FUTURE COMPUTING 2013

The Fifth International Conference on Future Computational Technologies and Applications

ISBN: 978-1-61208-272-1

May 27- June 1, 2013

Valencia, Spain

FUTURE COMPUTING 2013 Editors

Alexander Gegov, University of Portsmouth, UK

Petre Dini, Concordia University, Canada | China Space Agency Center, Beijing, China

FUTURE COMPUTING 2013

Foreword

The Fifth International Conference on Future Computational Technologies and Applications (FUTURE COMPUTING 2013), held between May 27 and June 1, 2013 in Valencia, Spain, targeted advanced computational paradigms and their applications. The aim was to cover (i) the advanced research on computational techniques that apply the newest human-like decisions, and (ii) applications on various domains. The new development led to special computational facets on mechanism-oriented computing, large-scale computing and technology-oriented computing. They are largely expected to play an important role in cloud systems, on-demand services, autonomic systems, and pervasive applications and services.

We take here the opportunity to warmly thank all the members of the FUTURE COMPUTING 2013 Technical Program Committee, as well as the numerous reviewers. The creation of such a broad and high quality conference program would not have been possible without their involvement. We also kindly thank all the authors who dedicated much of their time and efforts to contribute to FUTURE COMPUTING 2013. We truly believe that, thanks to all these efforts, the final conference program consisted of top quality contributions.

Also, this event could not have been a reality without the support of many individuals, organizations, and sponsors. We are grateful to the members of the FUTURE COMPUTING 2013 organizing committee for their help in handling the logistics and for their work to make this professional meeting a success.

We hope that FUTURE COMPUTING 2013 was a successful international forum for the exchange of ideas and results between academia and industry and for the promotion of progress in the field future computational technologies and applications.

We are convinced that the participants found the event useful and communications very open. We hope that Valencia, Spain provided a pleasant environment during the conference and everyone saved some time to explore this historic city.

FUTURE COMPUTING 2013 Chairs:

FUTURE COMPUTING General Chair

Javier Aguiar, University of Valladolid, Spain

FUTURE COMPUTING Advisory Chairs

Cristina Seceleanu, Mälardalen University, Sweden
Hiroyuki Sato, The University of Tokyo, Japan
Miriam A. M. Capretz, The University of Western Ontario - London, Canada
Kendall E. Nygard, North Dakota State University - Fargo, USA
Vladimir Stantchev, Berlin Institute of Technology, Germany
Marek J. Druzdzel, University of Pittsburgh, USA

FUTURE COMPUTING 2013 Industry/Research

Francesc Guim, Intel Corporation, Spain
Wolfgang Gentzsch, Expert HPC, Germany

FUTURE COMPUTING 2013

Committee

FUTURE COMPUTING General Chair

Javier Aguiar, University of Valladolid, Spain

FUTURE COMPUTING Advisory Chairs

Cristina Seceleanu, Mälardalen University, Sweden
Hiroyuki Sato, The University of Tokyo, Japan
Miriam A. M. Capretz, The University of Western Ontario - London, Canada
Kendall E. Nygard, North Dakota State University - Fargo, USA
Vladimir Stantchev, Berlin Institute of Technology, Germany
Marek J. Druzdzel, University of Pittsburgh, USA

FUTURE COMPUTING 2013 Industry/Research

Francesc Guim, Intel Corporation, Spain
Wolfgang Gentzsch, Expert HPC, Germany

FUTURE COMPUTING 2013 Technical Program Committee

Ofer Arieli, The Academic College of Tel-Aviv, Israel
Mohsen Askari, University of Technology Sydney, Australia
Jan Blech, fortiss GmbH, Germany
Radu Calinescu, Aston University-Birmingham, UK
Alberto Cano, University of Córdoba, Spain
Miriam A. M. Capretz, The University of Western Ontario - London, Canada
Massimiliano Caramia, University of Rome "Tor Vergata", Italy
Chin-Chen Chang, Feng Chia University, Taiwan, R.O.C.
Sung-Bae Cho, Yonsei University, Korea
Rosanna Costaguta, Universidad Nacional de Santiago del Estero, Argentina
Zhihua Cui, Taiyuan University of Science and Technology - Shanxi, China
Marc Daumas, INS2I & INSIS (CNRS), France
Isabel Maria de Sousa de Jesus, ISEP-Institute of Engineering of Porto, Portugal
Leandro Dias da Silva, Federal University of Alagoas, Brazil
Marek J. Druzdzel, University of Pittsburgh, USA
Dietmar Fey, Friedrich-Alexander-University Erlangen-Nuremberg, Germany
Francesco Fontanella, Università degli Studi di Cassino e del Lazio Meridionale, Italy
Miguelo Garcia, Polytechnic University of Valencia, Spain
Wolfgang Gentzsch, Consultant, Germany
Victor Govindaswamy, Texas A&M University-Texarkana, USA
Michael Grottke, University of Erlangen-Nuremberg. Germany

Miguel Angel Guevara López, Universidade do Porto, Portugal
Francesc Guim, Intel Corporation, USA
Yongjian Hu, University of Warwick - Coventry, UK
Jong-Gyu Hwang, Korea Railroad Research Institute (KRRI), Korea
Muhammad Iftikhar, Universiti Malaysia Sabah (UMS), Malaysia
Dalia Kriksciuniene, Vilnius University, Lithuania
Carlos León de Mora, Universidad de Sevilla, Spain
Lu Liu, University of Derby, UK
Yu-lung Lo, Chaoyang University of Technology, Taiwan
José María Luna, University of Córdoba, Spain
Constandinos Mavromoustakis, University of Nicosia, Cyprus
Ehsan Mousavi, University of Science and Technology - Tehran, Iran
Susana Munoz Hernández, Universidad Politécnica de Madrid, Spain
Kazumi Nakamatsu, University of Hyogo, Japan
Isabel L. Nunes, Universidade Nova de Lisboa - Caparica, Portugal
Panos M. Pardalos, University of Florida, USA
Przemyslaw Pochec, University of New Brunswick, Canada
Alex Popa, Aalto University, Finland
Aurora Pozo, Federal University of Paraná, Brazil
Prakash Ranganathan, University of North Dakota-Grand Forks, USA
Ivan Rodero, NSF Center for Autonomic Computing / Rutgers the State University of New Jersey, USA
Aitor Rodriguez, University Autonoma of Barcelona, Spain
Hiroyuki Sato, The University of Tokyo, Japan
Cristina Seceleanu, Mdh, Sweden
Georgios Sirakoulis, Democritus University of Thrace - Xanthi, Greece
Devan Sohier, Université de Versailles - St-Quentin-en-Yveline, France
Young-Joo Suh, Pohang University of Science and Technology (POSTECH), Korea
Antonio J. Tallón-Ballesteros, University of Seville, Spain
Steffen Thiel, Furtwangen University of Applied Sciences, Germany
Michael N. Vrahatis, University of Patras, Greece
Alexander Wijesinha, Towson University, USA
Zhengping Wu, University of Bridgeport, USA
Chao-Tung Yang, Tunghai University, Taiwan R.O.C.
Hongji Yang, De Montfort University (DMU) - Leicester, UK
Peng-Yeng Yin, National Chi Nan University, Taiwan
Zhibin Yu, Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, China
Yong-Ping Zhang, Huawei Technologies Co., Ltd., China
Zhiyong Zhang, Henan University of Science and Technology, P.R. of China

Copyright Information

Table of Contents

Eliminating the Operating System via the Bare Machine Computing Paradigm

Uzo Okafor, Ramesh K. Karne, Alexander L. Wijesinha, and Patrick Appiah-Kubi
Department of Computer and Information Sciences
Towson University
Towson, MD 21252 USA
uokafo1@students.towson.edu, {rkarne, awijesinha, appiahkubi}@towson.edu

Abstract - **Computer applications typically run under the control of intermediary system software that is in the form of an operating system such as Windows or Linux, or a small kernel. The application could also be embedded within the operating system or kernel itself. This paradigm makes applications dependent on an intermediary software layer. An alternative approach is to eliminate this layer by writing computer applications that can run directly on the hardware. This approach takes a small or tiny kernel to its extreme, eliminating the operating system, which results in a novel bare machine computing paradigm. In this paper, we describe the bare machine paradigm, and illustrate how to build self-supporting bare machine applications by eliminating application dependence on an operating system or kernel. The new paradigm requires that the developer be aware of the underlying hardware resources and use them efficiently for the needs of a given application suite. We also describe a set of generic bare interfaces that can be used across many pervasive devices as well as ordinary desktops and laptops. These interfaces have made it possible to build large bare applications. The bare machine paradigm paves the way for software interfaces to be incorporated into a chip, introducing a computing model where applications are independent of any intermediary software.**

Keywords - bare machine applications; bare machine computing; middleware; direct hardware interfaces; operating systems.

I. INTRODUCTION

Building bare machine applications, which are independent of any intermediary software, is daunting due to constraints imposed by the existing computer architecture and development environments. Most CPUs are designed to work with an operating system (OS) or kernel and do not provide any interfaces to directly control the hardware. In some cases, the kernel or virtual machine may allow an application direct hardware access, but does not fully relinquish its control to the application. However, for certain specialized applications and secure systems, even the presence of a small kernel may prevent the application from fully controlling its environment and managing the hardware.

We propose to eliminate the OS (or kernel) and give full control to applications. These applications are then able to run on the bare hardware without the need for any additional software layers. There is no persistent storage or any other resource to secure on a bare machine, device, or computing system. Moreover, only one bare application suite runs at a time. When an application is not running, the machine is not running any other code. It simply has memory, processors and an I/O controller to communicate with the applications when needed. Instead of an OS or kernel providing resources, an application suite manages the hardware. This does not mean that the applications replicate OS functionality. Rather, applications only contain code that is required for a given application suite. An application suite is modeled as an Application Object (AO) [6] that carries its own application and execution environment. For example, an AO may consist of a text processor/editor, a Webmail client, and a Web browser, and bare interfaces to the hardware. An AO programmer needs to have knowledge of the underlying resources, since an AO controls and manages all the hardware when it runs. A bare machine user carries a removable mass storage device to boot, load and run the application suite, thus making the machine bare when the AO is not loaded (since no OS or kernel is needed to run the suite).

When such bare machines are built, they become ownerless and can be used by anyone, anytime, and anywhere. Many complex bare applications have been built to illustrate the bare machine computing (BMC) paradigm. These include a Web server [4], Webmail server [1], conventional (non-HTTP) email server and client, VoIP soft-phone [9], SIP server, and bare PC clusters using split servers [13]. The development of such applications served as the motivation for designing the direct hardware interfaces to a bare PC (x86 architecture). These interfaces are generic and can be used to construct any bare machine application. One can make these interfaces and the BIOS part of the hardware in the future, thus creating a pure BMC environment, where there is no other software needed to run computer applications. A high-level methodology for developing bare machine applications was outlined previously [10]. Here, we provide details of how to develop such bare machine applications by using a set of generic hardware interfaces. In particular, this paper describes the direct hardware interfaces needed to eliminate the intermediary OS.

The rest of the paper is organized as follows. Section II provides the motivation for this work. Section III describes the bare machine computing paradigm and its characteristics. Section IV illustrates the development of bare machine applications using a step-by-step process. Section V presents the direct generic hardware interfaces. Sections VI, VII and VIII respectively cover the use of a bootable USB, memory map, and the novel features of this approach. Finally, Section IX gives the conclusion.

II. MOTIVATION

The following considerations serve as the motivation for developing BMC applications based on the underlying paradigm: the proliferation of operating systems (OS) and frequent new releases to replace them; the rapid obsolescence of existing computer applications; the myriad of programming languages and interfaces; and the heterogeneity of computer architectures and platforms. While many arguments can be given to support the current evolution of conventional systems/platforms and their advantages, the BMC paradigm has been shown to be feasible for building a variety of complex applications such as the servers, clients, and high-performance systems noted above.

Most current systems use complex concurrency control mechanisms, paging, virtual memory, and other well-known concepts that have evolved due to lack of large memory, memory costs, shared resources, and the need to serve multiple users. Under those conditions, computing evolved towards complex systems with rapid obsolescence and less security. The BMC paradigm [7] enables applications to directly communicate with hardware, thus eliminating all middleware. Using this paradigm, applications can be written in C/C++ or other languages, where an AO programmer can directly call hardware interfaces, as originally proposed in [8]. This paper extends those ideas to address general purpose development of bare machine computing applications and a set of generic interfaces to the hardware.

III. BARE MACHINE COMPUTING PARADIGM

The BMC paradigm was originally referred to as dispersed operating system computing (DOSC) [7], but we have seen further evolution of DOSC into the BMC concepts as shown in Fig 1. A conventional OS, kernel or embedded software acts as middleware between the hardware and an application. An application programmer is isolated from an application's execution environment, resource control and management. That is, the programmer has no direct control of the program's execution or the resources needed. In the BMC paradigm as shown in Fig 1, the OS is eliminated and the AO programmer is totally responsible for managing hardware resources. The AO programmer has knowledge and full control over a given application as well as its execution. Each AO only carries its needed controls and the direct hardware interfaces. The AO programmer is a domain expert for a given set of applications that are contained in a given AO. The BMC paradigm differs from conventional computing in two major ways. First, the machine is bare with no existing software and protected resources. Second, an AO programmer controls the program's execution and manages the hardware.

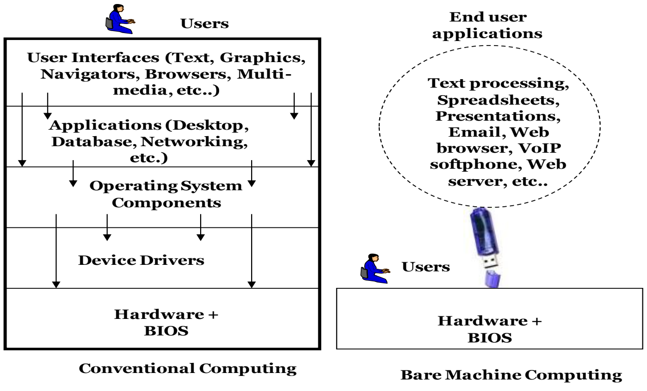

Users
User Interfaces (Text, Graphics, Navigators, Browsers, Multi-media, etc..)
Applications (Desktop, Database, Networking, etc.)
Operating System Components
Device Drivers
Hardware + BIOS
Conventional Computing
End user applications
Text processing, Spreadsheets, Presentations, Email, Web browser, VoIP softphone, Web server, etc..
Users
Hardware + BIOS
Bare Machine Computing

Figure 1. Conventional OS versus BMC paradigms

The BMC paradigm makes a computing device owner-less and simplifies the design of secure systems since there are less avenues of attack and no underlying middleware that an attacker can control. Viewed another way, when a device is bare and contains no valuable resources such as a hard disk or kernel, there is nothing to own or protect. In BMC, mass storage is external and detachable. The mass storage can also be on a network. In this approach, an AO is built for a given set of applications to run at a time on a machine as a single monolithic executable. The boot, load, executable, data and files are stored on a mass storage device such as a USB. When a USB is plugged into a computer, the machine boots and runs its own program without using any extra software or external programs. This implies that no dynamic link libraries (DLLs) or virtual machine code are allowed in this approach. What runs in the machine, is exactly what has been loaded (and nothing else).

This computing paradigm is different from conventional computing approaches since it is based on applications instead of computing environments. This is not a mini-OS or kernel, as there is no centralized program running in the machine to manage resources. Instead, the resources are managed by the applications themselves and run without using any OS/kernel or intermediary software.

A variety of attempts have been made to eliminate OS abstractions or bypass the OS. However, none eliminate the kernel or OS altogether. Thus, while the BMC paradigm resembles approaches that reduce OS overhead and/or use lean kernels such as Exokernel [2], IO-Lite [12], Palacio [11], and the Hardware Abstraction Layer [14] in Java, there are significant differences. These include self controlled applications and programmer-driven execution, and the lack of centralized code that manages system resources. A model to analyze tradeoffs between feature-rich and minimalist or "barebone" systems is presented in [15]. While such minimalist systems usually require an operating system or kernel, they may have also some characteristics in common with BMC systems.

IV. BMC APPLICATION DEVELOPMENT

In BMC, a suite of applications such as a text processor, Webmail server and Web browser can be bundled together and run without any OS or kernel support. Fig. 2 illustrates the major steps involved in developing

BMC applications. First, a choice has to be made about the suite of applications; next, the architecture of the CPU on which they will run has to be identified. Using today's CPUs, constructing a BMC application is a daunting task as they provide neither direct hardware interfaces, nor a development environment that facilitates building applications independent of an OS. For example, a bare PC requires the BIOS to boot, and an ARM processor requires a UBOOT tool. The program counter of a given processor is not directly accessible to the programmer. In a machine with an x86 CPU, the program counter can only be loaded by jumping to the task segment, where its value is stored and updated by the CPU. In a bare machine application, the program counter must be handled inside an application and not controlled by an OS or other software.

Memory needs or requirements must be considered for a given application's code, data and stack. The application programmer has to determine memory areas for the code, data and stack, as these applications run in a real memory. Real memory is cheap and affordable today. It is therefore feasible to avoid paging and virtual memory overhead, and the associated management. The absence of any other software in the system eliminates many unnecessary features commonly found in today's technology. Most BMC application suites only require small amounts of memory compared to OS-based applications. For very large applications, one can use mass storage to provide extended storage using swapping techniques. Section 7 describes details of the memory map created for some real-world applications using the BMC paradigm.

The next step is to construct an application suite using programs that are independent of any OS. This application suite should be able to run on any compatible CPU without changes or adaptations. Different CPU architectures have different compilers to compile code. This requires that I/O related code be identified and direct hardware interfaces be deployed. One of the key elements in writing BMC code is being able to differentiate between code that is OS dependent, code that is OS independent and code that is I/O related. For example, file I/O is OS dependent code and a for-loop is OS independent code. User interfaces to support keyboard, mouse and display are all I/O related code.

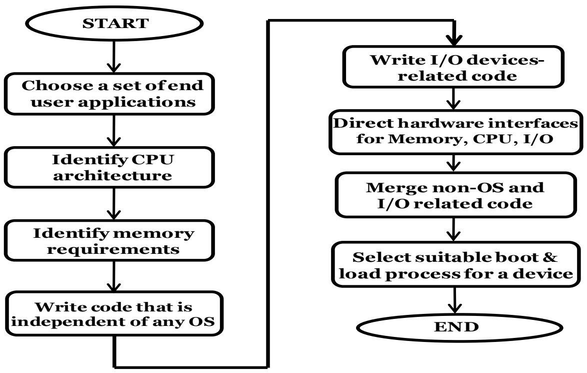

START
Choose a set of end user applications
Identify CPU architecture
Identify memory requirements
Write code that is independent of any OS
Write I/O devices-related code
Direct hardware interfaces for Memory, CPU, I/O
Merge non-OS and I/O related code
Select suitable boot & load process for a device
END

Figure 2. Steps in developing bare machine applications

Once OS dependent code and I/O related code are written (as hardware interfaces), they can all be integrated with the rest of the OS independent code and run as a single monolithic executable. The above approach introduces many challenges that must be addressed when developing BMC applications. They include the boot-up process and loading of an application suite. Each computing device is different in its boot process and the internal details are often hidden. Similarly, loading an application on a bare device also poses difficulties as it requires readily available tools that are OS dependent. Developing an OS independent loader requires a thorough knowledge of the CPU architecture and its development environment. Domain knowledge and related expertise for each CPU device are required to develop the bare boot and load processes.

V. DIRECT HARDWARE INTERFACES

Conventional computer applications and programming languages use OS calls or system calls injected at link time from an OS such as Windows [16] or Linux [3]. These calls include memory, keyboard, terminal screen, network, mass storage, and interrupts. Some OSs include in their repertoire other commonly used OS-independent functions such as memory copy, string operations and concurrency control that require system calls. Computer applications and the programmer expect these calls or interfaces to be included at compile and link time by a given compiler and linker.

Bare machine applications require system call equivalents (direct hardware interfaces) that are independent of any OS or kernel. These interfaces are directly controlled and accessed by an AO programmer. All of the above are factors to consider in determining the number of direct hardware interfaces needed for a suite of BMC applications. Some direct hardware interfaces used in BMC applications are discussed below.

A. Static and Dynamic Memory

Static memory needs depend on the size of code, data and stack needed to run a program. When an executable is created, this information is available to the programmer. Thus, for a given executable, one can specify its requirements for memory. An AO can also be designed that can read the existing memory and restructure its code, data and stack in real memory and external mass storage or network. The code image is small as there is only one AO running at a time in the machine, and applications that are related are grouped to run together.

Dynamic memory needs are however not known until run time. In a bare machine application, an AO programmer estimates the dynamic memory. Appropriate exceptions for memory can be set to manage dynamic memory; when large dynamic memory needs arise, one can use secondary storage in place of large dynamic memory. System calls similar to malloc() and free() can be designed to support dynamic memory management. One can allow the memory controller to communicate with an AO and thus provide appropriate memory interfaces to manage memory in the AO. As memory technology improves and becomes cheaper, it is also conceivable to assume full address space (4GB in a 32-bit architecture) in a machine to avoid all memory management issues and provide direct control to a given AO.

B. User Interfaces

The most common user interfaces are keyboard, mouse, touch-screen and terminal screen. These resources are managed by the OS in conventional systems. In bare machine applications, keyboard interfaces are part of an AO where the keyboard interrupt code places the data in a user buffer. Similarly, mouse data is also placed in a user buffer. An AO programmer designs the code to directly interface with a keyboard or a mouse. The terminal screen is usually controlled by a video memory or its graphics adaptor. An AO programmer can directly store output in video memory or write a bare video driver to control the screen. All device drivers supporting a bare application have to be bare and provide direct hardware interfaces to applications. They cannot, as is done, when an OS is present, be hidden from the application programmer. Other user interfaces have to be handled in a similar manner to the above interfaces.

C. Network Interfaces

Most ordinary computing devices today have one wired and one wireless network interface. The device drivers for a network interface are controlled by underlying OS. Bare machine device drivers that provide direct network interfaces to an AO are needed in BMC. Instead of current OS-dependent network drivers, an AO programmer can initialize a network driver, configure relevant internal registers, and read or write to buffers and control registers. Such a design allows direct communication to applications and avoids the need for any middleware. As the drivers are now encapsulated within an AO, the network hardware is not accessible to other applications when a given application suite is running in the machine. A bare PC USB device driver and its implementation are described in [5].

