality by the application running on a specific computing node [13].

• context switches: context switches are expensive and introduce a lot of overhead. Try to reduce them or minimize the impact of them, or even try to completely eliminate them as proposed in Chapter 3.

• loaded drivers for all devices: to reduce memory usage and cache disruption, devices not needed by the application should be disabled. This also means that the system should disable interruptions sent from the devices not needed, which would also reduce context switches. It should be taken into consideration that some hardware, even if not used, will need some initial setup (for example, to enable power-saving modes). This can be done by loading the driver, executing the set-up, and then unloading the driver from memory, leaving the hardware correctly initialized while reducing the usage of the system’s memory.

3 Single-Context Operating System and Application Environment

The minimal and modular Operating System kernel design proposed in this paper tries to create a tight coupling of the Operating System, hardware drivers, service daemons and the application in a single context, thus completely removing the context switches and creating a low overhead and more predictable execution environment.

The Figure 1 depicts how the Operating System and applications are mapped to memory in a general purpose Operating System. The kernel and the applications are mapped into the same virtual address space (usually the Operating System is mapped to the same addresses for every application virtual address space, for example Linux maps the Operating System to the higher half leaving the lower half of the address space for the application) but as the Operating System kernel and the application are running on different CPU rings, executing functions implemented in the Operating System from the application require the execution of a special CPU instruction that provokes a switch in the CPU ring and jump into the Operating System code. We have to be aware that usually daemons run as if they were just another application running in “user mode” CPU ring, implying that to schedule them on a CPU core a full context switch is needed.

Each kernel thread or application thread shares all the virtual address space of the kernel and the application except for the stack of each thread (for both kernel and application threads), that is mapped privately; the stacks are situated on the same address segment, but in different virtual address spaces so each thread has only access to its own stack. This system provides independent address space contexts for each thread, providing security between applications, as different programs cannot access data and code from each other, and also easier multitasking, as all applications can be mapped exactly to the same memory addresses without interference.
Figure 1. Typical Operating System Memory Space organization, showing a main monolithic kernel image with a few kernel threads, and one application with many threads (there could be many applications executing, each with its own virtual memory address space). The kernel and the application are mapped to the same address space, except for the stack of each thread, but run on different CPU rings.

In our design we remove the multiple virtual address space contexts and merge them into a single address space, at the cost of decreasing inter-application security and, in principle, also multitasking facilities. This implies that we have to be able to map the Operating System kernel, service daemons, and application to the same address space, meaning that it has to be clear in the design the address range for each segment or we will have to use PIC (Position-Independent Code) in those systems that allow it. Figure 2 shows one alternative, of the many possible, of the memory organizations for a joined address space of the Operating System and the application. The segments we have to reserve in memory are these:

- Kernel Text: the binary code of the basic kernel and where the functionality extension modules are loaded.
- Drivers Text: hardware driver binaries.
- Daemons Text: binary code of the substitute of the functionality of the service daemons.
- Application Text: binary code of the user’s application running on the computing node.
- A general growing heap: dynamically created data structures used by the kernel, drivers, daemons and applications are stored here.
- Thread stack space: individual stack space is reserved for each thread, with its maximum size fixed at the thread’s creation point. The number of stack spaces will be the same as of threads instantiated in the computing node.
- Hardware and file mapping space: memory address space reserved for mapping opened files from I/O devices and hardware resources.
- Firmware mapping space: space reserved for mapping the computing node’s firmware (a.k.a. Bios or UEFI in x86 based systems).

The memory address layout shown in the Figure 2 is an example of a possible organization the main “memory blocks”, and the final organization will depend on the particularities of the different hardware architectures. Also the ranges of addresses shown for each memory block are examples and have not been determined yet as this requires further study and will also change depending on the underlying hardware architecture.
In modern Operating Systems each thread stack is in a different memory address space (even when they share the rest of the address space with the other threads of the same application), but this would imply that changing the execution thread on a certain CPU core would imply to set-up the new stack, that is forcing a context switch, affecting negatively the Operating System overhead (because a context switch, on most architectures, flushes the whole TLB, thereby forcing page table walks, which implies a few memory reads). In our case, for changing the execution thread on a CPU core would be enough with setting correctly the stack registers.

From the user’s point of view, with this new design the application concept and behavior will remain the same as in current systems with the service nodes providing the user services (login, compilation, security...), there will be no changes on how the system is used or the programming model for the applications; users can continue using the system in the same way and they can program their applications using the same programming language, except for some of the Operating System calls.

The complexity of the service nodes would not be increased with the model discussed in this paper as most of the modifications would have to be implemented into computing node, the toolchain and the deployment of the application.

### 4 Application Life Cycle and Toolchain Modifications

To be able to successfully achieve the generation of single context tasks that are embedding the application, the Operating System, hardware drivers and the service daemons a new compilation schema is needed. Depending on how the toolchain and software stack is finally designed and implemented there might be the drawback that the application has to be recompiled, or at least relinked, whenever there is an update in the system’s hardware or main parts of the software stack (such as when the kernel’s ABI is changed). Although this situation could be mitigated by using PIC code and dynamically linked object code (similar to Windows DLLs or ELF shared libraries) in other situations.

The compiler has to be aware of the location of the Operating System calls and services, has to integrate the interruption handling with the application, and be aware of where the application text is going to be placed in memory. This design allows that many typical small Operating System calls could in principle be inlined in the application making these operations much faster than in typical Operating System kernel but the situations in
which this possibility is used might be carefully analyzed as the code duplication could in some architectures increase the cache miss rate, especially in often used functionality.

Conventional Operating System designs the system calls are implemented by the application throwing a trap, a.k.a. a software interruption (INT in x86 architecture [3]) a parameter that is the identifier of the desired system call, usually stored in a register. This is necessary in order to raise the CPU ring and decreasing it again to return from the system call. In our design, as the application is merged with the Operating System, executing under the same CPU ring and sharing the memory context, the system calls are implemented as simple function calls, thus greatly reducing the context switch overhead.

Figure 3 shows the application life cycle of the new design comparing it to current HPC application life cycle and pointing out the main differences:

0. Application design and implementation: the user writes the application and debugs it in its own system, as the redesign of the Operating System kernel affects the computing node software stack and modifies some of the common Unix/POSIX system calls a special library should be provided so that the application can run on top of the Operating System of the developer’s workstation (this can easily be achieved by the library translating the system calls into the native ones).

1. Access to the HPC machine: the user logs into the service node where he stores the code of the application and the data, as done in current systems.

2. Compilation: then the application is compiled using the specialized toolchain. This step is the same for the user as compared to current system but as described before, the binary generated by the compiler will have a different structure.

3. Application submission to the execution queue: the user submits the application and the data to run to the HPC workload management, which will automatically determine the assignment of the computing nodes and execute it in the moment it considers appropriate.

4. Computing Node setup: link the minimal Operating System with the necessary modules, drivers and the daemons replacement code with the application, generating the single context task binary for each different type of node in case of an heterogeneous environment, otherwise a single image might be enough.

5. Deployed on the computing nodes: deploy the binaries generated in the previous step accordingly with the requested topology for the execution of the application and the resources assigned by the workload manager of the service nodes.

6. Execution: run the application and store the generated output in the service nodes (or the I/O nodes), where the user can retrieve de results).

7. Cleaning: the computing nodes are “cleaned”; the application is removed and computing node is ready to accept new applications.

8. Service nodes can launch a new application as determined by the workload manager.

We can conclude that the application life cycle suffers some changes from the point of view of how it is internally handled by the HPC system but from the user’s point of view it remains mostly unchanged.