D. Process Interfaces

Many computer applications require process creation, deletion and management, which are usually controlled by an OS. In Intel x86 processors, process control and state are maintained by the CPU in a task segment. Interrupt gates are used to switch from one task to another. That can be done in a bare environment since these interfaces are accessible to an AO programmer. Control of the CPU is placed in an application program for creation of a new process (or a task). The global descriptor table (GDT) and local descriptor table (LDT) entries are used by the AO programmer to control task memory. Thus, when a machine becomes bare, the CPU and tasks are managed by an AO programmer. Task management in a bare machine is much simpler than in a conventional system, and the code size is also smaller compared to an OS-supported system. A conventional Web server system may be complex and create over 7000 tasks (in an x86 box) to provide high performance [4]. Process interfaces can eventually be generalized and made available to an AO programmer for any given CPU architecture. Today's machines hide all these interfaces under an OS or some form of similar middleware.

E. File Interfaces

In conventional systems, a file system is part of the operating system. File systems use some standard specifications such as FAT32 or NTFS. Files can be transported across multiple operating systems and applications if they use standard specifications in their design. In bare machine applications, persistent data is under the control of an AO programmer and the data itself is part of an AO. Programmers can use their own file storage specification or use a standard specification to transport files to non-bare systems. One can also do a raw file system in an AO to avoid all file management complexities and hide the files within an AO (the only AO in which they are visible). This may be the most secure way to implement a file system. File transfers can also be accomplished through a network or by message passing. A given file system interface uses a bare device driver and controls the relevant device operations.

F. Boot and Load Interfaces

Boot and load facilities are usually under control of the OS and the underlying BIOS calls. In BMC, these interfaces are controlled by the AO programmer to facilitate bare machine applications. Soft and hard boot can be used to control the machine when needed in bare machine applications. These interfaces also vary across platforms; ideally, a standard boot and load mechanism to run bare machine applications across multiple CPU architectures and machines is the best solution (what is described in Section VII, is a method that has been implemented for x86 Intel CPUs).

G. Compile, Link and Library Issues

Compilers and linkers generate different formats for executables, which pose problems in loading and running bare machine applications. There is a need for homogenization in these tools to develop common bare machine applications that can run on many pervasive devices. New programming tools can be developed to compile bare machine applications using existing libraries and batch files, or new features can be added into existing Microsoft Visual Studio and Eclipse development tools to provide bare machine compilation options. Common libraries such as string operations, memory operations, locking, shared memory, message passing, and concurrency control are system dependent and part of the OS libraries. However, they can be generalized and designed to run across many CPU architectures.

VI. BOOTABLE USB

In the BMC paradigm, applications are carried on a removable storage medium such as a CD/DVD or a flash drive. This device also carries a boot program to boot and load its own application object suite. A typical way to create a bootable USB is as follows. A bootable USB is created using a special tool written in C and assembly language. This tool is a batch file that runs in a DOS window. The USB is formatted for FAT32 before its use. The bootable USB should have three files as shown in Fig. 3b. The boot file is stored in the boot sector (**#0**), the prcycle.exe file is stored at **0x3be000**, and the application file (shown in Fig. 3a as shell.exe), is stored at **0x3c4000**.

The prcycle.exe file (**22, 037** bytes in size) contains assembly code to boot a bare PC, provides the user interface/menu, and facilitates the loading of AOs (in this instance, shell.exe). It enables the switching from real to protected mode and vice versa for handling low-level interfaces. It also contains, IDT, GDT, TSS and BIOS interrupts to provide the AO programmer with direct control of the CPU. This part of the application code thus plays a key role in enabling the programmer to manage the hardware resources in a bare PC. In summary, the batch file copies files onto the USB, installs a boot program, and creates a bootable USB. This entire process does not require any software other than what resides on the USB (and is thus part of the bare PC application). There is no dependence on any specialized commercial tool or software. This enables bare PC applications to be independent of any OS-related environments and tools. It is also possible to use existing boot tools to create a bootable USB; however those tools must guarantee high security if needed in a system. The approach proposed here demonstrates building bare machine computer applications in a single environment where every aspect of software development is controlled by an AO programmer with no other dependencies. This approach facilitates enhanced security to computer applications.

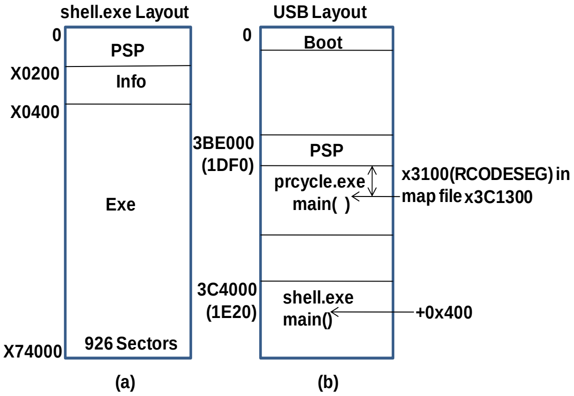


Figure 3. USB layout

VII. MEMORY MAP

As discussed in section 4, the AO programmer needs to design the real memory layout when developing a bare PC application. Fig. 4 shows a typical memory layout for a given application suite. An AO programmer prepares this map before designing a given application suite. The prcycle.exe program is used on the bare platform to load the AO at **0x600** in real mode memory. The main() entry point for prcycle.exe is located at **0x3100**, which can be obtained from the prcycle.map. When the PC is booted, it must jump to **0x3900** as instructed by this memory map. A user loads the example application (shell.exe) by using the menu provided by prcycle.exe (not shown here). The executable for this AO is loaded at **0x00111E00** as shown in Fig. 4. The reason for using this particular address for loading shell.exe is discussed below. Visual Studio 8.0 (and later editions) of compilers behave differently than the previous versions when generating an exe file. In previous versions, when the entry point in shell.map indicates **0001:00000000**, it usually implies that the main entry point in shell.exe is at **0x1000**. In newer versions, this is not the case. In Visual Studio 8.0 (C++ versions), the executable starts at address **0x400** instead of at **0x1000**. As shown in Fig. 4, the AO (shell.exe) is located at **0x00111E00**. The higher 16-bit address **0x0011** indicates that it is loaded above **1 MB** to load it in a protected mode memory address.

The lower 16-bit address **0x1E00** is derived as follows. The compiler start address for shell.exe is **0x0000**, but it actually starts at **0x400**. It was observed in the executable that the offset used by this compiler is **0x1e00** more than the actual offset in the executable. Thus, when the executable is relocated at **0x1e00**, the references to the variables were correct as it was generated by the compiler. The main entry point for shell.exe should be at **0x1e00** + **0x400** as shown in Fig. 4. A generic tool is needed to resolve such intricacies involved in generating a memory map for a mass storage device. This tool should consider compiler options, executable formats and map files to create a memory map that is suitable for a given bare machine device.

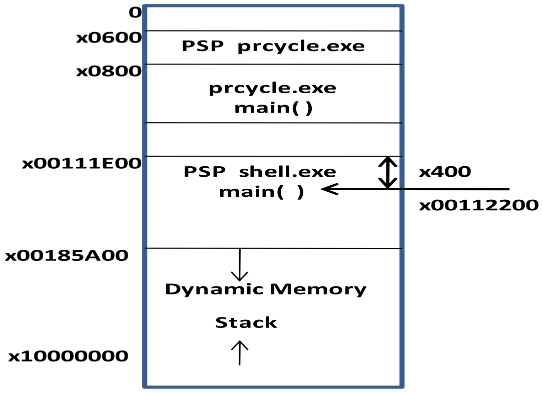


Figure 4. Memory map

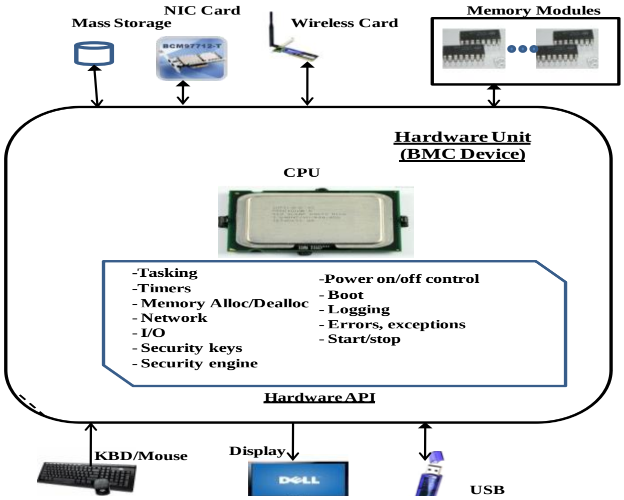


Figure 5. BMC device architecture

VIII. NOVEL FEATURES

BMC applications provide a new and innovative computer architecture that is based on current trends in technology. Fig. 5 illustrates a vision of future computing. This shows a BMC device that communicates with standard units such as memory, network card, wireless card, USB device, keyboard, mouse, display and mass storage. These units are common to many pervasive devices today. It is thus useful to write computer applications that target the BMC device as a baseline. Each device can run its own native application while using the standard hardware API as illustrated in the figure. All applications can access these interfaces and yet the hardware itself is bare. Until then, we can continue to provide these interfaces as software. The BMC architecture avoids heterogeneity in hardware, software, programming and tools.

IX. CONCLUSION

We described the BMC paradigm and showed how to build applications based on it. We identified the generic direct bare hardware interfaces needed to eliminate the OS/kernel. The BMC paradigm/approach enables these hardware interfaces to be incorporated in the hardware, thus making the latter more intelligent and able to communicate with the software. The interfaces were used to construct complex bare PC applications that have a small code footprint, are simple to use, provide high performance, and are inherently secure in design. We also presented a bare machine application architecture that enables a BMC device to be used for many pervasive applications. The new paradigm and approach will make it possible to save time, energy, and resources, while reducing the cost of developing applications for each pervasive device. The BMC paradigm demonstrates a new approach to computing based on completely self-supporting applications that eliminate all intermediary software.

REFERENCES

[1] P. Appiah-Kubi, R. K. Karne and A.L. Wijesinha, “The Design and Performance of a Bare PC Webmail Server,” The 12th IEEE International Conference on High Performance Computing and Communications, AHPCC, pp. 521-526, Sept. 2010.

[2] D. Engler, “The Exokernel Operating Systems Architecture,” Dept. of Elec. Eng. and Computer Science, Massachusetts Institute of Technology, Ph.D. Dissertation, 1998.

[3] FreeBSD/Linux Kernel Cross Reference, http://fxr.watson.org/fxr/source/kern/syscalls.c.

[4] L. He, R. K. Karne, and A. L. Wijesinha, “Design and Performance of a bare PC Web Server,” International Journal of Computer and Applications, vol. 15, pp. 100-112, June 2008.

[5] R. K. Karne, S. Liang, A. L. Wijesinha and P. Appiah-Kubi, “Bare PC Mass Storage USB Driver,” International Journal of Computer and Applications, March 2013.

[6] R. K. Karne, “Application-oriented Object Architecture: A Revolutionary Approach,” 6th International Conference, HPC Asia, Dec. 2002.

[7] R. K. Karne, V. Jaganathan, T. Ahmed and N. Rosa, “DOSC: Dispersed Operating System Computing,” OOPSLA, Onward Track, 20th Annual ACM Conference on Object Oriented Programming, Oct. 2005.

[8] R. K. Karne, V. Jaganathan and T. Ahmed, “How to run C++ Applications on a bare PC,” 6th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing (SNPD), pp. 50-55, May 2005.

[9] G. H. Khaksari, A. L., Wijesinha, R. K., Karne, L., He and S. Girumala, “A Peer-to-Peer Bare PC VoIP Application,” IEEE Consumer and Communications and Networking Conference (CCNC), pp. 803-807, Jan. 2007.

[10] G. H. Khaksari, A. L. Wijesinha, and R. K. Karne, “A Bare Machine Development Methodology,” International Journal of Computer Applications, vol. 19, no.1, pp. 10-25, Mar. 2012.

[11] J. Lange. et. al, “Palacios and Kitten: New High Performance Operating Systems for Scalable Virtualized and Native Supercomputing,” 24th IEEE International Parallel and Distributed Processing Symposium , Apr. 2010.

[12] V. S. Pai, P. Druschel and Zwaenepoel, “IO-Lite: A Unified I/O Buffering and Caching System,” ACM Transactions on Computer Systems, vol.18 (1), pp. 37-66, Feb. 2000.

[13] B. Rawal, R. K.Karne, A. L. Wijesinha, “Mini Web Server Clusters for HTTP Request Splitting,” IEEE International Conference on High Performance Computing and Communications, pp. 94-100, Sep. 2011.

[14] M. Schoeberl, S. Korsholm, T. Kalibera and A. P. Ravn, “A Hardware Abstraction Layer in Java,” ACM Transactions on Embedded Computing Systems, vol.10, no. 4, Article 42, Nov. 2011.

[15] S. Soumya, R. Guerin and K. Hosanagar, “Functionality-rich vs Minimalist Platforms: A Two-sided Market Analysis”, ACM Computer Communication Review, vol. 41, no. 5, pp. 36-43, Sep. 2011.

[16] Windows System Call Table, Googlecode.com, retrieved Feb 16, 2012. http://miscellaneuoz.google.com/svn/winsyscall.

An Orbit Tracking Algorithm in Quantum Systems

Shuang Cong
Dept. Automation
University of Science and Technology of China
Hefei, China
e-mail: scong@ustc.edu.cn

Jianxiu Liu
Dept. Automation
University of Science and Technology of China
Hefei, China
e-mail: zjxliu@mail.ustc.edu.cn

Abstract—**A convergence algorithm for the orbit tracking of the free-evolutionary target system in closed quantum system is studied in this paper. The unitary transformation is used to change the problem of orbit tracking into the one of state transferring. The Lyapunov function with a virtual mechanical quantity *P* is employed to design a Lyapunov–based controller for such a state transferring. The target states are divided into two classes: diagonal and non-diagonal. In the first class, the specific convergent conditions for the target state of diagonal states are studied; in the second class, the target states with non-diagonal superposition states and non-diagonal mixed-states are treated separately by two quite different ways. At last, the system simulation experiments are performed on a two-level quantum system and the tracking process is illustrated on the Bloch sphere.**

Keywords-orbit tracking; state transferring; the Lyapunov stability theorem; convergence

I. INTRODUCTION

In recent years, the closed quantum system control theory has been increasingly developed. Quantum state transferring and quantum system tracking have been fully researched. The effective control algorithm has been designed based on certain control theory, such as optimal control [1-3], Bang-bang control [4, 5], the Lyapunov-based control [6-9] and so on. Among them, optimal control produces iterative control algorithm, while bang-bang control is realized experimentally by pulses. Further demand for accuracy may results in the unacceptable number of pulses. The Lyapunov stability theorem may obtain an analytical algorithm, which can help us analyze the characteristics of the system.

In this paper, the quantum system orbit tracking is investigated. For the orbit tracking, the target system is the free-evolutionary of the quantum system to be controlled. What we expect is to make the controlled system track the target system. The control goal is carried on in two steps: firstly, the system orbit tracking problem is changed into the state transferring one; secondly, a convergent control algorithm is designed to complete the goal.

In fact, the control algorithm designed by the Lyapunov stability theorem is only stable, which cannot guarantee the quantum system converges to desired target state. For this purpose, a convergent rather than just stable control algorithm is needed to manipulate the quantum system. Some papers on this topic have been reported [9, 10]. Among them, complete state transferring with target state of eigenstates and diagonal mixed-states in closed quantum system has been proved to be convergent conditionally [11, 12]. However, there is an open problem on the convergence of the non-diagonal target states including the superposition states and some mixed-states. In [12], a Lyapunov function based on virtual mechanical quantity *P* was proposed to get the convergence conditions for the diagonal mixed-states. However, it did not give specific instructions on how to design *P*. We will discuss the detailed convergence conditions of diagonal target states and non-diagonal target states in this paper.

The rest of the paper is organized as follows. In Section II, the system model is described by the Liouville equation. The Lyapunov stability theorem is used to design the trajectory tracking control algorithm in Section III. Section IV is divided into two parts, the first part is to handle the convergence of initial target state with diagonal mixes-state and the second part is the one for non-diagonal initial target state. In Section V, numerical simulation experiments are performed on a two-level system. Finally, Section VI concludes this paper.

II. DESCRIPTION OF THE CONTROL SYSTEM MODEL

In this paper, we use quantum-Liouville equation to describe the control system model

$$i\hbar\frac{\partial}{\partial t}\hat{\rho}(t)=\left[H_0+\sum_m f_m(t)H_m,\hat{\rho}(t)\right]\quad\hat{\rho}(0)=\hat{\rho}_0 \tag{1a}$$

$$i\hbar\frac{\partial}{\partial t}\hat{\rho}_f(t)=\left[H_0,\hat{\rho}_f(t)\right]\quad\hat{\rho}_f(0)=\hat{\rho}_{f0} \tag{1b}$$

where H_0 is the free Hamiltonian representing the energy of the system and H_m represent system's control Hamiltonians, all of them will be assumed to be time-independent. $f_m(t)$ are time-dependent external control fields. The Planck constant is chosen as $\hbar=1$ for convenience. The system state is denoted by density matrix $\hat{\rho}(t)$, the initial states of which at $t=0$ is $\hat{\rho}_0$. Similarly, $\hat{\rho}_f(t)$ is the target system state and its initial value is $\hat{\rho}_{f0}$.

The control objective is to design a convergent control algorithm to make the state $\hat{\rho}(t)$ of (1a) follow target state $\hat{\rho}_f(t)$ of system (1b).

Firstly, to change the orbit tracking problem into a state transferring one, a unitary transformation $U(t)=\exp(-itH_0)$ is performed on system (1): $\rho(t)=U^{\dagger}(t)\hat{\rho}(t)U(t)$, $\rho_f(t)=U^{\dagger}(t)\hat{\rho}_f(t)U(t)$, where "†" denotes conjugate transpose, "^" denotes states before unitary transformation. The system (1) after this transformation is represented by

$$i\frac{\partial}{\partial t}\rho(t)=[\sum_m f_m(t)H_m(t),\rho(t)] \quad \rho(0)=\hat{\rho}_0 \tag{2a}$$

$$i\frac{\partial}{\partial t}\rho_f(t)=0 \qquad \rho_f(0)=\hat{\rho}_{f0} \tag{2b}$$

where $H_m(t)=U^{\dagger}(t)H_mU(t)$.

Calculate (2b), one gets

$$\rho_f(t)=\hat{\rho}_{f0}\,. \tag{3}$$

After the above transformation, we have changed the tracking problem into the state transferring one. At present, the control goal of system (1a) following target system (1b) becomes the one of regulating state of (2a) to target state (3).

III. Control Algorithm Derivation

Among many control methods, the Lyapunov-based method is simpler and easy to design. The basic idea of the Lyapunov method is to select a Lyapunov function $V(x)$ which satisfies the following three conditions: a) $V(x)$ is continuous and its first-order partial derivatives is also continuous in its definition; b) $V(x)$ is positive semi-definite, i.e., $V(x)\geq 0$; c) The first order time derivative of the Lyapunov function is negative semi-definite, i.e., $\dot{V}(x)\leq 0$.

There are usually three kinds of Lyapunov functions [11]. Here, the Lyapunov function based on virtual physical quantity is chosen

$$V(\rho)=tr(P\hat{\rho}), \tag{4}$$

where P is virtual physical observable operator.

To obtain a convergent control algorithm, the first-order time derivation of function (4) is obtained as

$$\dot{V}=-\sum_m f_m(t)tr\left(iH_m(t)[\rho(t),P]\right). \tag{5}$$

For the sake of simplicity and availability, we let each item on the right side of (5) of summation sign be non-positive in order to ensure $\dot{V}\leq 0$. The control algorithm can be derived as

$$f_m(t)=-k_m tr\left(iH_m(t)\left([\rho(t),P]\right)\right),\ k_m>0 \tag{6}$$

where, $k_m>0$ is the control gain to adjust the convergence speed of the system state.

Comparing system (1) with (2a) and (3), one can see that: the system (1) is autonomous and the system (2) is not, however, the time-dependent target state $\rho_f(t)$ in (1b) becomes a stationary state $\hat{\rho}_{f0}$ in (3).

As we know, the control algorithm (6) designed by the Lyapunov stability theorem is usually only a stable one, which can not guarantee that the system converges to target state. For this reason, we need to do further study to get the convergence condition $\dot{V}<0$, which may guide people to design a convergent control algorithm. Next, we study this problem in detail.

IV. Control Algorithm Convergence Analysis

In order to implement effective control on a quantum system, a convergent control algorithm is even more important than the control algorithm itself. In the process of state transferring, the desired target is a time-invariant state. The variety of quantum states such as eigenstates, superposition states and mixed-states makes the different expressions of target state. On the one hand, from the system control perspective, the initial state of target system will be grouped into stationary state and the non-stationary one. For the stationary target state, according to (1b), $i\hbar\frac{\partial}{\partial t}\hat{\rho}_f(t)=0$ holds, and so does $\hat{\rho}_f(t)=\hat{\rho}_f(0)$. In such a case, one can get $\left[H_0,\hat{\rho}_f(0)\right]=0$. The system tracking of (1a) is equivalent to the state transferring. For the non-stationary initial target state, the unitary transformation has changed the tracking problem into the state transferring one. Therefore, we can always change the system tracking into the state transferring. On the other hand, from the convergence of view, the initial target states are divided into two kinds. The first one is diagonal initial target state including eigenstates and some mixed-states. It can be represented by a diagonal density matrix. The second kind is non-diagonal initial target state, including superposition state and some mixed-states, whose convergence has not been resolved fully so far.