5 Possible limitations of a Single Context Operating System

The new kernel design proposed in this paper imposes a set of limitations in its capabilities and functionalities in comparison to modern general purpose kernels, such as Linux, of which the most relevant are:

- Lost of Address Space Contexts: this happens because the application and the Operating System are running in the CPU’s kernel ring and sharing the memory address space. As the computing nodes are isolated from the external world, software attacks do not have a possibility to happen. The security with the “external” world is controlled by the service nodes, that includes I/O operations and user data control.

- No Multitasking: there will be no possibility of multitasking inside a computing node, that should not be confused with a single task being able to execute many threads, which is something needed. This should not change how HPC systems work, as currently they do not offer multitasking at Compute Node level, even if in principle it could be possible.
Figure 3. Application life cycle in HPC with the proposed Operating System design

- No Multiuser: to allow multiple users in the same system first you need multitasking. As this is not supported, then multiple users is also not supported. But as with the multitasking, this will not generate any impact in current HPC systems as they currently do not support multiple users at the compute node level.

- Crashed Application Recovery: a computing node might completely halt (crash) because of a hardware failure or a software bug. In conventional systems the Operating System can cater for crashes affecting applications by closing them and resetting the status of the computing node to a usable state to accept new applications. But even conventional HPC systems need mechanism for externally resetting a computing node in case the Operating System kernel crashes, for example to recover from a Linux “kernel panic”. In the proposed design many of the application’s bugs and the Operating System crashes will have to be resolved externally, this means we might need the computing node resetting mechanisms more often than in current systems, but these mechanisms are, in any case, already implemented.

In summary, it can be observed that we lose some features of modern general purpose Operating Systems, but these are the features that, even if available, are not used in the HPC computing nodes making our approach feasible without affecting the user experience and without the need of a different hardware design.

6 Conclusion

In this paper we have argued that general purpose Operating System kernels, even when customized, are not well suited for modern HPC computing nodes as these do not need kernel features that are currently integrated and just increase the unpredictability of the system (affecting negatively the Operating System jitter), worsen the cache miss ratio and have high overhead for system calls.

For this reason we propose a complete redesign of the Operating System kernel for HPC, taking into consideration the modularity concept of microkernels and loading in run-time the features needed by the compute nodes and the application with the aim to reduce system’s overhead, improve the cache miss rate and increase the predictability. All this together will allow system calls to execute faster, and improve the synchronization...
of the system, enabling applications to be decomposed in smaller threads, as the amount of work of the body of
the thread does not need to be so big to compensate the thread creating overhead, increasing the level of usage
of resources of modern HPC systems due to the better scalability, thus reducing the total time to completion
of the application.

This is achieved without big changes from the user’s perspective as all the changes are integrated in the
Operating System and the toolchain that generates the binary code of the application. As the proposed kernel
design and functionality is a new philosophy and implies changes that drop not needed functionality provided
by general purpose kernels, many of the typical system calls of Unix/POSIX systems will have to be modified
with the implication that compatibility with legacy code that uses the eliminated or modified calls will have to
be adapted. But it should be considered that many of these often provided calls by general purpose kernels are
not often used in HPC programs or they will be provided in a different, and often more simple, manner.

As we progress the research, development and evaluation of this new design, further publications will be
published addressing the challenges that emerge and evaluating different solution, detailing the algorithms and
implementations used.

References

  Singhania: The multikernel, a new OS architecture for scalable multicore systems. Proceedings of the ACM

  An analysis of Linux scalability to many cores. Proceedings of the 9th USENIX conference on Operating


  ples, SOSP, December 1995.


  Multicore Architectures workshop, 2011.

[9] J. Ousterhout: Why aren’t Operating Systems getting faster as fast as hardware? Usenix Summer Con-

    November 2013.