For autonomous system (1), the LaSalle's invariant principle can be used to analyze the convergence, where two assumptions are needed [13]: Assumption 1: H_0 is strongly regular, i.e., all the transition frequencies (differences of pairs of energy levels) are different, viz. $\Delta_{jk}\neq\Delta_{pq}$, $(j,k)\neq(p,q)$, where $\Delta_{jk}=\lambda_j-\lambda_k$ and λ_j is an eigenvalues of H_0. Assumption 2: Control Hamiltonian H_m is full-connected, viz. $H_m \in \left\{\hbar h_{jk} \mid h_{jk} = |j\rangle\langle k| + |k\rangle\langle j|, j > k\right\}$, where $|j\rangle$ is the eigenstate associated with λ_j. Because of the unitary evolution of closed quantum system, if being reachable then target state must be unitarily equivalent to the initial state, i.e., there exists a unitary transformation U such that $\hat{\rho}_0 = U\hat{\rho}_{f0}U^{\dagger}$. We make it as Assumption 3.

The LaSalle's invariant principle is not able to deal with the non-autonomous system (2), however, the Barbalat lemma can be applied based on above three assumptions, whose content is [14]: If scalar function $V(x,t)$ satisfies: (1) $V(x,t)$ is lower bounded; (2) $\dot{V}(x,t)$ is negative semi-definite; (3) $\dot{V}(x,t)$ is uniformly continuous in time, then $\dot{V}(x,t) \to 0$ as $t \to \infty$. One can see from the Lyapunov function (4) that (4) satisfies all the three conditions of Barbalat lemma: (1) $V = tr(P\rho) \geq 0$ is lower bounded for a positive P; (2) Its first order derivative is negative semi-definite under control algorithm (6); (3) The third condition can be replaced by the existence and continuity of the second derivation of $V(x,t)$: $\ddot{V}(\rho,t) = -\sum_m f_m(t)\{tr(i\dot{H}_m(t)[\rho,P]) + tr(iH_m(t)[\dot{\rho},P])\}$ is bounded for a bounded input.

According to the Barbalat Lemma, the first derivation of the Lyapunov function converges to zero for $t \to \infty$, viz., $\dot{V}(\rho(\infty),\infty) = 0$. So a limitation states set at $t \to \infty$ is deduced by the Barbalat Lemma. We defined it as a stable set $\mathcal{R}$, which is a concept similar to invariant set. According to the formula (6), the states in stable set satisfy $f = 0$. For the non-autonomous system (2), if $\rho \in \mathcal{R}$, then $\dot{\rho}$=0 holds for $f = 0$. It means that, once the system evolves into the stable set, it will stop at this set.

The stable set $\mathcal{R}$ is the set of critical points on any dynamic trajectory, viz.

$$\mathcal{R} \equiv \left\{\rho_s : tr\left(iH_m(t)\left[\rho_s, P\right]\right) = 0, \forall m, t\right\}, \qquad (7)$$

where ρ_s denote critical stable points of (2), and $\rho_f \in \mathcal{R}$. The controlled system may converge to any one of states in stable set $\mathcal{R}$. Now P is to be constructed to make the system converge to the target state. According to Assumption 2, (7) is rewritten as $\mathcal{R} \equiv \left\{\rho_s : \left[\rho_s, P\right] = D\right\}$, where D is a diagonal matrix. Obviously, if P is chosen as a diagonal matrix, D is zero one, otherwise we can always design a set of proper eigenvalues of P to simplify (7) as [15]:

$$\mathcal{R} \equiv \left\{\rho_s : [\rho_s, P] = 0\right\}. \qquad (8)$$

Equation (8) is the stable set to be discussed in this paper. It is known that (4) is a function of state and the system (2) will converge to stable set (8). Whether the system converges to target state or not depends on the relative position among target state, the controlled initial state, and all other stable states than target state. To make the system converge to the target state, the following condition is needed, viz. the relationship among initial state, target state, and other stable states must satisfy [12]

$$v\left(\rho_f\right) < v\left(\rho_0\right) < v\left(\rho_s\right). \qquad (9)$$

Equation (9) is the condition to ensure the convergence of the controlled system. How to realize (9) is another key point which needs to be solved. In the following we'll focus on this topic by designing the suitable *P*. The solutions are given by two cases.

A. *The target state is diagonal mixed-state*

The convergent conditions for diagonal target states have been investigated adequately [11-13]. However, the concrete construction of *P* has not been mentioned in previous results. In this section, we'll discuss how to design *P* in (4) to satisfy (9).

Suppose the target state ρ_f is a diagonal target state and $\left\{\lambda_i, i = 1, 2 \cdots n\right\}$ is the eigen-spectrum of ρ_0. The target state ρ_f should be one permutation of $\left\{\lambda_i, i = 1, 2 \cdots n\right\}$, viz. $\rho_f = diag(\lambda_1, \lambda_2 \cdots \lambda_n)$. The other states ρ_s in $\mathcal{R}$ are the different permutations of eigen-spectrum. In order to construct a *P* satisfied (9), three steps are performed:

Firstly, P is constructed to make ρ_f be the point corresponding to the minimum value of (4), which is realized by the following lemma:

Lemma 1: If the diagonal target state is $\rho_f = diag(\lambda_1, \lambda_2 \cdots \lambda_n)$, the matrix P corresponding to ρ_f is $P = diag\left(p_1, p_2, \cdots p_n\right)$, then ρ_f is the point for the Lyapunov function (4) to be the minimum if the diagonal element p_i of P meets $\left(\lambda_i - \lambda_j\right)\left(p_i - p_j\right) < 0, \forall i \neq j$.

The proof of Lemma 1 is in Appendix 1.

Secondly, based on Lemma 1, a further study on *P* is carried out and (9) is divided into two parts:

Part 1: $v\left(\rho_f\right) < v\left(\rho_0\right)$

The condition $v\left(\rho_f\right) < v\left(\rho_0\right)$ indicates that the Lyapunov function value of initial state is larger than that of target state. Otherwise it is inconsistent with the monotonically decreasing of (4) and the target state will be unreachable. It is easy to obtain

$v(\rho_f)-v(\rho_0)=\sum_{i=1}^{n}(P)_{ii}\left(\lambda_i-(\rho_0)_{ii}\right)$. Based on the relationship between eigenvalues and matrix diagonal elements, the expression $\sum_{i=1}^{n}\lambda_i=\sum_{i=1}^{n}\mu_i=\sum_{i=1}^{n}(\rho_0)_{ii}=1$ holds, where $(\rho_0)_{ii}$ is the *i*-th diagonal element of the initial state ρ_0 . So there must be at least one k to make $\lambda_k<(\rho_0)_{kk}$ hold. If one wants to make $v(\rho_f)-v(\rho_0)=(P)_{kk}\left(\lambda_k-(\rho_0)_{kk}\right)+\sum_{i=1,i\neq k}^{n}(P)_{ii}\left(\lambda_i-(\rho_0)_{ii}\right)<0$ hold, where $(P)_{kk}$ is the *k*-th diagonal element of *P*, then we choose a certain k satisfied $\lambda_k<(\rho_0)_{kk}$, one gets

$$(P)_{kk}>\sum_{i=1,i\neq k}^{n}(P)_{ii}\left(\lambda_i-(\rho_0)_{ii}\right)\Big/\left((\rho_0)_{kk}-\lambda_k\right).\quad(10)$$

Maybe there are more than one k to satisfy $\lambda_k<(\rho_0)_{kk}$, we usually choose the one which makes $(P)_{kk}$ correspond to a larger value such that $v(\rho_f)<v(\rho_0)$ holds.

Part 2: $v(\rho_0)<v(\rho_s)$

ρ_s should be one of the permutations of eigen-spectrum. According to Assumption 3, $\rho_0=U\rho_f U^\dagger$, so one has

$$tr(P\rho_0)=tr(PU\rho_f U^\dagger)=\sum_{i=1}^{n}(P)_{ii}\sum_{j=1}^{n}(\rho_f)_{jj}\left(U_{ij}\right)^2$$

$$tr(P\rho_s)=\sum_{i=1}^{n}(P)_{ii}(\rho_s)_{ii}$$

$$tr(P\rho_0)-tr(P\rho_s)=\sum_{i=1}^{n}(P)_{ii}\left(\sum_{j=1}^{n}(\rho_f)_{jj}\left(U_{ij}\right)^2-(\rho_s)_{ii}\right)$$

ρ_s and ρ_f have the same spectrums, there must be $(\rho_f)_{kk}=(\rho_s)_{ii}$. For any unitary matrix U , there exist $UU^\dagger=U^\dagger U=I$ and $\sum_{j=1}^{n}\left(U_{ij}\right)^2=1$. So one can get

$$tr(P\rho_0)-tr(P\rho_s)=\sum_{i=1}^{n}(P)_{ii}\left(\sum_{j\neq k}^{n}(\rho_f)_{jj}\left(U_{ij}\right)^2-(\rho_f)_{kk}\sum_{j\neq k}^{n}\left(U_{ij}\right)^2\right)$$
$$=\sum_{i=1}^{n}(P)_{ii}\sum_{j\neq k}^{n}\left((\rho_f)_{jj}-(\rho_f)_{kk}\right)\left(U_{ij}\right)^2$$

For the above equation, there is at least one l to make $(\rho_f)_{ll}-(\rho_f)_{kk}<0$ hold. To make $tr(P\rho_0)-tr(P\rho_s)<0$, the following expression is needed

$$(P)_{ll}>(\sum_{i\neq l}^{n}(P)_{ii}\sum_{j\neq k}^{n}\left((\rho_f)_{jj}-(\rho_f)_{kk}\right)\left(U_{ij}\right)^2+(P)_{ll}\bullet$$
$$\sum_{j\neq k,j\neq l}^{n}\left((\rho_f)_{jj}-(\rho_f)_{kk}\right)\left(U_{lj}\right)^2)/\left((\rho_f)_{kk}-(\rho_f)_{ll}\right)\quad(11)$$

The above process is to construct *P* for diagonal target states. We conclude that: if the target state is of diagonal, a Hermite and positive diagonal matrix *P* is selected. To ensure the convergence, the diagonal elements of *P* must satisfy Lemma 1, (10) and (11) simultaneously.

B. *The target state is of non-diagonal density matrix*

It is more complicated to analyze the convergence for the non-diagonal target state. The idea is as follows: let the non-diagonal target state be changed into diagonal one and the virtual physical quantity *P* is designed as that in *A*. However, the superposition state is one kind of pure states, which can be represented by wave functions as $\rho_f=\left|\psi_f\right\rangle\left\langle\psi_f\right|$. In this case, the diagonalization of target state is not necessary. Next, we go onto the analysis of non-diagonal superposition state and mixed-state in detail.

1) *In the case of non-diagonal superposition state*

Prior to analysis, another lemma is introduced.

Lemma 2 [16]**:** For the *n*-level Hermite matrix *A* and *B*, if they are commutive, viz. [*A, B*] =0, then *A* and *B* own the same eigenstates. We rewrite *P* according to its eigen-decomposition as $P=\sum_k p_k\left|\psi_k\right\rangle\left\langle\psi_k\right|$, where $\left|\psi_k\right\rangle$ is its eigenstate and p_k is eigenvalue. According to $\rho_f\in\mathcal{R}$ and **Lemma 2**: *P* is demonstrated as

$$P=p_1\left|\psi_f\right\rangle\left\langle\psi_f\right|+\sum_{k=2}^{n}p_k\left|\psi_k\right\rangle\left\langle\psi_k\right|,\text{ where }\left|\psi_1\right\rangle=\left|\psi_f\right\rangle.\text{ And}$$
$$\left\langle\psi_i\middle|\psi_j\right\rangle=0,\text{ for }i\neq j.\quad(12)$$

And ρ_s should be

$$\rho_s=\lambda_1\left|\psi_f\right\rangle\left\langle\psi_f\right|+\sum_{k=2}^{n}\lambda_k\left|\psi_k\right\rangle\left\langle\psi_k\right|\quad\sum_{k=1}^{n}\lambda_k=1\quad(13)$$

It is known that the states ρ_0 and ρ_s have the same spectrum under the unitary evolution, and therefore ρ_s has the same eigenvalues with ρ_0 , so does the target state ρ_f .

Substituting $\rho_f=\left|\psi_f\right\rangle\left\langle\psi_f\right|$ into (13), the eigen-spectrum of ρ_f is $\{1,0,\cdots0\}$. Then for ρ_s , there is only one eigenvalue λ_i to be non-zero, viz. $\rho_s=\lambda_i\left|\psi_i\right\rangle\left\langle\psi_i\right|$ $(\lambda_i=1)$.

If it denotes $\rho_j=\left|\psi_j\right\rangle\left\langle\psi_j\right|$ in (12), then

$$\begin{cases} v\left(\rho_f\right)=tr(P\rho_f)=p_1 \\ v\left(\rho_0\right)=p_1 tr\left(\rho_f\rho_0\right)+\sum_{k=2}^{n} p_k tr\left(\rho_k\rho_0\right) \\ v\left(\rho_s\right)=p_j \quad (j\neq 1) \end{cases} \tag{14}$$

Combined (14) with (9), a suitable P must be constructed to satisfy

$$0<p_1<p_1 tr\left(\rho_f\rho_0\right)+\sum_{k=2}^{n} p_k tr\left(\rho_k\rho_0\right)<p_j\left(j\neq 1\right) \tag{15}$$

It can be seen from (15) that the virtual mechanical quantity P may not be a diagonal matrix for non-diagonal superposition target state. The P constructed based on (12) and (15) can guarantee the convergence of non-diagonal superposition target state, where (12) describes how to construct the eigenstates and (15) is to determine the eigenvalues. Moreover, the eigenvalue p_1 of P, whose corresponding eigenstate is the target state, is the smallest one.

2) In the case of non-diagonal mixed-state

For system (1), suppose the initial target state $\hat{\rho}_{f0}$ is a non-diagonal mixed-state. The solution of (1b) is $\hat{\rho}_f\left(t\right)=e^{-iH_0t}\hat{\rho}_{f0}e^{iH_0t}$. To deal with this situation, a unitary transformation has transformed the tracking problem of (1) into the state transferring one of (2). We follow the idea of changing the non-diagonal $\hat{\rho}_{f0}$ into a diagonal one by another unitary transformation and then a convergent control algorithm can be designed based on *A*.

In system (2), the target state $\hat{\rho}_{f0}$ is a Hermite matrix, so it exists another unitary transformation U_f to meet $U_f\hat{\rho}_{f0}U_f^{\dagger}=D_f$. It is performed on system (2), viz., $\rho'=U_f\rho U_f^{\dagger}$, $\rho'_f=U_f\rho_f U_f^{\dagger}=D_f$, then the system (2) becomes

$$i\hbar\frac{\partial}{\partial t}\rho'\left(t\right)=\left[H_0+\sum_{m=1}^{M} f_m\left(t\right)H_{mt},\rho'\left(t\right)\right] \quad \rho'\left(0\right)=U_f\hat{\rho}_0U_f^{\dagger} \tag{16a}$$

$$i\hbar\frac{\partial}{\partial t}\rho'_f\left(t\right)=0 \qquad \rho'_f\left(0\right)=D_f \tag{16b}$$

where $H_{mt}=U_fH_mU_f^{\dagger}$.

After unitary transformation U_f, the tracking of target system with non-diagonal initial state $\hat{\rho}_{f0}$ in (2) can be changed into the tracking of a diagonal stationary state in (16). According to (8), the stable set of (16) is still $\mathcal{R}\equiv\left\{\rho_s:[\rho_s,P]=0\right\}$. The convergence analysis is the same as that in *A*.

In conclusion, we have acquired the convergence conditions for non-diagonal target states.

V. Applications and Experimental Results Analyses

In this part, a two level atom system controlled by a single control field is considered. Take superposition target state for example, the effectiveness of the proposed method will be illustrated.

The free Hamiltonian of the controlled system (1) is $H_0=\omega\sigma_z$ and the control Hamiltonian is $H_1=\sigma_x$, where $\sigma_i(i=x,y,z)$ denotes Pauli matrix and $\sigma_x=\left[0\ \ 1;1\ \ 0\right]$, $\sigma_z=\left[1\ \ 0;0\ \ -1\right]$. Obviously, this example satisfies the three conditions in Section IV.

The initial state of (1a) is $\left|\psi_0\right\rangle=\frac{1}{\sqrt{3}}\left|0\right\rangle+\frac{\sqrt{2}}{\sqrt{3}}\left|1\right\rangle$ and the initial target state of (1b) is $\left|\psi_f\right\rangle=\frac{1}{\sqrt{8}}\left|0\right\rangle+\frac{\sqrt{7}}{\sqrt{8}}\left|1\right\rangle$. They are both non-diagonal superposition states. The design process of a convergent control algorithm is as follows:

1) Construct P

To construct *P*, a set of linearly independent vector $\left|\psi_k\right\rangle\left(k=1,2\right)$ is prepared. In this example, we choose $\left|\psi_1\right\rangle=\left|\psi_f\right\rangle$, $\left|\psi_2\right\rangle=e_1$. Then the Schmidt orthogonalization is performed. Suppose the orthogonalized vectors are $\left|s_1\right\rangle$ and $\left|s_2\right\rangle$, where $\left|s_1\right\rangle=\left|\psi_f\right\rangle$. According to (12), $P=p_1\left|s_1\right\rangle\left\langle s_1\right|+p_2\left|s_2\right\rangle\left\langle s_2\right|$ holds. The state except target state in $\mathcal{R}$ is $\rho_s=\left|s_2\right\rangle\left\langle s_2\right|$. Here, we choose $p_1=0.2, p_2=2$, then $P=\left[1.775\ \ \text{-0.595;-0.595}\ \ 0.425\right]$.

2) System Simulation results

The control gain in (6) is selected as $k=0.1$. The simulation results are showed in Fig. 1, where the red circle denotes the controlled initial state and the blue circle is the target state; the red line is the controlled trajectory and the arrow indicates its direction. Fig. 1(a) shows the state transferring process during $t\in[0,50]$. Fig. 1(b) is the control field.

To illustrate better the control strategy, the control field Fig. 1(b) is applied to the origianl system (1). The tracking results are showed in Fig. 2, where the red dashed line is the evolution curve of controlled state in (1a) and the blue solid line is the one of target state in target system (1b); the red circle and the blue circle indicate the initial location at the current period of the controlled state and the target state respectively; the arrow indicates the direction. In Fig. 2(a), the evolution trajectory at $t\in[0,8]$ is showed, from which one can see that the controlled system is asymptotically stable with respect to the target system on the Bloch sphere. Fig. 2(b) is the state trajectory at $t\in[8,30]$ and Fig. 2(c) is the magnified bottom view of Fig. 2 (b). We have specially labeled the different locations with black box. It can be seen

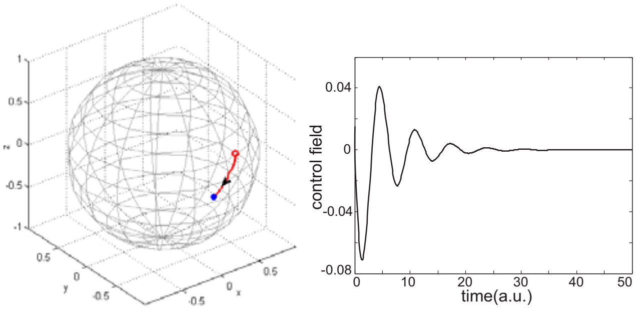
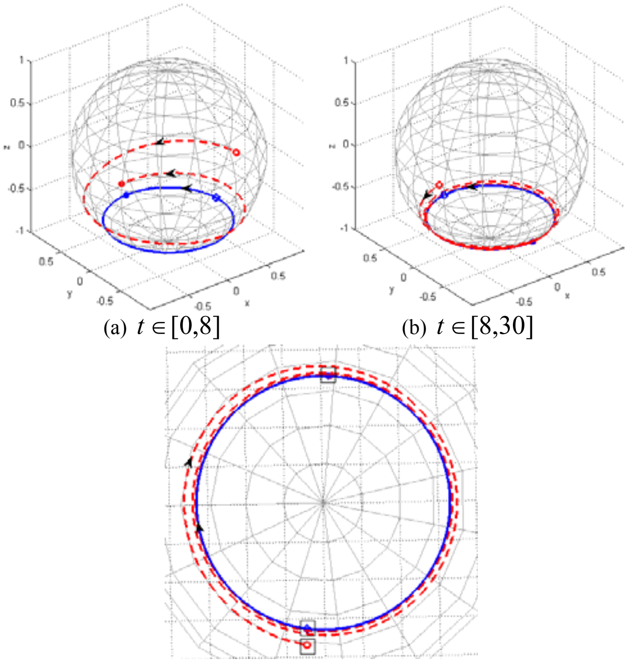


Figure 1(a). State evolution of (2)

Figure 1(b). Control field



(a) $t\in[0,8]$

(b) $t\in[8,30]$

(c) The bottom view of magnified (b)

Figure 2. The state trajectory tracking process of non-diagonal superposition target state

from Fig. 2(c) that the red circle overlapped the blue one at t=30 (the top box), so the tracking is completed at the moment. Since then, the controlled system would follow the target state in the target orbit. All the three figures demonstrated completely how the system (1a) to track the system (1b). If the index performance $v=\left\|\hat\rho(t)-\hat\rho_f(t)\right\|^2 = tr\left(\left(\hat\rho-\hat\rho_f\right)^\dagger\left(\hat\rho-\hat\rho_f\right)\right)$ is used to measure the tracking accuracy, then $v=9.41*10^{-5}$ holds at *t=50*.

In summary, for the initial target states including diagonal and non-diagonal target states, the controlled system will converge to its target system under the control algorithm (6) with P designed as in Section IV.

VI. Conclusion

We have proposed a convergent orbit tracking control algorithm for the free-evolutionary target quantum system in this paper. The unitary transformation was used to change the tracking problem into the regulation one. For the convergence analysis in state regulation, the target states were divided into diagonal and non-diagonal ones. For the former, we continued to perfect the convergence analysis of diagonal mixed-states. The explicit convergence conditions of P have been obtained. For the non-diagonal target state, if the superposition state was considered, a specific non-diagonal P was designed to ensure the convergence. If the target state was non-diagonal mixed-state, there must be a unitary transformation to change the Hermitian non-diagonal matrix into the diagonal one, and the convergence conditions could been obtained as that of diagonal mixed-state.

Appendix 1: The proof of Lemma 1.