An Energy Efficient Network Life Time Enhancement Proposed Clustering Algorithm for Wireless Sensor Networks

Ankit Sharma, Jawahar Thakur

H.P. University Shimla
Ankitcse.engg07@gmail.com

Abstract: Wireless sensor networking is an emerging technology that promises a wide range of potential applications in both civilian and military areas. A wireless sensor network (WSN) typically consist of a large number of low cost, low power and multi-functional sensor nodes that are deployed in a region of interest. Wireless sensor networks face many challenges caused by communication failures, storage and computational constraints and limited power supply. In WSN, the nodes are battery driven and hence energy saving of sensor nodes is a major design issue. Energy efficient algorithms must be implemented so that network lifetime should be prolonged. Lifetime of a network can be maximized through clustering algorithms, where cluster is responsible for sending the data to the base station and not all the nodes are involved in data transmission. Various clustering algorithms are deployed in past few years. In this paper we are proposing an algorithm which is a combination of Bacterial foraging optimization algorithm (BFO) which is a Bio-Inspired algorithm and LEACH and HEED protocols which enhances the lifetime of a network by dissipating minimum amount of energy.

Keywords

WSN, Bacterial Foraging Optimization, LEACH, HEED.

1 Introduction

Wireless sensor networks [3] are an emerging technology that has potential applications in surveillance, environment and habitat monitoring, structural monitoring healthcare and disaster management. A WSN monitors an environment by sensing its physical properties. It is a network of tiny, autonomous nodes that can acquire process and transmit sensory data over wireless medium. Typically, sensor nodes are grouped in clusters and each cluster has a cluster head. All the nodes forward their sensor data to the cluster head, which in turn routes it to a specialized node called sink node (Base Station) through a multi-hop wireless communication as shown in the figure. However often the sensor network is very small and consists of a single cluster with a single base station. Other scenarios such as multiple base stations or mobile nodes also possible.

![Figure 1. Sensor nodes scattered in a sensor field.](image)
WSN issues such as node deployment, localization, energy aware clustering and data aggregation are often formulated as optimization problems. Traditional analytical optimization techniques such as linear, non-linear and quadratic programming, Newton-based techniques and interior-point methods require enormous computational efforts, which grow exponentially as the size of the problem increases. An optimization method that requires moderate memory and computational resources and yet produces good results is desirable, especially for implementation on an individual sensor node. These give motivation for heuristic algorithms such as PSO [10], [5], Genetic algorithm (GE) [6], differential evolution (DE) [7] and Bacterial Foraging Algorithm (BFO) [1]. Bio-Inspired optimization methods are computationally efficient to analytical methods.

2 Architecture of wireless sensor network


(a). **Sensor node**: a sensor node is the core component of the wireless sensor network because it can perform multiple roles in a network, such as sensing, data processing, data storage and routing.

(b). **Clusters**: Clusters are the organization unit for WSNs. The dense nature of these networks requires the need for them to be broken down into clusters to simplify tasks such as communication.

(c). **Cluster Heads**: Cluster heads are the organization leader of a cluster. They often are required to organize activities in the cluster. These tasks include but are not limited to data aggregation and organizing the communication schedule of a cluster.

(d). **Base Station**: the base station is at the upper level of hierarchical WSN. It provides the communication link between the sensor network and the end user.

(e). **End-User**: The data in the sensor network can be used for a wide range of applications [3]. Therefore for a particular application may make use of the network data over the internet, using a PDA, or even a desktop computer. All the processing is done at the end user when data from various nodes are collected.

3 Background study

3.1 Low-Energy Adaptive Clustering Hierarchy (LEACH)

LEACH [2] protocol that is proposed by Heinzelman et al is an elegant solution to power constraint problem, by forming enough number of clusters in a self organized manner. It is the first dynamic cluster head protocol specifically for WSN using homogenous stationary sensor nodes randomly deployed. LEACH is suited for applications which involve constant monitoring and periodic data reporting. LEACH protocol runs in many rounds. Each round contains two phases: cluster setup phase and steady phase.