Prove: P is a diagonal matrix, one gets $\dot v\left(\rho_f\right)=0$ from (10).

$$\ddot v(\rho)=-i*\sum_m f_m\left\{tr\left(\dot H_{mt}[\rho,P]\right)+tr\left(H_{mt}[\dot\rho,P]\right)\right\}$$

$$\ddot v\left(\rho_f\right)=-\sum_m f_m^2 tr\left(\left[H_{mt},\rho_f\right]*\left[P,H_{mt}\right]\right)$$

$$=\sum_m f_m^2 tr\left(\left[H_{mt},\rho_f\right]*\left[H_{mt},P\right]\right)$$

Let $A=\left[H_{mt},\rho_f\right], B=\left[H_{mt},P\right]$, then $(A)_{ij}=\left(\lambda_j-\lambda_i\right)\left(H_{mt}\right)_{ij},(B)_{ij}=\left(p_j-p_i\right)\left(H_{mt}\right)_{ij}$, so

$$tr(AB)=\sum_{i=1}^n\sum_{j=1}^n A_{ij}B_{ji}=\sum_{i=1}^n\sum_{j=1}^n\left(\lambda_j-\lambda_i\right)\left(p_i-p_j\right)\left(H_{mt}\right)_{ij}^2 .$$

$$=-\sum_{i=1}^n\sum_{j=1}^n\left(\lambda_j-\lambda_i\right)\left(p_j-p_i\right)\left(H_{mt}\right)_{ij}^2$$

If ρ_f is a stable state, then $\ddot v\left(\rho_f\right)>0$, one gets: $\left(\lambda_i-\lambda_j\right)\left(p_i-p_j\right)<0,\forall i\neq j$.

Let $\{\mu_1,\mu_2\cdots\mu_n\}$ be the spectrum of ρ_f with μ_i arranged in a non-increasing order, viz $\mu_1<\mu_2<\cdots<\mu_n$. Then the corresponding P is $P=diag\left(p_1,p_2,\cdots p_n\right)$ and $p_1>p_2>\cdots>p_n$ is obtained by the above description. Any other states ρ_s in stable set $\mathcal{R}$ can be obtained by m times swapping arbitrary two elements of $\left\{\mu_1,\mu_2\cdots\mu_n\right\}$. We let $bool=tr(P\rho_f)-tr(P\rho_s)$.

Suppose the spectrum from smallest to largest of target state is $\left\{\mu_1,\mu_2,\cdots,\mu_i,\cdots,\mu_j,\cdots,\mu_k,\cdots\mu_n\right\}$, then:

$i\leftrightarrow j$:

$bool=p_i\left(\mu_i-\mu_j\right)+p_j\left(\mu_j-\mu_i\right)=\left(p_i-p_j\right)\left(\mu_i-\mu_j\right)<0$

$i\leftrightarrow j, j\leftrightarrow k$:

$$
\begin{aligned}
bool &= p_i\left(\mu_i-\mu_j\right)+p_j\left(\mu_j-\mu_k\right)+p_k\left(\mu_k-\mu_i\right)\\
&= p_i\left(\mu_i-\mu_j\right)+p_j\left(\mu_j-\mu_i+\mu_i-\mu_k\right)+p_k\left(\mu_k-\mu_i\right)\\
&=\left(p_i-p_j\right)\left(\mu_i-\mu_j\right)+\left(p_j-p_k\right)\left(\mu_i-\mu_k\right)<0
\end{aligned}
$$

$i \leftrightarrow j, j \leftrightarrow k, k \leftrightarrow l$:

$$
\begin{aligned}
bool &= p_i\left(\mu_i-\mu_j\right)+p_j\left(\mu_j-\mu_k\right)+p_k\left(\mu_k-\mu_l\right)+p_l\left(\mu_l-\mu_i\right)\\
&= p_i\left(\mu_i-\mu_j\right)+p_j\left(\mu_j-\mu_i+\mu_i-\mu_k\right)\\
&\qquad +p_k\left(\mu_k-\mu_i+\mu_i-\mu_l\right)+p_l\left(\mu_l-\mu_i\right)\\
&=\left(p_i-p_j\right)\left(\mu_i-\mu_j\right)+\left(p_j-p_k\right)\left(\mu_i-\mu_k\right)+\left(p_k-p_l\right)\left(\mu_i-\mu_l\right)<0
\end{aligned}
$$

and so on. Finally, we get $v\left(\rho_f\right)<v\left(\rho_s\right)$. Lemma 1 is proved.

ACKNOWLEDGMENT

This work was supported in part by the National Key Basic Research Program under Grants No. 2011CBA00200, the National Science Foundation of China under Grant No. 61074050.

REFERENCES

[1] A. P. Peirce, M. A. Dahleh, and H. Rabitz, “Optimal control of quantum-mechanical systems: Existence, numerical approximation, and applications”, Physical Review A, Vol. 37, No. 12, Jun. 1988, pp. 4950–4964, doi: 10.1103/PhysRevA.37.4950.

[2] S. Cong, Y. Zhang, and K. Li, “Optimal control of mixed-state quantum systems based on Lyapunov method”, Biosignals, Rome, 2011, pp. 22-30.

[3] M. Sugawara, “General formulation of locally designed coherent control theory for quantum system”, Journal of Chemical Physics, Vol. 118, No. 15, Apr. 2003, pp. 6784-6800, doi:10.1063/1.1559680.

[4] J. R. West, D. A. Lidar, B. H. Fong and M. F. Gyure, “High fidelity quantum gates via dynamical decoupling”, Physical Review Letter, Vol. 105, No. 23, Dec. 2010, pp. 230503: 1-4, doi: 10.1103/PhysRevLett.105.230503.

[5] X. Peng, D. Suter, and D. A. Lidar, “High fidelity quantum memory via dynamical decoupling: theory and experiment”, Journal of physics B: Atomic, Molecular and Optical Physics, Vol. 44, July 2011, pp. 154003: 1-6, doi:10.1088/0953-4075/44/15/154003.

[6] J. Liu and S. Cong, “Trajectory Tracking of Quantum States based on Lyapunov method”, The 9th IEEE International Conference on Control & Automation (ICCA 11), Santiago, Dec. 2011, pp.318-323, doi: 10.1109/ICCA.2011.6137903.

[7] M. Mirrahimi, G. Turinici, and P. Rouchon, “Reference Trajectory Tracking for Locally Designed Coherent Quantum Controls”, Journal of Physical Chemistry A, Vol. 109, No. 11, 2005, pp. 2631-2637, doi: 10.1021/jp0472461.

[8] K. Beauchard, J. Coron, M. Mirrahimi, and P. Rouchon, “Implicit Lyapunov control of finite dimensional schrodinger equations”, Systems & Control Letters, Vol. 56, 2007, pp.388-395, doi:10.1016/j.sysconle.2006.10.024.

[9] M. Mirrahimi, P. Rouchon, and G. Turinici, “Lyapunov control of bilinear Schrodinger equations”, Automatica, vol. 41, Aug. 2005, pp. 1987-1994, doi: 10.1016/j.automatica.2005.05.018.

[10] M. Mirrahimi and P. Rouchon, “Trajectory tracking for quantum systems: a Lyapunov approach”, Proc. International Symposium MTNS, 2004, pp. 1-6.

[11] K. Sen and S. Cong, “Population control on equilibrium states of quantum systems via Lyapunov method”, Acta Automatica Sinica, Vol. 36, No. 9, Sep. 2010, pp. 1257-1263, doi: 10.1016/S1874-1029(09)60052-5.

[12] S. Kuang and S. Cong, “Lyapunov stabilization strategy of mixed-state quantum systems with ideal conditions”, Control and Decision, Vol. 25, No. 2, Feb. 2010, pp. 273-277.

[13] J. LaSalle and S. Lefschetz, Stability by Lyapunov’s Direct Method with Applications, 1rd ed., New York: Academic Press, 1961, pp. 56-60.

[14] M. Hou, G. Duan and M. Guo, “New versions of Barbalat's lemma with applications”, Journal of Control Theory & Application, Vol. 8, Iss. 4, No. 2010, pp. 545-547, doi: 10.1007/s11768-010-8178-z.

[15] S. Cong and J. Liu, “Orbit Tracking Control of Quantum Systems”, arXiv:1211.5447v2,17 Jan 2013

[16] R. I. Karasik, K. P. Marzlin, B. C. Sanders, and K. Birgitta Whaley, “Criteria for dynamically stable decoherenec-free subspace and incoherently generated Coherences”, Physical Review A, Vol. 77, No. 5, 2008, pp. 052301: 1-12, doi: 10.1103/PhysRevA.77.052301.

Face Recognition Using 1DLBP Texture Analysis

Amir Benzaoui and Abdelhani Boukrouche
Laboratory of Inverses Problems, Modeling, Information and Systems (PI:MIS)
Department of Electronics and Telecommunication
University of 08 Mai 1945
P.O box 401, Guelma, Algeria
amirbenzaoui@gmail.com abdelhanib@yahoo.fr

***Abstract*— A new algorithm for face recognition is proposed in this work; this algorithm is mainly based on Local Binary Pattern texture analysis in one dimensional space and Principal Component Analysis as a technique for dimensionalities reduction. The extraction of the face's features is inspired from the principal that the human visual system combines between local and global features to differentiate between people. Starting from this assumption, the facial image is decomposed into several blocks with different resolutions, and each decomposed block is projected in one dimensional space. Next, the proposed descriptor is applied for each projected block. Then, the resulting vectors will be concatenated in one global vector. Finally, Principal Component Analysis is used to reduce the dimensionalities of the global vectors and to keep only the pertinent information for each person. The experimental results have shown that the proposed descriptor Local Binary Pattern in one Dimensional Space combined with Principal Component Analysis have given a very significant improvement at the recognition rate and the false alarm rate compared with other methods of face recognition, and a good effectiveness against different external factors as: illumination, rotations, and noise.**

Keywords— face recognition; local binary pattern (LBP); local binary pattern in one dimensional space (1DLBP); texture description; dimesionalitiy reduction; Principal Compenent Analysis (PCA).

I. INTRODUCTION

The automatic analysis of the human face has become recently an active research area in the artificial vision and patterns recognition domains, due to its important use in several applications such as electronic election, biometrics and video surveillance [1, 2, 3]. Face analysis includes face detection and tracking, face recognition, age and gender recognition, and emotion recognition. Human face is dynamic entity, which changes under the influence of several factors as pose, size, occlusion, background complexity, lighting and the presence of some components such as mustaches, beard, and glasses. So, the essential key for any face analysis problem is on how to find an efficient descriptor to represent and to model the face in a real context?

The crucial step in any problem of face analysis is the phase of features extraction. In this phase, there are two major approaches, local and global approaches. Global approaches are based on pixel information; all pixels of the facial image are treated as a single vector; the vector size is the total number of the image pixels [4]. Most of the methods of this approach use another space of representation (subspace) to reduce the number of pixels and to eliminate the redundancies. Principal Component Analysis (PCA) [5] and Linear Discernment Analysis (LDA) [6] are the most popular methods used to reduce the dimensions and to select the useful information. However, these approaches are not effective in the unconstrained cases, i.e., situation where occlusion, lighting, pose, and size of the face are uncontrolled.

Recently, the scientists concentrate on local approaches, which are considered as a robust approaches in the unconstrained cases compared with global approaches; in this case, the face analysis is given by the individual description of its parts and their relationships, this model corresponds to the manner of perception by the visual human system. The methods of this approach are based on the extraction of features from the facial image and the definition of an adequate model to represent the face [7]. Several methods and strategies have been proposed to model and classify faces essentially based on the texture, normalized distances, angles and relations between eyes, mouth, nose and edge of the face. Local Binary Pattern (LBP) [8], Local Gabor Binary Pattern (LGBP) [9] and Oriented Edge Magnitudes (POEM) [10] are the recent methods in this approach.

Psychological and neuroscience studies have showed that the human visual system combines between local and global features to differentiate between people [11]. LBP is the best descriptor for capturing the local features, but it is not performed in the description of the global features [8]. From these assumptions, we propose in this work a new feature extraction method based on a descriptor proposed in our laboratory in [12, 13, 14], called 1DLBP (Local Binary Pattern in One Dimensional Space), inspired from classical LBP. The proposed method, which is applied on face recognition, is characterized by the combination of the local and global features for modeling faces. The algorithm of extraction is decomposed into five principal stages; first, the input image is decomposed into several blocks with different resolutions. A vertical projection in one dimensional space is applied for each decomposed block. Next, the proposed descriptor is applied on each projected block. Then, the resulting vectors from each block are concatenated in one global vector. Finally, Principal Component Analysis is needed to regroup the global vectors, to reduce the dimensionalities and to keep only the useful information for each individual.

This paper is organized as follows: in the next section, we describe the classical LBP and histogram feature. In Section 3, the proposed algorithm of feature extraction for face recognition is presented. For this purpose, *chi-square* distance is required to measure similarities between face templates. In Section 4, we present our experimental results by applying the proposed algorithm on ORL and AR databases. Finally, a conclusion related to this work is given in Section 5.

II. LOCAL BINARY PATTERN (LBP)

The original LBP operator introduced by Ojala *et al.* [15], which has been used for texture discrimination, has shown a powerful and effective results against to the variations in rotation and illumination. The operator labels the pixels of an image by thresholding the 3×3 neighborhood of each pixel with the central value and considering the result as a binary code. Next, the histogram of the labels can be used as a texture descriptor (see Figure 1); for a given pixel $g_c(x_c,y_c)$ from gray image, its texture LBP is calculated by comparing g_c with its neighbors pixels P on a circle of radius R (see Figure 2 for more details on circular neighborhood). The value of LBP(g_c) is obtained as:

$$\mathrm{LBP}^{P,R}(x_c, y_c) = \sum\nolimits^{P}_{i=1} S(g_i^{P,R} - g_c)\, 2^{i-1} \qquad (1)$$

$$\text{S(x) is defined as: } S(x) = \begin{cases} 1 & \text{if } x \geq 0; \\ 0 & \text{otherwise;} \end{cases} \qquad (2)$$

The major disadvantage of the original LBP operator resides in the size of the descriptor, a mask of 3×3 pixels cannot capture the structures of large scale which can considered as a dominants structures in the image. Recently, the size of the operator has been extended by using a mask with different large sizes. Figures 2.a, 2.b, 2.c show three examples of the extended LBP.

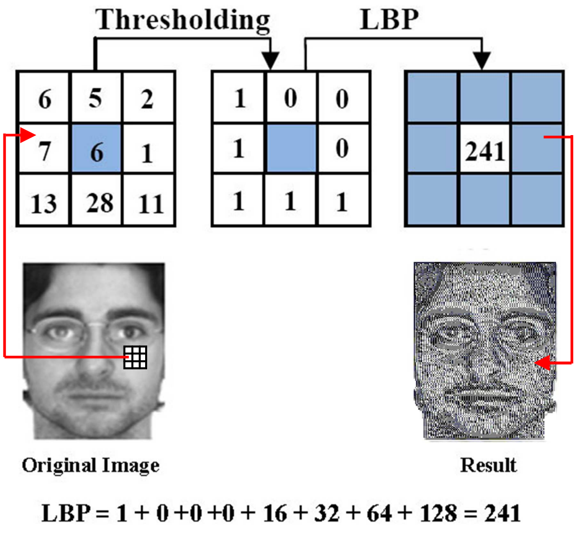
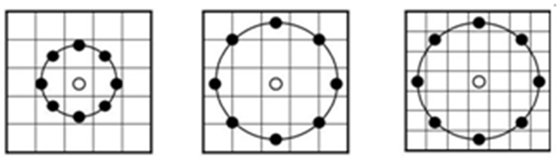
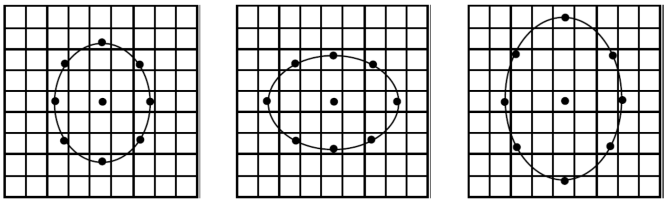


Figure 1. LBP calculation performed into 3×3 neighborhood.



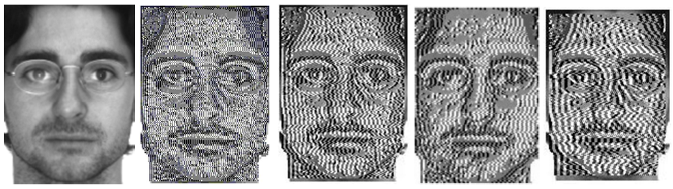
(a) (b) (c)

Figure 2. Neighborhood set for different (P,R). (a) The basic LBP operator (P,R) = (8,1) (b) LBP with circular neighborhood (8,2). (c) LBP with circular neighborhood (8,3).



(8,2,3) (8,3,2) (8,3,4)

Figure 3. ELBP samples with different extention [16].



Original image LBP (8,1) LBP (8,2) ELBP (8,3,2) ELBP (8,3,4)

Figure 4. Results of LBPs application with differentd masks .

Another type of the extended operators of LBP called: Elliptical Local Binary Patterns (ELBP) [16]. In ELBP, at each pixel $g_c(x_c,y_c)$, we consider its surrounding pixels that lie on an ellipse (see Figure 3) with (x_c,y_c) is the center. ELBP of (x_c, y_c) with P neighboring pixels at (R1, R2) distances is computed as:

$$\mathrm{ELBP}^{P,R1,R2}\ (x_c, y_c) = \sum\nolimits^{P}_{i=1} S(g_i^{P,R1,R2} - g_c)\, 2^{i-1} \qquad (3)$$

S(x) function is defined as (2).

In details, the coordinates of the i th neighboring pixel of (x_c,y_c) are calculated using the formulas:

$$\text{angle-step} = 2 * \pi / P \qquad (4)$$

$$x_i = x_c + R1 * \cos((i - 1) * \text{angle-step}) \qquad (5)$$

$$y_i = y_c + R2 * \sin((i - 1) * \text{angle-step}) \qquad (6)$$

However, the extend versions of LBP operators and the Elliptical LBPs present a good results by capturing the local and global patterns but they are not performed for capturing the micro characteristics (fine details) of the human face. Figure 4 shows the rsults of LBP and ELBP applications using different masks.

III. PROPOSED APPROACH

The proposed algorithm used to extract information for face recognition is described in the following recapitulation; next, we present each step in details. We consider that we have a gallery θ of biometric samples with P persons, S biometric sample (image) per person, N training images for each person

P_i and M testing images (example θ=400, P=40, S=10, N=5, M=5, with ORL database, θ=2600, P=100, S=26, N=13, M=13 with AR database). The process of features extraction is composed of six principal stages for each person:

- Preprocessing of the N images.
- Decomposition of each image N_i into several blocks with different resolution.
- Projection of each block decomposed in one dimensional space.
- Application of the proposed descriptor 1DLBP for each projected block.
- Concatenation of the resulting vectors of an image N_i in one global vector V_i.
- Dimensionalities reduction of the V grouped global vectors using PCA.

A. *Preprocessing*

The objective of the preprocessing is the modification of the source's image representation to facilitate the task of the following steps and to improve the rate of recognition. First, the facial image is converted into grayscale image. Next, every grayscale image is filtered by median filter to suppress noise. Lastly, the noise suppression image is then adjusted to improve the contrast of the image.

B. *Image decomposition*

Most LBP operators characterize the face texture distribution of each pixel with its neighborhood only. But, the differences between two faces can be demonstrated not only by the texture distribution of each pixel with its neighborhood, but also by the relative connection with other pixels. With this intention, we have decomposed the original image into several sub-images (see Figure 5) to characterize better the details and the relationships between all the image pixels. Next, the extracted histograms will be concatenated in one global vector in the next stages. With this technique, we can obtain the fine details and the relative connections between all pixels.

C. *1D vertical projection*

The 1D projection of rows or columns of each level provides an effective mean to describe better the local and global patterns. Figure 6 presents an example of vertical projection. The objective of the projection is to validate the descriptor LBP in one dimensional space to find another mean for describing and analyzing better the human face's texture.

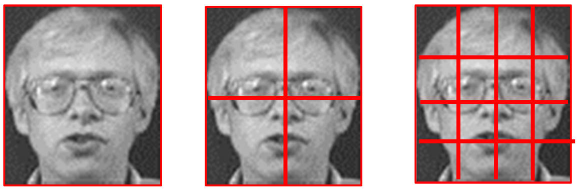


Figure 5. Image decomposition in differents blocks.

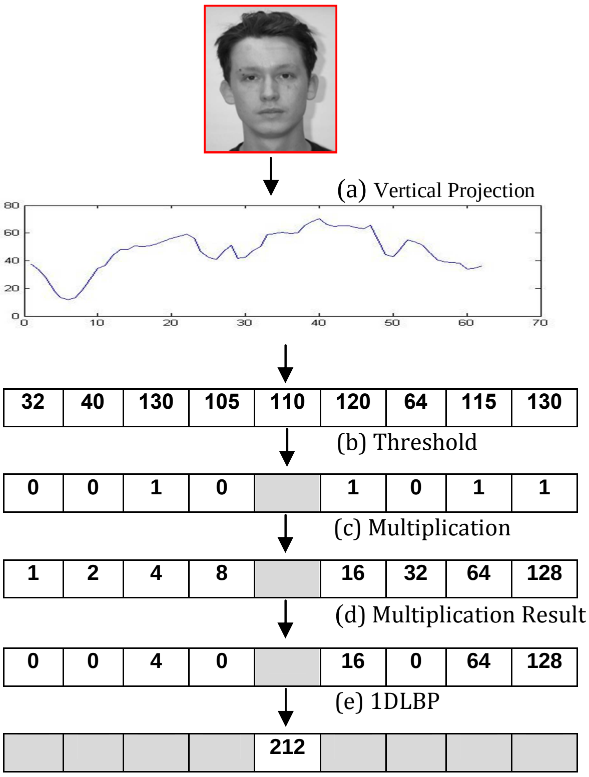
D. *1DLBP application*

The concept of the 1DLBP method consists in a binary code describing the local agitation of a segment in 1D signal. It is calculated by thresholding of the neighborhood values with the central value. All neighbors get the value 1 if they are greater or equal to the current element and 0 otherwise. Then, each element of the resulting vector is multiplied by a weight according to its position (see Figure 6.c). Finally, the current element is replaced by the sum of the resulting vector. This can be recapitulated as follows:

$$1DLBP = \sum_{n=0}^{N-1} S(g_n - g_0).2^n \tag{7}$$

S(x) function is defined as (2).

g_0 and g_n are respectively the values of the central element and its 1D neighbors. The index *n* increases from the left to the right in the 1D string as shown in Figure 6.c. The 1DLBP descriptor is defined by the histogram of the 1D patterns.