In cluster setup phase, it performs organization of cluster and selection of cluster head. Selected cluster heads broadcast a message to all the other sensors in the network informing that they are the new cluster heads. All non cluster head nodes which receive this advertisement decide which cluster they belong to based on the signal strength of the message received. All non cluster head nodes transmit their data to the cluster head, while the cluster head transmit the data to the remote station that is the base station (BS). Cluster head node is much more energy sensitive than being a non-cluster node. Head nodes would quickly use up their limited energy. Thus, LEACH incorporates randomized rotation of the high-energy cluster head among the sensors.

The sensor nodes elect themselves to be cluster head at any given time with a given probability. The decision of whether a node elevates to cluster head is made dynamically at a time interval. The elevation decision is to be made solely by each node independent of other nodes. This is done to minimize overhead in cluster head establishment. This decision making is a function of the percentage of optimal cluster heads in a network (to be determined prior to application) in combination with how often and the last time a given node has been a cluster head in the past. The threshold function is defined as

\[ T(n) = \{P/ (1-P \mod (1/P)) \text{ if } n \in G \]

\[ 0 \quad \text{otherwise} \]
Where \( n \) is the given node, \( P \) is the a priori probability of a node being elected as a cluster head, \( r \) is the current round number and \( G \) is the set of nodes that have not been elected as cluster heads in the last \( 1/P \) rounds. Each node during Cluster head selection will generate a random number between 0 and 1. If the number is less than the threshold \( (T(n)) \) the node will become a cluster head.

### 3.2 Hybrid energy-efficient distributed clustering (HEED)

Hybrid Energy-Efficient Distributed Clustering (or HEED) [4] is a multi-hop clustering algorithm for wireless sensor networks, with a focus on efficient clustering by proper selection of cluster heads based on the physical distance between nodes. The main characteristics [ ] of HEED algorithm are to:

(a). Distribute energy consumption to prolong network lifetime;
(b). Minimize energy during the cluster head selection phase;
(c). Minimize the control overhead of the network.

The most important aspect of HEED is the method of cluster head selection. The selection of cluster head is basically depends upon the following two parameters:

1. **The residual energy** of each node is used to probabilistically choose the initial set of cluster heads. This parameter is very commonly used in many other clustering schemes also.

2. **Intra-Cluster Communication Cost** is used by nodes to determine which cluster they are going to join. This scenario is useful if a given node falls within the range of more than one cluster head. In HEED it is important to identify the range of a node in terms of its power levels as a given node will have multiple discrete transmission power levels. The power level used by a node for intra-cluster announcements and during clustering is referred to as cluster power level. Low cluster power levels promote an increase in spatial reuse while high cluster power levels are required for inter-cluster communication as they span two or more cluster areas.

Therefore, when choosing a cluster, a node will communicate with the cluster head that yields the lowest intra-cluster communication cost. The intra-cluster communication cost is measured using the Average Minimum Reach-ability Power (AMRP) measurement. The AMRP is the average of all minimum power levels required for each node within a cluster range \( R \) to communicate effectively with the \( i \)th cluster head. The AMRP of \( i \)th node then become a measure of the expected intra-cluster communication energy if this node is elevated to cluster head. Utilizing AMRP as a second parameter in cluster head selection is more efficient then a node selecting the nearest cluster head.

### 3.3 Bio-inspired algorithm [9]

#### 3.3.1 Bacterial foraging algorithm (BFO)

Bacterial Foraging [8] Optimization Algorithm (BFO) [1] is a well-known computational methodology which is based on the study of the bacterial foraging behaviors. The complex but organized activities exhibited in bacterial foraging patterns could inspire a new solution for optimization problems. The underlying mechanism of the surviving of bacteria, especially \( E. \ coli \) in a complex environment has been reported by researchers in the area of biological sciences. Inspired from these phenomena, BFO was developed as an optimization algorithm by K. M. Passino [1], in which the self-adaptability of individuals in the group searching activities has attracted a great deal of interests. The classical bacterial foraging optimization (BFO) system consists of three principal mechanisms, namely, chemotaxis, reproduction, and elimination dispersal.

**Chemotaxis:** This process simulates the movement of an \( E.Coli \) cell through swimming and tumbling via flagella. Biologically an \( E.Coli \) bacterium can move in two different ways. It can swim for a period of time in the same direction or it may tumble, and alternate between these two modes of operation for the entire lifetime.

**Swarming:** An interesting group behavior has been observed for several motile species of bacteria including \( E.coli \) and \( S. \) typhimurium, where intricate and stable spatio-temporal patterns (swarms) are formed in semisolid nutrient medium. A group of \( E.coli \) cells arrange themselves in a traveling ring by moving up the nutrient gradient when placed amidst a semisolid matrix with a single nutrient chemo-effector. The cells when stimulated by a high level of \( \text{succinate} \), release an attractant \( \text{aspartate} \), which helps them to aggregate into groups and thus move as concentric patterns of swarms with high bacterial density.
Reproduction: The least healthy bacteria eventually die while each of the healthier bacteria (those yielding lower value of the objective function) asexually split into two bacteria, which are then placed in the same location. This keeps the swarm size constant.

Elimination and Dispersal: Gradual or sudden changes in the local environment where a bacterium population lives may occur due to various reasons e.g. a significant local rise of temperature may kill a group of bacteria that are currently in a region with a high concentration of nutrient gradients. Events can take place in such a fashion that all the bacteria in a region are killed or a group is dispersed into a new location. To simulate this phenomenon in BFOA some bacteria are liquidated at random with a very small probability while the new replacements are randomly initialized over the search space.

4 Proposed algorithm

All the clustering algorithms that we have studies here have their own pros and cons. In our proposed algorithm we primarily focus on increasing the life-time of the network so that energy nodes will not die soon and the life time of network should be prolonged. The proposed algorithm is implemented in 3 phases

(a). Phase 1: the nodes will be moved within the cluster through the foraging strategy of the E.Coli bacteria that is proposed in bacterial foraging optimization (BFO) [1]. The sensor nodes are moved within in the cluster so that a proper inter-node distance (calculated prior to the application) is maintained. This distance is enforced into the algorithm because if the nodes move so close then it may happen after n iterations that they accumulate in the same region. It should be avoided because if they are accumulated in the small area then the data they sense will be duplicated and that is not desirable in the network. Hence a proper inter node distance is maintained. Also, it may happen that some nodes are moved so far away from the other nodes in the cluster then they will took more energy from the battery and will die soon. This is also not a favorable situation because we know that wireless sensor nodes are battery driven and it is not possible in most cases to replace the battery, as our aim in this algorithm is to prolong the network lifetime.

(b). Phase 2: In the second phase of the algorithm we select the cluster head according to the LEACH [2] protocol and the HEED [4] protocol. This will work as; the selection of the cluster head is according to the probability function given in the LEACH protocol and the residual energy of the sensor node. The more is the residual energy the more are the chances of a sensor node to become a cluster head. It is because of the fact that as we know that cluster head communicates directly with the base station so cluster head will require more energy. Hence the sensor nodes that have the highest residual energy have the highest probability of being selected as cluster head.

(c). Phase 3: In the third phase of the algorithm, the actual transmission of the data is done. In this phase the cluster heads gather the data sensed by the sensor nodes in their cluster region and then it will send the data to the base station. The transmission can be either by a multi-hop method or can be single hop. In multi-hop method, the cluster head can send data to another cluster head and it can then send to the base station. This scenario is followed because sometimes some cluster head are far away from the base station and if they directly send the data to the base station they will deplete their energy very soon. Hence it can be avoided by multi-hop method. In single hop method, cluster heads are directly send the data to the base station. This scenario is helpful in those cases where the network is very small.

5 Limitation of the work

The proposed algorithm is just a proposal and the algorithm is justified only when the simulation will be carried out by using a proper simulator. Hence the work is purely an idea for the clustering in the wireless sensor networks so that network life time should be increased.