32	40	130	105	110	120	64	115	130
0	0	1	0		1	0	1	1
1	2	4	8		16	32	64	128
0	0	4	0		16	0	64	128
				212				

Figure 6. Example of vertical projection in 1D space + 1DLBP application.

Copyright (c) IARIA, 2013. ISBN: 978-1-61208-272-1

E. *Concatenation of the resulting vectors*

The proposed descriptor 1DLBP is applied on all blocks of the decomposed image with the different resolution presented in Figure 5. The extracted histograms from each block are concatenated in one global histogram (vector) representing one face image V_i. A problem of information redundancies is appeared due to the important size of each global vector. To resolve this problem, we have used the Principal Component Analysis (PCA) as a technique of dimensionalities reduction, to regroup all the global vectors of each person Xi in one global matrix and to select the useful information needed to modeling each person.

F. *Dimensionalities reduction with PCA*

The principal idea of PCA is to represent a group of images (vectors) of a same person X in another space of lower dimension; this space is constructed from a set of training images. PCA begins with a set of 1D training vectors of the same class; each vector Γ_i represents a training image I_i (I = 1...N), and construct $O_i = \Gamma_i - \acute{\Gamma}$ where $\acute{\Gamma}$ represents the mean vector of all Γ_i vectors. Then, determinate the eigenvectors μ_i of the covariance matrix $C = \sum_i O_i \times O_i^T$. The first K "Principal axis" corresponds to the K largest eigenvalues (the value of K is chosen in the same that the sum of the first axis K provides a large proportion of the total eigenvalues sum).

Now, given a new image ξ considered as a test example. First, we built its 1D representation using the extraction method proposed in this work (stages: 1-5), and we subtract the average face as follows: $\theta = \xi_{1D} - \acute{\Gamma}$. Next, we project the image in the principal axis elaborated as: $\theta_p = \sum_{i=1,K} \mu_i^T (\theta - \mu_i)$. Finlay, we calculate the chi-Square distance to classify the image ξ in the nearest class corresponding to a degree of similarities that exceeds a fixed threshold.

$$\text{Dist}_{\text{chi-square}}(X, Y) = \sum_{i=0}^{M} \frac{(xi-yi)2}{xi+yi} \qquad (8)$$

IV. EXPERIMENTAL RESULTS

To evaluate the performances of the proposed algorithm, we have carried out several tests on ORL and AR databases; we randomly selected half of samples for training set and the remaining samples for testing set. In all our experiments, we considered the average recognition rates of several random permutations (50 permutation with ORL database and 100 permutations with AR database), and we compared the obtained results (identification and false alarm rates) with other methods using the same testing protocols. Our experiments are implemented with *Matlab* 2010a, *Windows* 7, HP Core 2 Duo, 3 Ghz CPU with 2 Gb Ram.

A. *ORL database*

The ORL database contains 400 frontal images in different facial expression, conditions of illumination, hairstyles with or without beard, moustaches and glasses for 40 persons, 10 images for each person. Each sample is a 92×112 gray image, with tolerance for some tilting and rotation of up to 20° (see Figure 7).

B. *AR database*

The AR database was collected at the Computer Vision Center in Barcelona, Spain in 1998 [17]. It contains images of 116 individuals (63 men and 53 women). The imaging and recording conditions (camera parameters, illumination setting, and camera distance) were carefully controlled and constantly recalibrated to ensure that settings are identical across subjects. The resulting RGB color images are 768×576 pixels in size. The subjects were recorded twice at a 2–week interval. During each session 13 conditions with varying facial expressions, illumination and occlusion were captured (see Figure 8).

First, we applied the methods inspired from the LBP texture analysis in the two databases (classical LBP, extended LBP, Elliptical LBP, and the one dimensional projected LBP). The performances of these methods are shown in Table 1.

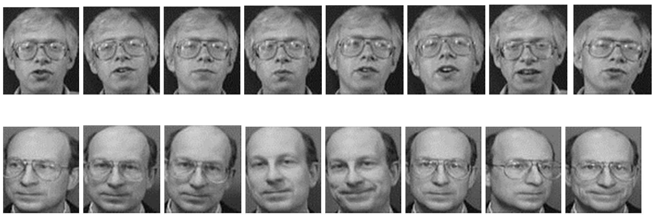


Figure 7. Some images from ORL database.

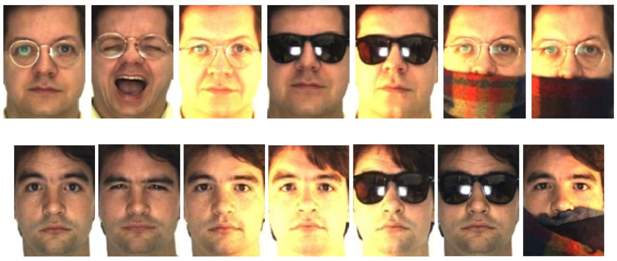


Figure 8. Some images from AR database [17].

TABLE I. COMPARATIVE RECOGNITION RESULTS OF THE ISPIRED LBP METHODS ON ORL AND AR DATABASES

RR: Recognition Rate %.
FAR: False Alarm Rate %.

	ORL **RR %**	**AR** **RR %**	**Average** **RR %**	**Average** **FAR %**
LBP (8,1)	85,2	87,5	86,4	3,62
LBP (8,1) PCA	91,4	93	92,2	1,92
LBP (8,2)	86	89,4	87,7	3,1
ELBP (8,3,2)	84,3	88,1	86,2	2,97
ELBP (8,3,4)	83,8	86,9	85,4	3,03
1DLBP	92	92,6	92,3	2.36
1DLBP PCA	95,8	97,9	**96,9**	**1.44**

TABLE II. COMPARAISON WITH OTHER STATE OF THE ART APPROACHS

	ORL RR %	AR RR %	Average RR %	Average FAR %
LBP (8,1) PCA	91,4	93	92,2	1,92
1DLBP PCA	95,8	97,9	**96,9**	**1.44**
Local Gabor Binary Pattern (LGBP)	94,1	96,2	95,2	1,87
POEM	90,9	89,4	90,2	2,6
PCA	85	81,2	83,1	5,12
LDA	82,4	84,7	83,6	5,36

The results of the experiments clearly showed that the projected Local Binary Pattern with dimensionalities reduction using PCA enhances the recognition performance in all configurations and presents a very good improvement and significant results in recognition rate, false alarm rate against other variants of LBP.

We also conducted tests comparing our method against recent and classical state-of-the-art approaches using the similar protocols under more challenges and scenarios. The results, shown in Table 2, indicate clearly the effectiveness of our approach which outperforms all other methods.

The facial images are taken with ORL database which is considered as a stable database in the unconstrained cases, and the AR database which presents a very good variation, in lighting, occlusion and facial expression, to measure the performances of the approach in the difficult situations. The comparison results presented in Table 2 shows that our proposed approach presents very good results with AR database which indicates that this approach has an important effectiveness against to the variations in different factors like: occlusion, lighting, rotation, and noise.

Another conclusion we can make from Table 1 and Table 2 is that 1DLBP + PCA is much better than 1DLBP only; the association of the PCA as a robust technique in the dimensionalities reduction is very interesting to improve the performances of the proposed approach.

V. CONCLUSION AND FUTURE WORK

Facial image can be considered as a composition of local and global features. Starting from this assumption, we have successfully developed a new algorithm for texture face discrimination, this algorithm is primary based on Local Binary Patterns but projected in one dimensional space; it combines between local and global features, capable of recognizing faces in different situations. Each facial image is decomposed on multi blocks with different resolution and each decomposed block will be vertically projected in one dimensional space. Next, the proposed descriptor is applied on each projected block. Then, the extracted vectors from each block will be concatenated in one global vector. Finally, Principal Component Analysis is applied on the regrouped vectors to reduce the dimensionalities of the concatenated vectors and to extract the useful information. The experimental results applied on ORL and AR database have showed that the proposed approach has given a very significant improvement at the recognition rate, false alarm rate and a good effectiveness against different external factors as: occlusion, illumination, rotations, and noise.

As a prospect of this work, we hope to apply the proposed descriptor in other application of face analysis, like age, or gender recognition, to apply the same descriptor with other modalities of biometric system, like the use of the human ear in identity recognition.

REFERENCES

[1] M. Khasawneh, M. Malkawi, O. Al-Jarrah, T.S. Hayajneh, and M.S. Ebaid, “A Biometric-Secure e-Voting System for Election Processes,” Proc. IEEE Sym. on Mechatronics and its Applications (ISMA 8), Amman, Jordan, May 2008, pp. 371-376. doi: 10.1109/ISMA.2008.4648818

[2] A. Hadid, J.L. Dugelay, and M. Pietikainen, “On the use of Dynamic Features in Face Biometrics: Recent Advances and Challenges,” Signal, Image and Video Processing (Springer), vol. 5, no. 4, Nov 2011, pp. 495-506. doi: 10.1007/s11760-011-0247-3

[3] L. Ding and A.M. Martinez, “Features versus Context: An approach for precise and detailed detection and delineation of faces and facial features,” IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 32, no. 11, Nov 2010, pp. 2022-2038. doi: 10.1109/TPAMI.2010.28

[4] R.S. Kathavarayan and M. Karuppasamy, “Identification of Untrained Facial Image in Combined Global and Local Preserving Feature Space,” International Journal of Biometrics and Bioinformatics (IJBB), vol. 4, no. 1, Mar 2010, pp. 1-12.

[5] M.A. Turk and M.P. Pentland, “Eigenfaces for recognition,” Journal of Cognitive Neuroscience, vol. 3, no. 1, 1991, pp. 71-86. doi: 10.1162/jocn.1991.3.1.71

[6] N. Belhumeur, J.P Hespanha, and D.J. Kriegman, “Eigenfaces vs Fisherfaces: Recognition using class specific linear projection,” IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 19, no. 7, Jul 1997, pp. 711-720. doi: 10.1109/34.598228.

[7] T. Cootes, C. Taylor, H. Kang, and V. Petrovié, “Handbook of Face Recognition”. Chapter.3: “Modeling Facial Shape and Appearance”. Springer, 1st Edition, 2005, pp. 40. doi: 10.1007/b138828

[8] T. Ahonen, A. Hadid, and M. Pietikainen, “Face Recognition with Local Binary Pattern,” Proc. Springer. European Conference on Computer Vision (ECCV 8), Prague, Czech, May 2004, pp. 469-481. doi: 10.1007/978-3-540-24670-1_36

[9] N. Nguyen, L. Bai, and L. Shen, “Local Gabor Binary Pattern whitened PCA: A novel approach for face recognition from single image per person,” Proc. IEEE / Springer Conf. On Biometrics (ICB 3), Alghero, Italy, Jun 2009, pp. 269-278. doi: 10.1007/978-3-642-01793-3_28.

[10] N.S. Vu and A. Caplier, “Face recognition with patterns of oriented edge magnitudes,” Proc. Springer. European Conference on Computer Vision. (ECCV 11), Heraklion Crete, Greece, Sep 2010, pp. 313-326. doi: 10.1007/978-3-642-15549-9_23.

[11] A.J O'Toole, D.A Roark, and H. Abdi, “Recognizing moving faces: A psychological and neural synthesis,” Trends in Cognitive Science, vol. 6, no. 6, Mar 2010, pp. 261-266.

[12] L. Houam, A. Hafiane, R. Jennane, A. Boukrouche, and E. Lespessailes, “Trabecular Bone Anisotropy Characterisation Using 1D Local Binary Patterns,” Proc. Springer Conf. Advanced Concepts for Intelligent Vision Systems (ACIVS 12), Sydney, Australia, Dec 2010, pp. 105-113, doi: 10.1007/978-3-642-17688-3_11

[13] L. Houam, A. Hafiane, R. Jennane, A. Boukrouche, and E. Lespessailes. “Trabecular Bone Texture Classification Using 1D LBP and Wavelets Coefficients in High-pass Bands,” Proc. Conf. Signal, Image, Vision and their Application (SIVA), Guelma, Algeria, Nov 2011, pp. 116-121.

[14] L. Houam, A. Hafiane, R. Jennane, A. Boukrouche, and E. Lespessailes, “Texture characterization using Local Binary Pattern and Wavelets. Application to Bone Radiograph,” Proc. IEEE Conf. Image Processing, Theory, Tools and Applications (IPTA 3), Istanbul, Turkey, Oct 2012, pp. 371-376. doi: 10.1109/IPTA.2012.6469546

[15] T. Ojala, M. Pietikainen, and D. Harwood, “A comparative study of texture measures with classification based on featured distribution,” Pattern Recognition, vol. 29, no. 6, 1996, pp. 51-59. doi: 10.1016/0031-3203(95)00067-4

[16] H.T. Nguyen, N.S. Vu, and A. Caplier, “How far we can improve micro features based face recognition systems?,” Proc. IEEE Conf. Image Processing, Theory, Tools and Applications (IPTA 3), Istanbul, Turkey, Oct 2012, pp. 350-353. doi: 10.1109/IPTA.2012.6469533

[17] A.M. Martinez and R. Benavente, “The AR face database”, CVC Technical Report, 24, 1998.

Enhancing Hebbian Learning in Biological Neural Cultures Through Electrical Stimulation

Víctor Lorente, Jose Manuel Ferrández
Dpto. Electrónica, Tecnología de Computadores y Proyectos,
UPCT, Cartagena, Spain
{victor.lorente, jm.ferrandez}@upct.es

Eduardo Fernández
Instituto de Bioingeniería
Universidad Miguel Hernández
Elche, Spain
e.fernandez@umh.es

Felix de la Paz
Dpto. de Intelegencia Artificial
UNED
Madrid, Spain
delapaz@dia.uned.es

Abstract— **Electric stimulation has been widely used to induce changes in neuronal cultures coupled to microelectrode arrays (MEAs). In this paper, we used low-frequency current stimulation on dissociated cultures of hippocampal cells to study how neuronal cultures could be trained with this kind of stimulation. We show that persistent and synchronous stimulation of adjacent electrodes may be used for creating adjacent physical or logical connections in the connectivity graph following Hebb's Law.**

Keywords- Cultured neural networks; Hebbian Law; Induced plasticity; Learning.

I. Introduction

Using biological nervous systems as conventional computer elements is a fascinating problem that permits the hybridization between Neuroscience and Computer Science. This synergic approach can provide a deeper understanding of natural perception and may be used for the design of new computing devices based on natural computational paradigms. Classical computational paradigms consist in serial and supervised processing computations with high-frequency clocks silicon processors, with moderate power consumption, and fixed circuits structure. However the brain uses millions of biological processors, with dynamic structure, slow commutations compared with silicon circuits, low power consumption and unsupervised learning. This kind of computation is more related to perceptual recognition [1,2], due to the natural variance of the perceptive patterns and the a priori lack of knowledge about the perceptual domain.

A real biological processor with millions of biological neurons and a huge number of interconnections would provide much more computational power instead of their low transition rates due to high number of computing elements and the extraordinary network capability of adaptation and reconfiguration to unknown environments. This extraordinary capability is related with natural unsupervised learning.

Microelectrode Arrays (MEAs) have been designed for direct culturing neural cells over silicon or glass substrates, providing the capability to stimulate and record simultaneously populations of neural cells. The use of dissociated cortical neurons cultured onto MEAs represents a useful experimental model to characterize both the spontaneous behaviour of neuronal populations and their activity in response to electrical and pharmacological changes.

Learning is a natural process that needs the creation and modulation of sets of associations between stimuli and responses. Many different stimulation protocols have been used to induced changes in the electrophysiological activity of neural cultures looking for achieve learning [3-13] and low-frequency stimulation has brought good results to researchers enhancing bursting activity in cortical cultures [10,11].

Hebbian learning describes a basic mechanism for synaptic plasticity wherein an increase in synaptic efficacy arises from the presynaptic cell's repeated and persistent stimulation of the postsynaptic cell. The theory is commonly evoked to explain some types of associative learning in which simultaneous activation of cells leads to pronounced increases in synaptic strength. Basically the efficiency of a synaptic connection is increased when presynaptic activity is synchronous with post-synaptic activity. In this work, we use this kind of stimulation to create adjacent physical or logical connections in the connectivity graphs using Hebb's Law.

In previous papers, we used a specific low-frequency current stimulation on dissociated cultures of hippocampal cells to study how neuronal cultures could be trained with this kind of stimulation [14, 15]. We showed that persistent and synchronous stimulation of adjacent electrodes may be used for creating adjacent physical or logical connections in the connectivity graph following Hebb's Law. In later experiments, we have used different parameters for this stimulation to check if those connections can be created stimulating with different configurations. The results provided in this paper show that low-frequency stimulation can create adjacent connections with different amplitude values. In addition, we present new results explaining when such connections have been established in a dissociated culture of hippocampal neurons grown onto a MEA.

ISBN: 978-1-61208-272-1

The outline of the paper is as follows. We first introduce the evolution of the related work on this topic. Next, we present the methods for addressing Hebbian Learning throw electrical stimulation. The following section shows the results obtained using a specific stimulation with our experimental setup on hippocampal cultures to train them. We conclude by discussing some crucial aspects of the research and the remaining challenges.

II. RELATED WORK

The first studies demonstrating functional plasticity in cultured networks began 1990s. The research group of Akio Kawana at NTT in Japan reported that tetanic stimulation through one or several electrodes resulted in plasticity [3]. They observed a change in the probability of evoking bursts by test pulses, as well as a change in the rate of spontaneous bursting, as a result of repeatedly evoking bursts using strong tetanic stimulation. Jimbo et al. reported similar results with a different tetanic stimulation and used voltage clamp to observe inward currents associated with evoked bursts [4]. The following year, Jimbo et al. reported that tetanizing a single electrode resulted in changes in the responses to test pulses to other electrodes [5]. In another paper, Jimbo et al. used simultaneous tetanization through a pair of electrodes to induce more precise forms of plasticity, expressed in detailed spike patterns evoked by electrical (probe) pulses [6]. Since then, a few other groups have reported on other forms of plasticity in MEA neural cultures.

Typically, these later papers have focused on more abstract plasticity results, more related to the network level than to the synaptic level. For instance, Shahaf and Marom reported that networks could be learn to respond in specific ways to test pulses, by repeatedly stimulating until the desired response was obtained [7], while Ruaro et al. reported that cultured networks could learn to extract a specific pattern from a complex image that had been presented repeatedly as spatial patterns of multielectrode stimulation [8].

In the following years, researchers have tried using more complex stimulation patterns in order to induce plasticity in neural cultures. Wagenaar et al. [9] looked for plasticity expressed in changes in spontaneous burst patterns, and in array-wide response patterns to electrical stimuli, following several induction protocols related to the previous ones, as well as some novel ones. Madhavan et al. [10] investigated patterns of spontaneous multi-single-unit activity to study the potential role of bursts of action potentials in memory mechanisms. Their analysis revealed spatiotemporally diverse bursts occurring in well-defined patterns, which remained stable for several hours. Chao et al. [11] compared five established statistical methods to one of their own design, called center of activity trajectory (CAT), to quantify functional plasticity at the network level. Stegenga studied the possibility of changing the spatio-temporal structure of spontaneous bursts using different configurations of tetanic stimulation. They obtained a profile of the array-wide spiking rate, a *burst profile* (BP) and also calculated the per-electrode spiking rate profile, the *phase profiles* (PPs). None of their stimulation methods had a measurable effect on of the specific burst statistics (peak firing rate, rise and fall times). However, they found many PP changes in their experiments, which can be seen as a confirmation that the analysis is sensitive to changes in the network.

Other researchers have focused in changing some stimulation parameters (voltage vs. current, frequency, amplitude…) to achieve learning. Brewer et al. [12] used chronic stimulation for getting an increase in evoked spike counts per stimulus and in spiking rate. The results obtained suggested that plastic network changes induced by chronic stimulation enhance the reliability of information transmission and the efficiency of multi-synaptic network communication. In turn, Martinoia et al. [13] applied low-frequency stimulation constantly applied over weeks. They found that the stimulation had a delayed effect modulating responsiveness capability of the network without directly affecting its intrinsic *in vitro* development.

III. METHODS

A. Cell Culture Preparation

Dissociated cultures of hippocampal CA1-CA3 neurons were prepared from E17.5 sibling embryos (Figure 1). During the extraction of the hippocampus a small amount of cortical tissue will have inevitably also been included. Tissue was kept in 2ml of HBSS. 10mg/ml of trypsin was added to the medium and placed in a 37° C water bath for 13 min for subsequent dissociation. The tissue was then transferred to a 15 ml falcon containing 4ml of NB/FBS and triturated using combination of fine pore fire polished Pasteur pipettes (Volac). Cells were then transferred onto 12 well plates (Corning Incorporated) containing glass coverslips (Thermo Scientific).

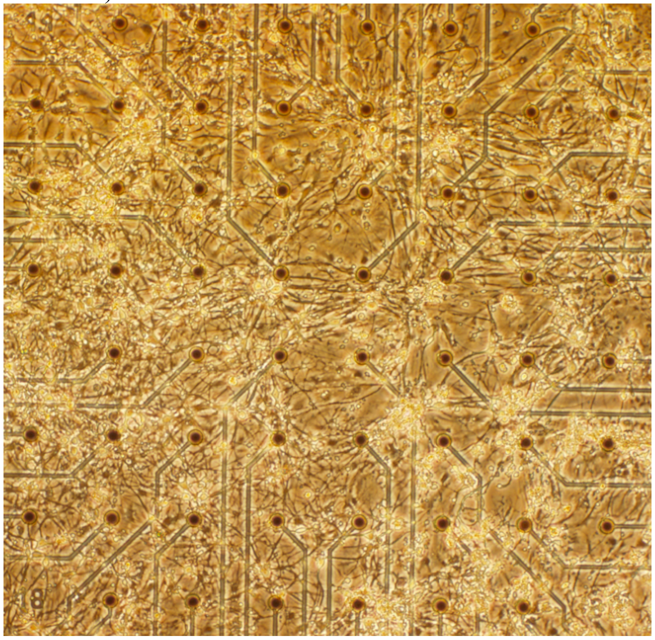


Figure 1. Hippocampal CA1-CA3 culture (21 DIV) on a microelectrodes array.

The coverslips were pre-treated overnight with PDL (50mg/ml), a synthetic molecule used as a coating to enhance cell attachment. The PDL was then aspirated away and the coverslips washed twice with PBS. This was then followed by a final coating of laminin (50µg/ml), a protein found in the extracellular matrix, to further help anchor the dissociated hippocampal cells. The cells were maintained in a mixture of 500ml NB/B27 (promotes neural growth) and 500ml NB/FBS (promotes glial growth), each supplemented with Glutamax and Pen/Strep (dilution 1/100). Glutamax improves cells viability and growth while preventing build up of ammonia and Pen/Strep helps to prevent any infections. Cell density for each coverslip was roughly 200000 cells. Cells were kept in an incubator at 37˚ C in 6% CO2.

B. *Expeerimental Setup*

Microelectrode arrays (Multichannel systems, MCS) consisted of 60 TiN/SiN planar round electrodes (200 µm electrode spacing, 30 µm electrode diameter) arrange in a 8x8 grid were used. Two pairs of electrodes were selected for creating functional connections between them. The activity of all cultures was recorded using a MEA60 System (MCS). After 1200X amplification, signals were sampled at 10kHz and acquired through the data acquisition card and MCRack software (MCS). Electrical stimuli were delivered through a two-channel stimulator (MCS STG1002) to each pair of electrodes.

C. *Expeerimental Protocol*

Three experiments were carried out using a total of 15 cultures during 2-3 weeks. Five cultures were stimulated in each experiment with a low-frequency electrical stimulation, which differs only on a few parameters. Each experiment started when cultures had 14DIV. The following summarizes the experiments and stimulation applied to the cultures:

1) *Experiment1 (E1): Cultures ID48-52*

a) *Days of experiment:* 16.

b) *Stimulation1:* train of 5 biphasic pulses cathodic-first (50 µA peak, 100 µs phase, 50ms ISI) was delivered every 3s for 10 min.

2) *Experiment2 (E2): Cultures ID68-72*

a) *Days of experiment:* 10.

b) *Stimulation2:* train of 5 biphasic pulses cathodic-first (60 µA peak, 100 µs phase, 50ms ISI) was delivered every 3s for 8 min.

3) *Experiment3 (E3): Cultures ID73-77*

a) *Days of experiment:* 11.

b) *Stimulation3:* train of 5 biphasic pulses cathodic-first (40 µA peak, 100 µs phase, 50ms ISI) was delivered every 3s for 8 min.

In every experiment, two pairs of electrodes with no logical connections between them were selected using connectivity diagrams based on cross-correlation. In every stimulation session these steps were followed:

1) Spontaneous activity was recorded for 2 min after a recovery period.

2) Cultures were then stimulated through the two pairs of electrodes using the corresponding stimulation protocol.

3) Spontaneous activity was recorded for 2 min after the stimulation.

D. *Analysis Performed*

We observed the spontaneous activity of the cultures before and after the stimulation experiments, as well as their evoked response to the applied stimulus. Extensive burst analysis, post-stimulus time histograms and functional connectivity were the main analysis performed to the registered data.

Correlation and information theory-based methods are used to estimate the functional connectivity [16, 17] of in-vitro neural networks: Cross-correlation, Mutual Information, Transfer Entropy and Joint Entropy. Such methods need to be applied to each possible pair of electrodes, which shows spontaneous electrophysiological activity. For each pair of neurons, the connectivity method provides an estimation of the connection strength (one for each direction). The connection strength is supposed to be proportional to the value yielded by the method. Thus, each method is associated to a matrix, the Connectivity Matrix (CM), whose elements (X, Y) correspond to the estimated connection strength between neuron X and Y.

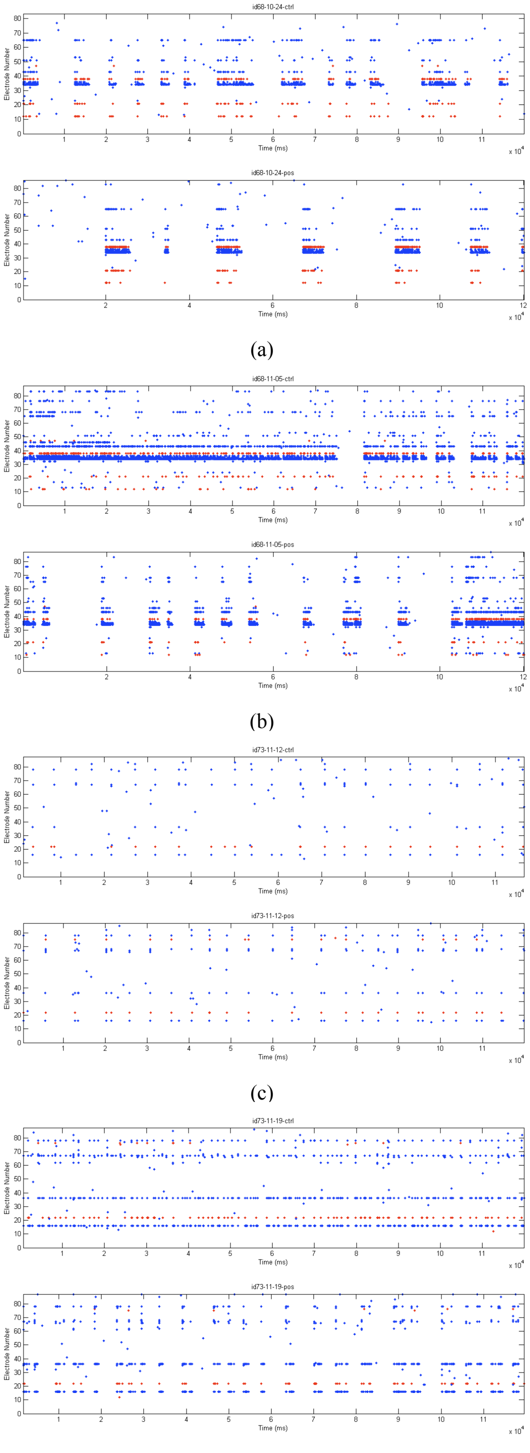
High and low values in the CM are expected to correspond to strong and weak connections. By using such approach, inhibitory connections could not be detected because they would be mixed with small connection values. However, non-zero CM values were also obtained when no apparent causal effects were evident, or no direct connections were present among the considered neurons.

In our experiments, *Connectivity maps* offered a visualization of the connectivity changes that occur in the culture. Connectivity maps were generated using the connectivity matrix (CM) obtained after applying the analysis and Cross-Correlation or Mutual Information. By setting thresholds in the CM, it is possible to filter out some small values that may correspond to noise or very weak connections. In consequence, these maps show the strongest synaptic pathways, and can be used for visualizing the neural weights dynamics, and validate the achieved learning.

IV. RESULTS

The low-frequency current stimulation used in this study had an impact on the electrophysiological responses of the cultures, as previous studies had reported [13]. Raster plots showed that all of the stimulations provided induce changes in the firing frequency of the cultures.

ISBN: 978-1-61208-272-1



(a)

(b)

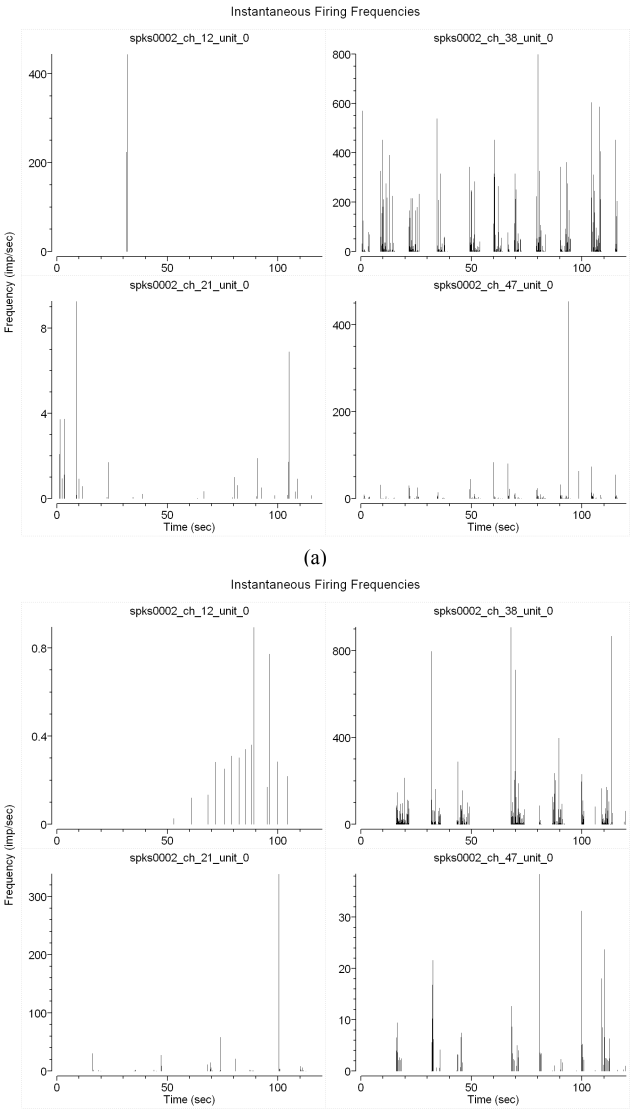
(c)

(d)

Figure 2. Raster plots extracted from cultures of experiments E2 and E3. (a) (21DIV) and (b) (32DIV) belong to ID68 from E2, (c) (30DIV) and (d) (37DIV) belong to ID73 from E3. Each figure is divided in two graphs, which show the spiking activity of the culture before and after stimulation. Raster plots show a change in the spiking activity, changing from a uniform activity before stimulation to a more concentrated activity after stimulation. This result is emphasized after the third week in vitro due to the maturing occurred in the cultures.

Furthermore, we can observe some kind of reorganization in the firing activity, from a uniform spiking activity to a discrete spiking activity. After the third week in vitro, the bursting activity becomes more frequent and robust and this effect is much more evident than during the first weeks (Figure 2).

The change on the spiking activity of the cultures can also be seen clearly observing the instantaneous firing frequencies (Figure 3) and the interspike intervals (Figure 4) over the time. Instantaneous firing frequency graphs shows that stimulated electrodes start firing in more separated period of times after stimulation but each firing period last longer.

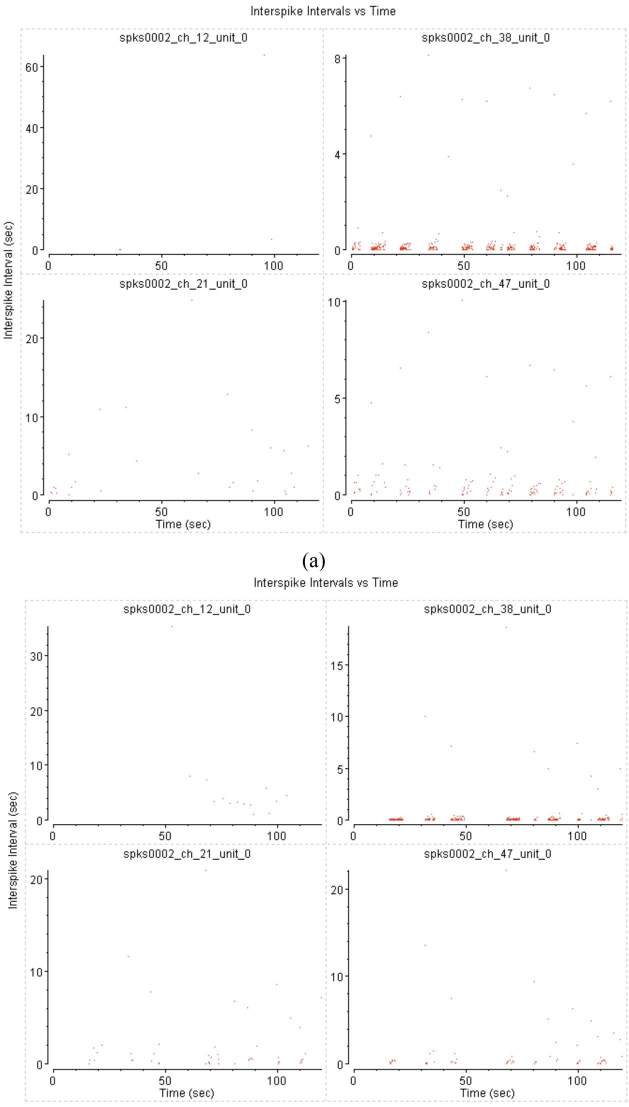


(a)

(b)

Figure 3. Instantaneous firing frequencies of stimulated electrodes in the culture ID68 (22DIV) from E2 before (a) and after (b) stimulation. A change in the spiking periods can clearly be seen, which are less in quantity and longer in duration. This culture only created a connection between the second pair of electrodes (38, 47), which had an impact on the instantaneous firing frequencies. Connected electrodes fire in the same firing periods, whereas not connected electrodes had no firing relation.

In addition, interspike intervals graphs show the previous results in the spiking periods but also it can be seen that the ISI decrease both in value and dispersion. Both effects are related to the stimulation, which modulates the firing capabilities of the cultures.



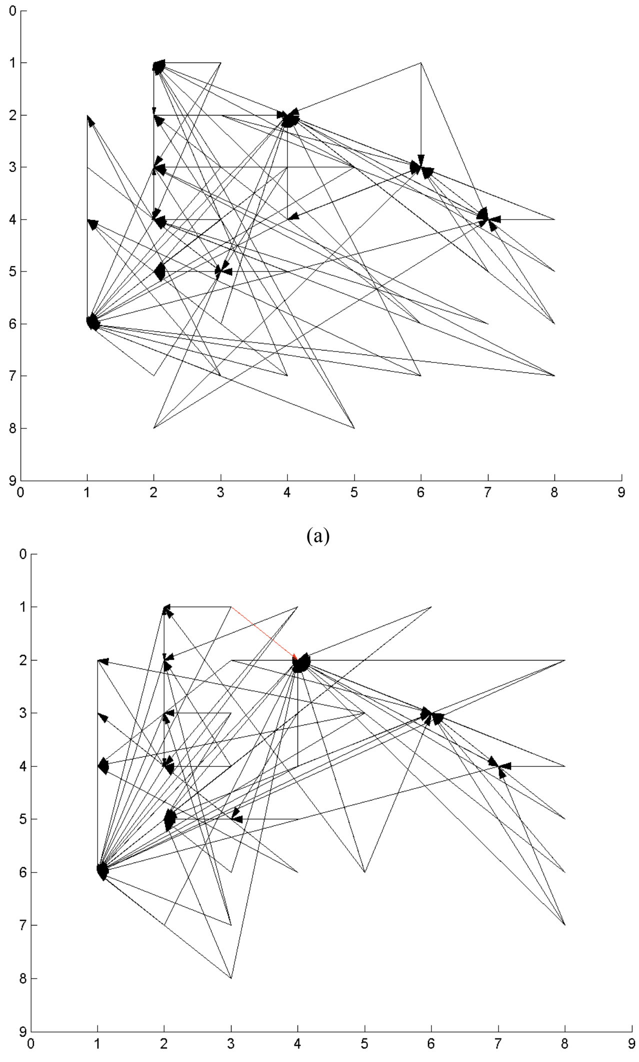
(a)

(b)

Figure 4. Interspike intervals on recordings from stimulated electrodes in the culture ID68 (22 DIV) from E2 before (a) and after (b) stimulation. Spiking periods have change after stimulation (b). ISI has decreased both in value and dispersion after stimulation.

Connectivity diagrams based on cross-correlation between electrodes showed some kind of connections reorganization after stimulations, concentrating them in a few electrodes. Furthermore, adjacent physical or logical connections in the connectivity graph following Hebb's law appeared in some pairs of stimulated electrodes (Figure 5).

Electrodes with created connections between them can distinctly be detected with the instantaneous firing frequencies graphs. Figure 3 showed two pair of stimulated electrodes (12, 21 and 38, 47) before and after the stimulation session. The firing periods of the electrodes from the second pair follow exactly each other, whereas the firing periods of the first pair of electrodes do not match. Furthermore, the electrodes of the second pair change both the firing periods after stimulation. This features indicates that there exits a strong connection between them.



(a)

(b)

Figure 5. Connectivity graphs based on cross-correlation between electrodes. The graph belong to the culture ID48 (E1) at 25 DIV. Pair of electrodes 31, 42 and 52, 53 were stimulated with low-frequency current stimulation with 50 μA biphasic pulses. (a) No logical connections were observed before stimulation. (b) A connection (red arrow) between electrodes 31 and 42 has appeared.

All of the cultures of E1 created a connection between the paired stimulated electrodes, whereas 60% of the cultures of E3 and only 20% of the cultures of E2 showed that connection. In some cases, the connection was intermittent, lasting one to several days. In others, a persistent connection was created. Finally, some cultures did not create any kind of connections. In this way, Hebbian tetanization created ad-hoc permanent or transient logical connections by modifying the efficiency of the paths between the selected electrodes. We speculate that the failed cultures may be caused by a not-homogeneous culture growth between the electrodes or by the neurobiological properties of the connections as will be confirmed using histological techniques in future works. In this case, using low-frequency current stimulation with 50 μA biphasic pulses provided the best results for creating connections following Hebb's law.

V. CONCLUSIONS

Learning in biological neural cultures is a challenging task. Different authors have proposed different methods for inducing a desired and controlled plasticity over the biological neural structure. Low-frequency stimulation has brought good results to researchers enhancing bursting activity in cortical cultures.

In this paper, we have shown that using this kind of stimulation it is possible to create adjacent physical or logical connections in the connectivity graph following Hebb's Law and such connections induce changes in the electrophysiological response of the cells in the culture, which can be observed in the different analysis performed. Furthermore, low-frequency stimulation induces changes using different values of current amplitude and stimulation time. Persistent and synchronous stimulation of relevant adjacent electrodes may be used for strengthen the efficiency of their connectivity graph. These processes may be used for imposing a desired behaviour over the network dynamics. In this work, a stimulation procedure is described in order to achieve the desired plasticity over the neural cultures, and shaping in this way the functional connectivity of the neural culture.

In future works, we will use different kind of electrical stimulations, such as tetanic stimulation, and try to find what are the optimal parameters of every stimulation that induce persistent changes in the cultured network. These induced connections will be used for driving a robot using Braitenberg's principles.

ACKNOWLEDGMENT

This work is being funded by grant 12361/FPI/09 from Séneca Foundation, from the region of Murcia and by the project 2010V/PUNED/0011 from UNED.

REFERENCES

[1] J.A. Anderson and E. Rosenfeld, "Neurocomputing: Foundations of research". MIT Press, Cambridge, 1988.

[2] The Facets Project. Accessed March 2013 from http://facets.kip.uni-heidelberg.de/

[3] E. Maeda, Y. Kuroda, H.P. Robinson and A. Kawana, "Modification of parallel activity elicited by propagating bursts in developing networks of rat cortical neurons". European Journal of Neuroscience, vol. 10, num. 2, 1998, p. 488-96.

[4] Y. Jimbo, H.P. Robinson and A. Kawana, "Strengthening of synchronized activity by tetanic stimulation in cortical cultures: application of planar electrode arrays". IEEE Transactions on Biomedical Engineering, vol. 45, num. 11, 1998, p. 1297-304.

[5] Y. Jimbo, T. Tateno and H.P.C. Robinson, "Simultaneous induction of pathway specific potetiation and depression in networks of cortical neurons". Biophysical Journal, vol. 76, num. 2, 1999, p. 670-678.

[6] T. Tateno and Y. Jimbo, "Activity-dependent enhancement in the reliability of correlated spike timings in cultured cortical neurons". Biological Cybernetics, vol. 80, num. 1, 1999, p. 45-55.

[7] G. Shahaf and S. Marom, "Learning in networks of cortical neurons". Journal of Neuroscience, vol. 21, num. 22, 2001, p. 8782- 8.

[8] M.E. Ruaro, P. Bonifazi, and V. Torre, "Toward the neurocomputer: image processing and pattern recognition with neuronal cultures". IEEE Transactions on Biomedical Engineering, vol. 52, num. 3, 2005, p. 371-83.

[9] D.A. Wagenaar, J. Pine, and S.M. Potter, "Searching for plasticity in dissociated cortical cultures on multi-electrode arrays". Journal of negative results in biomedicine, vol. 5, num. 16, 2006.

[10] R. Madhavan, Z.C. Chao, and S.M. Potter, "Plasticity of recurring spatiotemporal activity patterns in cortical networks". Physical Biology, vol. 4, num. 3, 2007, p. 181-93.

[11] Z.C. Chao, D.J. Bakkum, and S.M. Potter, "Region-specific network plasticity in simulated and living cortical networks: comparison of the center of activity trajectory (CAT) with other statistics". Journal of Neural Engineering, vol. 4, num. 3, 2007, p. 294-308.

[12] A.N. Ide, A. Andruska, M. Boehler, B.C. Wheeler, and G.J. Brewer, "Chronic network stimulation enhances evoked action potentials". Journal of Neural Engineering, vol. 7, num. 3, February 2010, 16008.

[13] L.L. Bologna, T. Nieus, M. Tedesco, M. Chiappalone, F. Benfenati, and S. Martinoia, "Low-frequency stimulation enhances burst activity in cortical cultures during development". Neuroscience, vol. 165, 2010.

[14] V. Lorente, J.M. Ferrández, F. de la Paz, and E. Fernández, "Training hippocampal cultures using low-frequency stimulation: Towards Hebbian Learning". 8th International Meeting on Substrate-Integrated Microelectrodes Arrays, 2012, pp. 90-91.

[15] J.M. Ferrández, V. Lorente, F. de la Paz, and E. Fernández, "Training Biological Neural Cultures: Towards Hebbian Learning". Neurocomputing, 2012.
http://dx.doi.org/10.1016/j.neucom.2012.09.031

[16] O. Sporns and G. Tononi G, "Classes of network connetivity and dynamics". Complexity, vol. 7, 2002, pp. 28–38.