6 Conclusions and future work

Scalability and density of deployments, environmental uncertainties and constraints in energy, memory, bandwidth and computing resources pose serious challenges to the developers of WSNs. Issues of node deployment; localization, energy-aware clustering and data aggregation are often formulated as optimization problems. Many analytical methods suffer from slow or lack of convergence to the final solutions. The bio-inspired algorithms are inspired from the nature’s perfection to deal with complex scenarios. So the wireless networks developers are learning from the nature how it’s dealing with problems from ages. . In this paper we have proposed an energy efficient algorithm which enhances the lifetime of the wireless sensor network.
Future work can be done by simulating the above stated algorithm by using simulators by setting up the parameters to check whether the algorithm is enhancing the lifetime of the network or not. The performance analysis of the algorithm must be performed by doing the simulations.

References


<table>
<thead>
<tr>
<th>Authors</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Abdoldina F.</td>
<td></td>
</tr>
<tr>
<td>Akhmedov D.</td>
<td></td>
</tr>
<tr>
<td>Anjomshoa M.</td>
<td></td>
</tr>
<tr>
<td>Bantikyan H.</td>
<td></td>
</tr>
<tr>
<td>Belan V.</td>
<td></td>
</tr>
<tr>
<td>Belous L.</td>
<td></td>
</tr>
<tr>
<td>Bopeyev T.</td>
<td></td>
</tr>
<tr>
<td>Chagovets V.</td>
<td></td>
</tr>
<tr>
<td>Cherepynets V.</td>
<td></td>
</tr>
<tr>
<td>Egupov S.</td>
<td></td>
</tr>
<tr>
<td>Grechanovsky A.</td>
<td></td>
</tr>
<tr>
<td>Huthaifa A.</td>
<td></td>
</tr>
<tr>
<td>Ivanov A.</td>
<td></td>
</tr>
<tr>
<td>Kamalam G.</td>
<td></td>
</tr>
<tr>
<td>Karachevtsev V.</td>
<td></td>
</tr>
<tr>
<td>Khodakovskyy M.</td>
<td></td>
</tr>
<tr>
<td>Komukhaev E.</td>
<td></td>
</tr>
<tr>
<td>Kornaga Y.</td>
<td></td>
</tr>
<tr>
<td>Kosevich M.</td>
<td></td>
</tr>
<tr>
<td>Loutsky G.</td>
<td></td>
</tr>
<tr>
<td>Maslianko P.</td>
<td></td>
</tr>
<tr>
<td>Merjvinskyy P.</td>
<td></td>
</tr>
<tr>
<td>Minukhin S.</td>
<td></td>
</tr>
<tr>
<td>Mukhin V.</td>
<td></td>
</tr>
<tr>
<td>Muratov D.</td>
<td></td>
</tr>
<tr>
<td>Nallasivam V.</td>
<td></td>
</tr>
<tr>
<td>Natsik V.</td>
<td></td>
</tr>
<tr>
<td>Ogar T.</td>
<td></td>
</tr>
<tr>
<td>Polyakov V.</td>
<td></td>
</tr>
<tr>
<td>Rubin Y.</td>
<td></td>
</tr>
<tr>
<td>Rubio Bonilla D.</td>
<td></td>
</tr>
<tr>
<td>Rudenko P.</td>
<td></td>
</tr>
<tr>
<td>Salleh M.</td>
<td></td>
</tr>
<tr>
<td>Schubert L.</td>
<td></td>
</tr>
<tr>
<td>Sharma A.</td>
<td></td>
</tr>
<tr>
<td>Smirnov S.</td>
<td></td>
</tr>
<tr>
<td>Stepanian S.</td>
<td></td>
</tr>
<tr>
<td>Thakur J.</td>
<td></td>
</tr>
<tr>
<td>Torgonin E.</td>
<td></td>
</tr>
<tr>
<td>Yelubayev S.</td>
<td></td>
</tr>
<tr>
<td>Zobnina V.</td>
<td></td>
</tr>
<tr>
<td>Zolot A.</td>
<td></td>
</tr>
</tbody>
</table>