[17] K.J. Friston, C. Frith, R. Frackowiak, "Time-dependent changes in effective connectivity measured with PET". Human Brain Mapping, vol. 1, 1993, pp. 69–79.

Power Consumption-based Application Classification and Malware Detection on Android Using Machine-Learning Techniques

Thomas Zefferer*, Peter Teufl*, David Derler*, Klaus Potzmader*
Alexander Oprisnik*, Hubert Gasparitz* and Andrea Hoeller*
*Insitute for Applied Information Processing and Communications, Graz University of Technology
{thomas.zefferer, peter.teufl}@iaik.tugraz.at
{dderler, klaus.potzmader, oprisnik, hubert.gasparitz, aoeller@student.tugraz.at}@student.tugraz.at

***Abstract*—Mobile computing has significantly gained importance during the past years and is expected to remain one of the most relevant future computing trends. Smartphones represent a key component of mobile computing based solutions and allow end users to conveniently access services and information. Due to their continuously growing importance and popularity, smartphones have recently become a common target for malware. Unfortunately, capabilities of malware-detection applications on smartphones are limited, as implemented security features such as sandboxing or fine-grained permission models restrict capabilities of third-party applications. These restrictions prevent malware-detection applications from accessing information, which is required to identify malware, and hence render the implementation of reliable malware-detection solutions on smartphones difficult. To overcome this issue, we propose an alternative malware-detection method for smartphones that relies on the smartphone's measured power consumption. We propose two different machine-learning techniques that allow for a classification of applications according to their power consumption and hence facilitate the identification of suspicious and potentially malicious software components. The capabilities of the proposed techniques have been assessed by means of an evaluation with real-world applications running on physical smartphones. The results of this evaluation process demonstrate the applicability of power consumption based classification and malware-detection approaches in general and of the two proposed machine-learning techniques in particular.**

***Keywords*—Android; power consumption; application classification; malware detection; machine learning**

I. Introduction

Powered by the emergence of smartphones, mobile computing has significantly gained popularity and relevance during the past years. Smartphones have become part of our daily life and have significantly changed the way we access information, communicate, and interact with each other. Considering current sales and usage statistics [1], it can be expected that mobile computing in general and smartphone based solutions in particular will continue to play a major role in future.

The recent success of popular smartphone platforms such as Apple iOS [2] or Google Android [3] has unfortunately turned these platforms into attractive targets for malware. Recent reports [4] show that smartphone malware must be expected to evolve to a major issue in mobile computing in future. By exploiting specific functionality provided by the infected smartphone platform, smartphone malware can cause financial losses by calling premium-rate numbers or by compromising smartphone based authentication schemes of e-banking solutions. During the past years, especially the Android platform has been frequently targeted by smartphone malware. A recent example is Eurograbber. In 2012, this Android malware has been used to steal 47 million USD from European bank accounts by intercepting SMS based authentication processes of e-banking portals [5]. Android seems to be especially prone to malware due to the platform's support of alternative application sources that usually lack extensive malware checks, and due to the broad functionality offered by Android's public APIs. These APIs grant application developers as well as attackers for instance full access to incoming and outgoing SMS messages, or facilitate the execution of arbitrary background tasks.

The factual vulnerability of the Android platform against malware raises the need for reliable methods to distinguish benign apps from malicious ones and to detect unwanted behavior on smartphones. In the desktop-computer domain, this functionality is typically implemented by anti-virus software, which is able to detect malicious software at runtime. Unfortunately, the deployment of anti-virus software on smartphone platforms in general, and on Android in particular is difficult. This is mainly due to the fact that Android (as well as other smartphone platforms) implements several security features on operating-system level that limit access rights and capabilities of third-party applications. For instance, all smartphone applications are executed in a sandbox and unable to access resources of other applications being installed and executed on the same device. While implemented security features definitely improve the system's basic security, they render the implementation of supplementary security software difficult. For instance, the implemented sandbox feature prevents anti-virus software on Android smartphones from collecting information that is required to reliably detect smartphone malware at runtime.

The integrated security features that limit the capabilities of classical malware-detection methods can theoretically be bypassed by rooting the smartphone's operating system. However, this is not really an option in practice, as it significantly decreases the smartphone's overall security and enables additional attack vectors.

The reliable detection of malware on non-rooted smartphones is still an unsolved problem that definitely needs to be addressed to assure the security of future mobile computing. To overcome this problem, we propose a new technique that compensates the lack of required information about running applications by making use of side-channel information being available on non-rooted Android smartphones. Concretely, our technique measures and analyses the power consumption of applications running on Android smartphones. We show that an application's power consumption correlates with its implemented functionality and that applications can hence be classified according to their power consumption. We further show that this classification can be used to identify malicious applications.

The remainder of this paper is structured as follows. In Section II, existing malware-detection approaches for smartphones are briefly surveyed and limitations of these approaches are identified. Subsequently, Section III introduces the PowerTutor tool [6] and explains our approach to measure the power consumption of applications on smartphones. In Section IV, we propose two methods to analyze collected power-consumption measurements based on approved machine-learning techniques. We evaluate the capabilities of the proposed methods to classify applications and to distinguish benign applications from malicious ones in Section V. Finally, conclusions are drawn in Section VI.

II. Related Work

During the past years, several approaches to detect and analyze malware on mobile platforms have been introduced. Basically, existing approaches can be classified into static and dynamic analysis methods. Static analysis refers to the inspection of an application's source code or binary package without running it, whereas dynamic analysis involves running the application to capture additional information.

Dynamic analysis includes techniques such as Information Flow Analysis, where private data is labeled and prevented from leaving the device. TaintDroid [7] is an Android kernel extension that follows this approach and allows for dynamic taint tracking. Dynamic approaches, which apply machine-learning techniques to distinguish benign applications from malicious ones, include [8] by Shabtai et al. and [**?**] by Burguera et al. Both references include extensive listings of related Android-based malware-detection systems. Most of these approaches run candidate applications in a sandbox to derive measurements, such as system-call intervals and networking usage.

A comprehensive overview of dynamic malware-analysis techniques is provided by Egele et al. in [9]. Many of these methods are highly advanced in detecting and analyzing malware. However, these methods usually require complex external analysis frameworks and can hardly be deployed on non-rooted end-user devices, due to their requirement to deeply integrate into the smartphone's operating system.

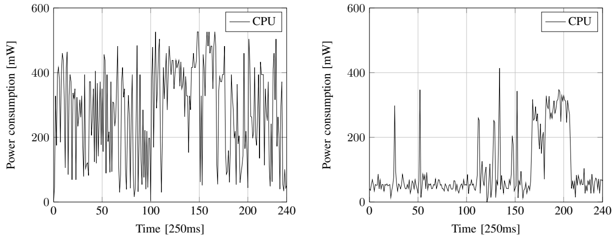
The technique presented in this paper addresses this problem and facilitates dynamic malware detection directly on non-rooted Android phones by analyzing the devices' power consumption. A related approach to analyze the power-consumption in order to detect malware has been followed by Jacoby and Davis [10], who have proposed an intrusion detection system that correlates various attack scenarios to typical power consumptions. Additional work has been published by Buennemeyer et al. [11] [12], who propose systems, which use power profiles of phones to detect malware targeting battery drainage. Our technique follows a similar approach but extracts more detailed information from the collected power-consumption measurements in order to classify applications and to detect malware.

Obviously, the collection of accurate power-consumption measurements on smartphones is a key aspect of our technique. We discuss details of this aspect in the next section.

III. Measuring the Power Consumption of Smartphones

Measurements of a smartphone's power consumption build the basis of the proposed classification and malware detection techniques. To collect the required power-consumption measurements, we rely on the PowerTutor tool by Zhang et al. [6]. Another tool that would allow for the acquisition of this kind of information is Trepn [13]. In contrast to PowerTutor, Trepn uses hardware sensors and thus promises more exact measurements. However, Trepn is limited to the Snapdragon mobile development platform [14] and can hence not be applied on typical Android based end-user devices.

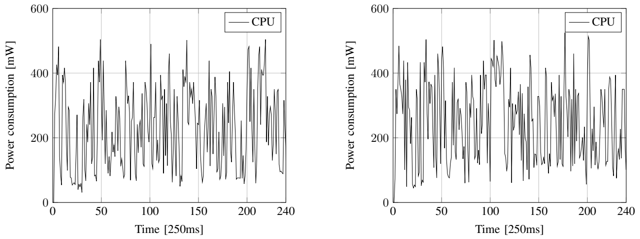
The PowerTutor tool is basically a smartphone application that measures the power-consumption of all applications running on the same smartphone. For each application, the power consumption of the six smartphone components *CPU*, *Audio*, *Display*, *Wi-Fi*, *3G* and *GPS* is measured separately. Figure 1 shows the measured power consumption of the CPU component caused by the two applications Android Browser and Lookout Mobile Security.



((a)) Android Browser ((b)) Lookout Mobile Security

Fig. 1: One minute plots of CPU power consumption

Although there are obvious differences in the power consumption of these two applications, an immediate identification and classification of applications based on such measurements is usually not possible. This is due to the fact that the measured power consumption is not only influenced by the application itself, but also by other effects, such as varying user inputs, the processed data, different screen orientations, the deployment of hardware acceleration techniques, or 3G or WiFi signal reception. Figure 2, which shows two different measurements of the



((a)) Angry Birds - Run 1 ((b)) Angry Birds - Run 2

Fig. 2: One minute plots of CPU energy consumption

same application, illustrates this fact. Although stemming from the same application, the two measured power-consumption traces are quite different.

Disturbing influences render the determination of a simple and unique power-consumption signature for a given application or smartphone state impossible. To overcome this problem, we propose two analysis techniques that rely on approved machine-learning approaches. The proposed techniques can be used to classify smartphone applications according to their power consumption. Details of the proposed techniques are discussed in the next section.

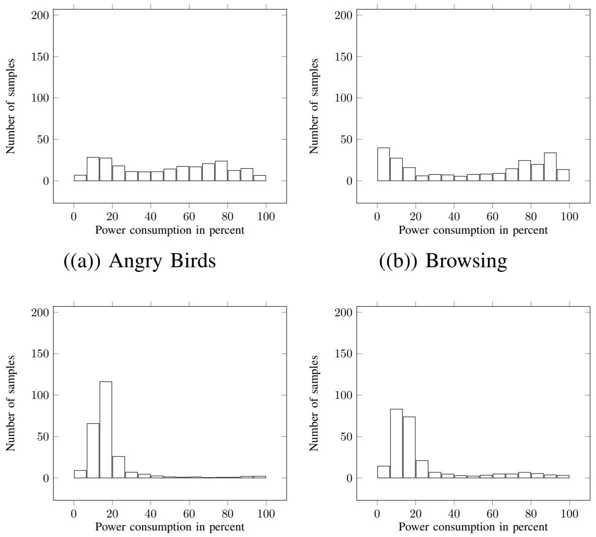
IV. Classification Techniques

During the past years, different machine-learning techniques for the classification of data have been proposed. For the given scenario, i.e., the classification of smartphone applications based on their power consumptions, two techniques have been chosen and adapted to the given requirements. Both techniques consist of a *learning phase* and a *classification phase*. During the learning phase, well-known input data is used to train a model. In the subsequent classification phase, the trained model is used to classify unknown input data. The two techniques are discussed in more detail in the following subsections.

A. Power-Consumption Histograms

This technique is rather simple and counts how often a specific application is on a certain power-consumption level. In order to model this, we have computed power histograms by dividing the interval between 0% power consumption and 100% power consumption into 15 disjoint and equal-sized intervals. A histogram is then created by simply assigning each data point to exactly one interval and counting the data points in each interval. In order to cope with differences in the absolute power consumption, the values have been normalized appropriately. During the learning phase, the average histograms have been created by measuring the power consumption of well-know applications. Figure 3 shows some examples of average histograms for different applications that have been obtained during the training phase.

In the classification phase, the histograms of applications to be classified are compared with the trained average histograms by applying distance-measures such as cosine similarity. To assess the capabilities of this approach, this technique has been evaluated in a real-world scenario. Results of this evaluation process are presented and discussed in Section V.



((a)) Angry Birds ((b)) Browsing

((c)) Idle ((d)) Lookout Mobile Sec.

Fig. 3: Average histograms of different applications

B. MFC Coefficients and Gaussian Mixture Models

This technique makes use of *Mel Frequency Cepstral Coefficients (MFCC)* to classify smartphone applications based on their power consumption. This technique has originally been introduced for speaker-recognition systems [15] [16] and is also frequently used for music similarity finders [17][18]. In such systems, MFC coefficients and their distribution are extracted from recorded voice or music using complex transformations as implemented by the *melcepst* function [19]. The distributions of the extracted MFCC are then used to create a *Gaussian Mixture Model (GMM)* for each MFCC. The resulting GMM define a unique representation of the recorded voice or music. Later recordings of voice or music can be compared to existing representations in order to implement voice-recognition and music-similarity finders.

Our intention behind using a speaker recognition approach was to map the problem of matching voice recordings to a person to the problem of matching power measurements to an application. Spoken voice recordings vary in pitch and frequency and are very unlikely to be equal between two recordings. This, naively speaking, resembles the problem we face with power-consumption measurements.

Our implementations are based on an existing speaker-recognition implementation by Anil Alexander [20]. This implementation relies on GMM and MFCC and can be customized with a number of parameters including the number of Gaussians and the number of MFCC to use. Experiments have shown that for our purposes best results can be achieved with three Gaussians and twelve MFCC. Hence, during the learning phase the distributions of twelve MFCC are computed from power-consumption measurements for each class of application. The computed distributions of the twelve MFCC are then approximated using a GMM with three Gaussians. The resulting GMM finally represents the result of the learning phase. Figure 4 illustrates the distribution of two different MFCC and the resulting GMM.

During the classification phase, MFCC are derived from power-consumption measurements of the application to be classified. For each derived MFCC, the best matching GMM is selected out of all GMM that have been obtained during the learning phase. By combining the classification results of all twelve MFCC, the best matching application class is finally determined.

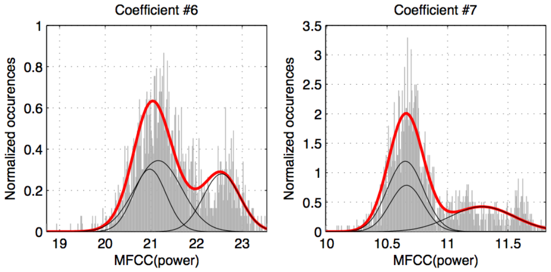


Fig. 4: Gaussian Mixture Models of two MFCC derived from power-consumption measurements

V. Evaluation

We have evaluated the reliability and efficiency of the proposed feature extraction techniques by testing prototype implementations of the two techniques in a real-world scenario. Required power-consumption measurements have been acquired using the PowerTutor tool. For convenience reasons, the classification itself has been performed off the mobile device, as the learning phase (especially for the speaker recognition based approach) is rather slow. This subsection describes the model that has been used to classify applications, discusses details of the dataset creation, and presents results that have been obtained by applying the two classification techniques introduced in Section IV.

A. Classification Model

Applications with the same or almost the same purpose are expected to cause similar power consumptions. Therefore, we have roughly grouped applications into distinct sets according to their purpose. The resulting list of groups is no comprehensive classification scheme of all available applications. It is merely a logical grouping of the power-consumption measurements we gathered in this experiment and does raise no claim to completeness. Based on the gathered measurements, the following six groups of applications have been defined: *Games*, *Internet*, *Idle*, *Malware*, *Music*, and *Multimedia*. Note that malware and security software have been assigned to the same group. Both malware and security software usually remain idle in the background until being activated by a certain event (e.g., reception of a command message via SMS). This comparable behavior leads to a comparable power consumption too and justifies a common classification of these both types of application.

B. Dataset Creation

PowerTutor provides specific measurements for each running application. However, in practice these application specific measurements have turned out to be not as reliable and accurate as desired. Therefore, we have refrained from using application specific measurements and have relied on system-wide power-consumption measurements provided by PowerTutor instead.

We have further limited subsequent analysis steps to the measured power consumption of the smartphone's CPU. Although PowerTutor also provides measurements for other smartphone components such as the display or the GPS receiver, measurements of these components have been omitted in order to reduce computation costs when learning and due to the fact that these components often lack activity.

To evaluate the proposed classification techniques, we finally created 96 system-wide power-consumption measurements (CPU) using a customized instance of PowerTutor. To facilitate a subsequent analysis, we have adapted PowerTutor such that beside the measurement values themselves also the device model, the capture date, and the sample rate have been stored. The 96 captured measurements (sixteen measurements per application group) have been limited to the length of about one minute, with a total of 247 data points per measurement. We have cut off the trace length after about one minute, as this is a realistic time-frame for real-world scenarios. In total, six devices have been used to collect the measurements (three Samsung Galaxy S2 smartphones and three HTC Desire devices). To reduce noise, only the application to be measured and PowerTutor have been active during the measurements.

C. Results

The 96 captured measurements have been used to evaluate the efficiency and reliability of the proposed classification techniques. As quality indicators, the positive predictive value (PPV, also referred to as precision), the true positive rate (TPR, also referred to as recall or sensitivity), the true negative rate (TNR, also referred to as specifity), the accuracy, and the area under the receiver operating characteristic (AUC) have been used. According to its definition, PPV refers to the correct positive classification in relation to all positive classifications. Accordingly, TPR refers to true positives given all real positives. TNR denotes true negatives (TN) given all negatives. Accuracy is the relation between correctly classified samples given all samples. The receiver operating characteristic is a graphical representation of the trade-off between TPR and FPR (1-TNR). AUC (also sometimes denoted as AUROC) refers to the area below this resulting curve.

TABLE I: HISTOGRAM BASED APPROACH: CONFUSION MATRIX FOR CATEGORIES GAMES (G), INTERNET (IN), IDLE (ID), MALWARE (MW), MUSIC (MU), AND MULTIMEDIA (MM)

	G	IN	ID	MW	MU	MM
G	13.98	1	0	1.02	0	0
IN	1	13.14	0	0	0	1.86
ID	0	0	12	4	0	0
MW	0	0	3.97	10.07	1.96	0
MU	0	0	0	2	9.24	4.76
MM	0.27	0.75	0	0	0	14.98

In order to appropriately divide the available measurements in training and test data, we have folded the available dataset using 10-fold cross validation. To enhance the robustness of the obtained results, average values over 100 runs are presented.

TABLE II: CLASSIFICATION RESULTS (HISTOGRAM BASED APPROACH)

Category	PPV	TPR	TNR	Accuracy	AUC
Games	0.87	0.92	0.98	0.97	0.95
Internet	0.82	0.88	0.98	0.95	0.93
Idle	0.75	0.75	0.95	0.92	0.85
Malware	0.63	0.59	0.91	0.87	0.75
Music	0.58	0.83	0.98	0.91	0.90
Multimedia	0.94	0.69	0.92	0.92	0.80

For our performance evaluation, a confusion matrix for the six predefined application categories has been created, which can be interpreted in the following way: Values in the diagonal of the matrix have been classified correctly (true positives), values within a row not in the diagonal represent false negatives and values within a column not in the diagonal represent false positives. Other values are considered true negatives.

TABLE III: MFCC AND GMM BASED APPROACH: CONFUSION MATRIX FOR CATEGORIES GAMES (G), INTERNET (IN), IDLE (ID), MALWARE (MW), MUSIC (MU), AND MULTIMEDIA (MM)

	G	IN	ID	MW	MU	MM
G	10.40	3.46	0.01	0	0.55	1.58
IN	2.07	12.67	0	0	0.26	1
ID	0.39	0	11.65	3.6	0.18	0.18
MW	0.07	0.01	2.56	13.15	0	0.21
MU	0.22	0.81	0	0.97	9.39	4.61
MM	0.96	2.97	0.01	0.03	1.20	10.83

TABLE IV: CLASSIFICATION RESULTS (GMM BASED APPROACH)

Category	PPV	TPR	TNR	Accuracy	AUC
Games	0.65	0.74	0.95	0.90	0.85
Internet	0.79	0.64	0.91	0.89	0.78
Idle	0.73	0.82	0.97	0.93	0.90
Malware	0.82	0.75	0.94	0.92	0.85
Music	0.59	0.82	0.97	0.91	0.90
Multimedia	0.68	0.59	0.91	0.87	0.75

Obtained results of the histogram based approach are shown in Table I and Table II. In case of the MFCC based approach, best results have been achieved with 3 Gaussians and twelve MFCC. The performance evaluation results of the MFCC based approach are outlined in Table III and Table IV.

D. Discussion

From these results, various findings can be derived. Mobile security applications and malware running in the background can generally be distinguished from application being active at the moment (with the exception of system services). Games, Internet, music and multimedia applications are distinguishable as well. Music and multimedia applications are more difficult to distinguish correctly. However, given their relating purposes this is plausible. Streaming a YouTube video with sound is not too different from listening to music while reading related information displayed by the music player.

The obtained results have also revealed that the MFCC based approach works better for the distinction between the categories Idle and Malware. Therefore, this approach seems to be more suitable for malware-detection purposes. On the other hand, the histogram approach constitutes a fast classification method, suitable for mobile devices with limited computational power.

VI. Conclusion and Future Work

Malware on smartphones is a growing issue and a major challenge for future mobile computing solutions. To overcome this challenge, new and innovative methods to detect malware on smartphones are needed. In this paper, we have tested the hypothesis that the power consumption of smartphones correlates with the kind of applications being executed on the smartphone and that this correlation allows for a classification of applications and a detection of malicious software. To test this hypothesis, we have proposed two machine-learning techniques that can be used to classify unknown applications according to their power consumption. We have further assessed the validity of the general hypothesis and the capabilities of the proposed machine-learning techniques by means of a concrete prototype implementation and a succeeding evaluation in a real-world scenario. The conducted assessment has corroborated the constructed hypothesis and has shown the capabilities of the proposed techniques to correctly classify smartphone applications according to their power consumptions.

Although first results are promising, this work mainly represents a proof of concept and a solid basis for future work. In a next step, we plan to port the entire classification onto a smartphone in order to render external classification frameworks unnecessary. Power measurements can already be collected directly on the smartphone using tools such as PowerTutor. Since information on the smartphone's power consumption is publicly available on Android smartphones, our solution does not require root access to the operating system and is hence applicable on virtually all end-user devices. We are also planning to refine the proposed techniques and to enhance the current prototype in order to achieve even more accurate results and to be able to classify multiple applications running simultaneously on a smartphone.

References

[1] GO-Gulf.com, "Smartphone Users Around the World - Statistics and Facts," 2013. [Online]. Available: http://www.go-gulf.com/blog/smartphone/

[2] Apple, "Apple iOS 6," 2013. [Online]. Available: http://www.apple.com/ios/

[3] Google, "Android," 2013. [Online]. Available: http://www.android.com/

[4] Lookout Mobile Security, "2011 Mobile Threat Report," Tech. Rep. [Online]. Available: https://www.mylookout.com/mobile-threat-report

[5] InformationWeekSecurity, "Zeus Botnet Eurograbber Steals $47 Million," 2012. [Online]. Available: http://www.informationweek.com/security/attacks/zeus-botnet-eurograbber-steals-47-millio/240143837

[6] L. Z. L. Zhang, B. Tiwana, R. P. Dick, Z. Q. Z. Qian, Z. M. Mao, Z. W. Z. Wang, and L. Y. L. Yang, "Accurate online power estimation and automatic battery behavior based power model generation for smartphones," p. 105, 2010. [Online]. Available: http://dl.acm.org/citation.cfm?id=1878961.1878982

[7] W. Enck, L. P. Cox, P. Gilbert, and P. Mcdaniel, "TaintDroid : An Information-Flow Tracking System for Realtime Privacy Monitoring on Smartphones," *Design*, pp. 1–6, 2010. [Online]. Available: http://www.usenix.org/event/osdi10/tech/full_papers/Enck.pdf

[8] A. Shabtai, U. Kanonov, Y. Elovici, C. Glezer, and Y. Weiss, “Andromalyâ: a behavioral malware detection framework for android devices,” *Journal of Intelligent Information Systems*, vol. 38, no. 1, pp. 161–190, 2011. [Online]. Available: http://www.springerlink.com/index/10.1007/s10844-010-0148-x

[9] M. Egele, T. Scholte, E. Kirda, and C. Kruegel, “A survey on automated dynamic malware-analysis techniques and tools,” *ACM Computing Surveys*, vol. 44, no. 2, pp. 1–42, 2012. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2089125.2089126

[10] G. A. Jacoby, R. Marchany, and N. J. I. Davis, “Battery-based intrusion detection a first line of defense,” 2004. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1437827

[11] T. K. Buennemeyer, T. M. Nelson, L. M. Clagett, J. P. Dunning, R. C. Marchany, and J. G. Tront, “Mobile Device Profiling and Intrusion Detection Using Smart Batteries,” pp. 296–296, 2008. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4439001

[12] H. Kim, J. Smith, and K. G. Shin, “Detecting energy-greedy anomalies and mobile malware variants,” *Power*, pp. 239–252, 2008. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1378600.1378627

[13] Liat Ben-Zur, “Developer Tool Spotlight - Using Trepn Profiler for Power-Efficient Apps,” 2011. [Online]. Available: https://developer.qualcomm.com/blog/developer-tool-spotlight-using-trepn-profiler-power-efficient-apps

[14] Bsquare, “Snapdragon Based Products and Services,” 2013. [Online]. Available: http://www.bsquare.com/products/snapdragon-based-products-and-services

[15] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, “Speaker Verification Using Adapted Gaussian Mixture Models,” *Digital Signal Processing*, vol. 10, no. 1-3, pp. 19–41, Jan. 2000. [Online]. Available: http://dx.doi.org/10.1006/dspr.1999.0361

[16] G. Kumar, K.A.Prasad Raju, Dr.Mohan Rao CPVNJ, and P.Satheesh, “Speaker Recognition Using GMM.” [Online]. Available: http://core.kmi.open.ac.uk/display/910430

[17] B. Logan and A. Salomon, “A music similarity function based on signal analysis,” in *IEEE International Conference on Multimedia and Expo, 2001. ICME 2001.* IEEE, 2001, pp. 745–748. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1237829

[18] K. West and P. Lamere, “A Model-Based Approach to Constructing Music Similarity Functions,” *EURASIP Journal on Advances in Signal Processing*, vol. 2007, no. 1, p. 024602, Jan. 2007. [Online]. Available: http://dl.acm.org/citation.cfm?id=1288980.1289110

[19] Mike Brooks, “VOICEBOX: Speech Processing Toolbox for MATLAB.” [Online]. Available: http://www.ee.ic.ac.uk/hp/staff/dmb/voicebox/voicebox.html

[20] Alexander Anil, “Automatic Speaker Recognition: A Simple Demonstration using Matlab,” 2004. [Online]. Available: http://www.anilalexander.org/publications/Speaker_Recognition_Intro_A-A.pdf

P-SQLITE: PRAM-Based Mobile DBMS for Write Performance Enhancement

Woong Choi, Sung Kyu Park, Seong Min Kim, Min Kyu Maeng, †Ki-Woong Park, and Kyu Ho Park
Department of Electrical Engineering
Korea Advanced Institute of Science and Technology (KAIST)
†Daejeon University
Daejeon, Korea
Email: {wchoi, skpark, smkim, mkmaeng}@core.kaist.ac.kr, †woongbak@dju.kr, kpark@ee.kaist.ac.kr

***Abstract*—Recent advances in mobile services have led to ever-increasing demands for high performance mobile database due to its data-centric characteristics. However, NAND flash memory, which is the main storage medium of mobile device has drawbacks in write operation because of its erase-before-write characteristics. In an attempt to overcome the drawbacks of the NAND flash memory, we propose a new mobile database management system (MDBMS), called P-SQLITE. It is enhanced with the find-grained data management techniques for using phage change random access memory (PRAM) as a caching layer between main memory and NAND flash memory. P-SQLITE continuously monitors the write patterns to identify hot chunk which is fine-grained file access unit and migrates the hot chunks from NAND flash memory to the PRAM. The experimental results show that the P-SQLITE improves the write performance by 47% than the previous MDBMS.**

***Keywords*-MDBMS; NAND flash memory; PRAM**

I. Introduction

In current mobile devices, NAND flash memory has led the mobile device market as a main storage system due to its portability, large capacity, and performance of read operation. Even though it has outstanding features for mobile environment, it has two serious limitations. One of the problem is read and write asymmetry where the write operation is almost 10 times slower than the read operation. If the application running on the mobile device is write-intensive, the performance of NAND flash memory is degraded itself. Also, garbage collection overhead is another drawback of it. Because NAND flash memory cannot execute in-place update which updates previously saved data in the same physical location of its cell, it invalidates the original data in a previous location and stores updated data in a new location. When the occupation of invalidated data in NAND flash memory is increased in some degree, garbage collection is performed to reclaim the invalidated data. Since the garbage collection causes additional write and erase operations, performance degradation occurs.

In order to overcome these drawbacks, non-volatile random access memories (NVRAM) such as phase change RAM (PRAM) [1], ferroelectric RAM (FRAM) [2], and magnetoresistive RAM (MRAM) [3] are widely employed as a candidate for the storage system of the mobile device to replace NAND flash memory. Among these memories, PRAM is the closest to being on the market. One of the attractive points of PRAM compared to NAND flash memory is its speed of write operation. Additionally, it has in-place update characteristics where NAND flash memory doesn't offer. However, PRAM has a capacity problem due to the limitation of the current technology. To overcome this problem, hybrid approach, which is the combination of NAND flash memory and PRAM is studied currently in the storage area [4], [5]. From this, it is possible to achieve the benefits of NAND flash memory (large capacity, fast read speed) and PRAM (fast write speed, in-place update).

Concurrently, mobile database management system (MDBMS) has been developed as a software data management system of the mobile device. Especially, SQLite [6] is widely used as a database to manage the data of mobile applications in the Android [7] platform which is the famous mobile Operating System (OS) released by Google. The performance of SQLite is bounded by that of storage because it stores the whole database in the host file system as a single file [8]. It means that the execution time of mobile application is closely related to the degradation of NAND flash memory performance due to the write performance.

In this paper, we propose a new mobile database management system, called P-SQLITE to enhance the write performance of MDBMS by applying hybrid architecture composed of NAND flash memory and PRAM. P-SQLITE detects the fine-grained hotness (write count) of MDBMS chunks which are fine-grained file access units and transfers them to PRAM. Also, it contains chunk migration considering the limited capacity of PRAM so that more recent written chunks are placed in the PRAM.

The remaining part of this paper is organized as follows. Section II introduces the previous work to enhance MDBMS with various methods. In Section III, overall architecture of P-SQLITE which is our proposed MDBMS are explained. Detailed descriptions of its modules and functionalities are discussed in Section IV. Evaluation for them are summarized in Section V while the conclusion and future work are given in Section VI.

ISBN: 978-1-61208-272-1

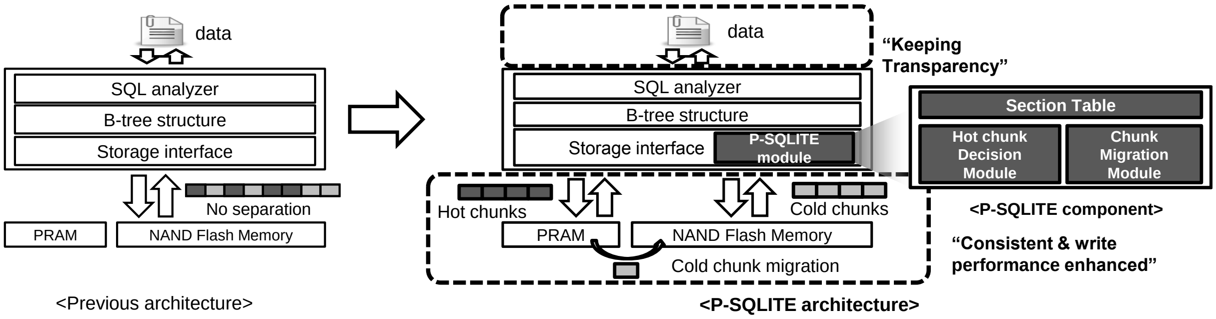


Figure 1: Overall Architecture of P-SQLITE compared with Previous MDBMS Architecture

II. Related Work

There are several researches to leverage the write performance of the file system equipped with NAND flash memory. As a recent study of software support, Yanfei Lv et al. [9] proposed operation-aware buffer management in the flash-based system to reduce the cost of read/write performance of NAND flash memory. They suggest Flash-based Operation-aware buffer Replacement (FOR) algorithm which considers both asymmetry of read/write speed and operation wise statistics to achieve high performance. However, it fails to support consistency of the dirty data in the buffer in case of unexpected power failure which brings the loss of valuable and necessary user data in the mobile devices.

PFFS2 [10] suggested PRAM as a metadata storage to reduce garbage collection overhead and write operations of NAND flash memory in the file system layer. Metadata is the data which indicates the fundamental information data of the file. In this research, they use virtual metadata storage which enables the migration when there is excess of metadata in PRAM. However, it is not appropriate to apply this method to improve the write performance of MDBMS since it handles a database as a single file.

Also, there were some efforts to leverage write performance on MDBMS. Y. park et al. [11] studied the effect of rollback journal files of MDBMS. Rollback journal files are back-up files to prevent unexpected loss of power during the modification. They decrease the write latency of the journal files and the number of garbage collections in NAND flash memory by storing them into the PRAM. Even if they increase the performance of MDBMS, it cannot handle the excess of the journal files due to various mobile databases since they assume the sufficient size of PRAM.

By analyzing the relationship between previous works and MDBMS, some important criteria should be guaranteed to improve its performance. First of all, gap between the hybrid storage architecture and current MDBMS have to be solved. Because current MDBMS is mismatched with the hybrid storage architecture in respect of data storing, this bottleneck should be solved to increase the write performance. Finally, the limited capacity of PRAM should be considered. Without the practical restriction of PRAM size, it is hard to assure improvement of write performance where PRAM is full.

III. P-SQLITE Overview

In this section, the overall design of P-SQLITE and its process sequence are described. It contains P-SQLITE module to leverage the write performance of the storage. Even when it needs additional operations compared to previous MDBMS to achieve fine-grained management of data, we improve the write performance of MDBMS through P-SQLITE.

Figure 1 shows the overall architecture of P-SQLITE compared with previous MDBMS architecture. In the case of previous MDBMS, data is firstly inserted as structured query language (SQL). After that, the SQL analyzer parses and figures out the SQL. Then, the data with the SQL is stored in the form of a node or nodes of B-tree and stored in a file through the storage interface. However, previous MDBMS doesn't use the database chunk as a unit of data management where it is a node of B-tree and fundamental structure of MDBMS. Thus, this architecture considers neither the write frequency of each chunk of the file nor hybrid storage system to enhance the write performance.

On the other hand, P-SQLITE enables fine-grained hot-cold separation and migration of chunks on the hybrid architecture composed of PRAM and NAND flash memory. To do this, it contains new modules in the storage interface. The P-SQLITE module is composed of three main parts which are *section table, hot chunk decision module, and chunk migration module*. *Section table* records the write count of each chunk periodically. Based on this information, *hot chunk decision module* decides whether chunk is allocated in PRAM or not. The module puts marks for selected chunks on section table, not placing them at that moment. When actual writing is requested for the marked chunks again,the chunk actually is stored in NVRAM at that moment. This lazy policy reduces unnecessary migration of a chunk from NAND flash memory to NVRAM. Lastly, *chunk migration module* migrates relatively colder chunks from PRAM to NAND flash memory to cover the limited capacity of PRAM.

TABLE I. STRUCTURE OF SECTION TABLE

Chunk#	State	Device	WriteCount	EWMA
0	USED	PRAM	5	4.93
1	CHANGE	NAND	2	3.18
2	USED	NAND	4	2.09
3	USED	NAND	1	0.67
4	UNUSED	-	-	-
...	...	...	...	...

IV. MAIN MODULES OF P-SQLITE

In this section, main modules of P-SQLITE are presented. P-SQLITE consists of three components which are a section table, hot chunk decision module, and chunk migration module. We provide the detailed design and implementations of each component in following sub-sections.

A. *Section Table*

Section table is designed to handle the data with fine-grained chunk unit rather than file unit which are relatively coarse-grained. To manage the data with chunk unit, write count and location of each chunk should be recorded. Table I shows the structure of the section table. It has four indexes which are state, device location, write count in current transaction period and exponentially weighted moving average (EWMA) [12] for each chunk. The state column represents whether this chunk is used (USED), not used yet (UNUSED), or ready to change location into other devices (CHANGE). In this case, we added CHANGE state to apply the lazy policy which is explained in previous section. Secondly, the device column indicates whether each chunk is allocated in PRAM or NAND flash memory. The write count and EWMA columns are used to decide how corresponding chunks are frequently and recently written. Detail explanation for above two indexes are introduced in the next section. While implementing section table, it occupies two bytes in memory space per chunk which has size of two kilobytes. Since the space for section table is only a thousandth portion of original data, its overhead is negligible.

B. *Hot Chunk Decision Module*

The hot chunk decision process consists of new write count expectation based on write count history and total performance gain calculation based on write count expectation. First of all, defining the transaction period is needed to implement this decision module.

1) Transaction Period: Decision of period is important since it affects the reflection degree of recent tendency. To reflect recent tendency of write count, P-SQLITE updates write count of each chunk in every period. Since the database executes write operation in a transaction unit, we set the period as several execution of write transactions. We call this number of transactions as *transaction period*. If the transaction period is too short, high prediction miss is expected since number of sample data is too small. Furthermore, a short transaction period incurs frequent write count prediction process which causes performance overhead. On the other hand, performance is degraded by unnecessary migration because most of chunks are considered as recent data if the transaction period is too long.

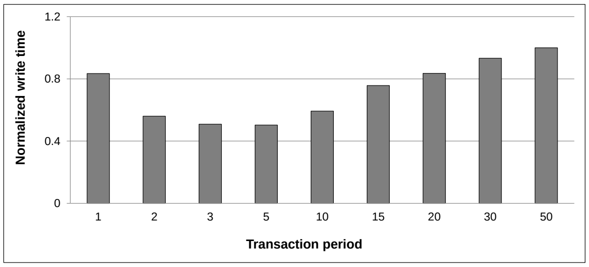


Figure 2: Normalized Write Time per Transaction Period

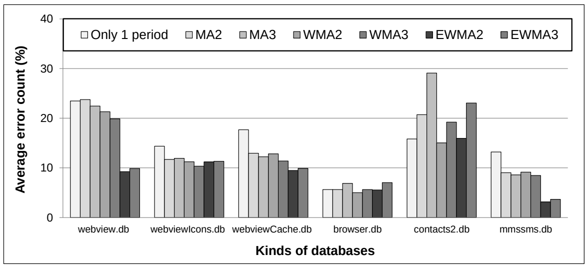


Figure 3: Avg. Error Rate during Various Transaction Periods

To verify the effect of transaction period and determine an appropriate value, we conduct experiment with six databases which are associated with web browser, message and phone book application with 1000 write transactions. Also, simple prediction policy which considers chunk as hot if write count of that chunk is over half of transaction count in a period are adopted to calculate migration penalty. As shown in Figure 2, there is relationship between transaction period and write performance as we mentioned. From the experimental result, we set the five transactions as a transaction period which gives the shortest write time.

2) Write Count Expectation: P-SQLITE predicts the write count in the future based on the previous write count of each chunk. Thus, prediction failure results in designating cold chunks as hot chunks and vice versa. In order to reduce an error rate in history based write count expectation, accurate expectation method is needed. Therefore, we evaluated various history-based prediction methods such as moving average (MA), weighted moving average (WMA), and exponentially weighted moving average (EWMA) with several periods.

To find out the best method which give least error, we did experiment with the three different methods and various periods. For this experiment, same applications which are mentioned in the previous section are adopted. As shown in Figure 3, EWMA with two periods gives least error value which is 9% on average and 15% at most for the various applications. Therefore, we selected this EWMA method to find out the expectation value of a next transaction period.

$$n^*_{t+1} = \alpha n_t + (1-\alpha)n^*_t$$
$$(\alpha = 2/N + 1)$$

,where (1)

n^*_{t+1}: EWMA of next period,
n_t: write count of current period,
n^*_t: EWMA of current period,
N: transaction period size.

The EWMA of next period is calculated by (1). P-SQLITE calculates the new EWMA for next period using current write count and EWMA values. The newly calculated EWMA value at the end of a transaction period is stored in the EWMA column for next calculation again.

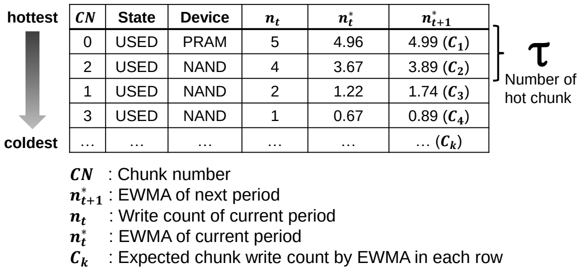
3) Gain Calculation for Hot Chunk Decision: To select the set of the hot chunks which will bring the highest performance gain based on the calculated EWMA, P-SQLITE calculates the effective performance gain of each chunk. This calculation mainly consists of two parts: reordering the Section Table by new EWMA value of next period and calculating the gain by each row. Firstly, Reordering section table arranges the table in descending order to facilitate hot chunks selection. Then, P-SQLITE consider both performance gain by allocating the chunks in PRAM and performance loss by chunk migration from PRAM to NAND flash memory due to the PRAM capacity limitation.

As the first step of hot chunk decision, recorded section table is sorted by calculated EWMA value in descending order except for unused chunks as shown in Figure 4. In the figure, τ represents the number of hot chunks which are selected from the top of the table. P-SQLITE regard τ as the meaningful threshold of hot chunks.

$$TotalGain(\tau)$$
$$= Gain_{PRAM}(\tau) - Loss_{Migration}(\tau)$$

, where (2)

$Gain_{PRAM}(\tau)$: performance gain by alloting chunks in PRAM,
$Loss_{Migration}(\tau)$: performance loss by chunk migration from PRAM to NAND.



CN	State	Device	n_t	n^*_t	n^*_{t+1}
0	USED	PRAM	5	4.96	4.99 (C_1)
2	USED	NAND	4	3.67	3.89 (C_2)
1	USED	NAND	2	1.22	1.74 (C_3)
3	USED	NAND	1	0.67	0.89 (C_4)
…	…	…	…	…	… (C_k)

CN : Chunk number
n^*_{t+1} : EWMA of next period
n_t : Write count of current period
n^*_t : EWMA of current period
C_k : Expected chunk write count by EWMA in each row

Figure 4: Reordered Section Table according to New EWMA

After that, total performance gain is calculated as equation of τ. As shown in (2), the equation consists of two terms which are performance gain by allotting the numbers of chunks from the hottest order in PRAM and performance loss by chunks migrated from PRAM to NAND flash memory.

Finally, the detailed formula is provided in (3). First of all, the performance gain part is calculated by multiplication of the benefit by using PRAM per write count of a chunk and expected write count of the chunks set by τ. The benefit by using PRAM per a chunk write count is defined as the difference between write speed of PRAM and NAND flash memory. Expected write count is acquired by EWMA value in the table. As the number of chunk write increases, the performance gain also increases. Secondly, the performance loss is obtained by multiplication of the migration overhead per chunk and the number of chunk which will be migrated to NAND flash memory. If PRAM is full and new chunk is allocated in PRAM, relatively colder chunks are migrated to NAND flash memory in P-SQLITE. At this time, migration overhead is defined as the summation of read speed of PRAM and write speed of NAND flash memory. The more migrated chunks P-SQLITE has, the more migration loss is generated.

$$TotalGain(\tau)$$
$$= (NAND_{WS} - PRAM_{WS})\sum_{i=1}^{\tau} C_i$$
$$-(PRAM_{RS} + NAND_{WS}) \times Chunk_{MN}(\tau)$$

, where (3)

$NAND_{WS}$: Write speed of NAND,
$PRAM_{WS}$: Write speed of PRAM,
C_i: Expected chunk write count by new EWMA in each row,
$PRAM_{RS}$: Read speed of PRAM,
$Chunk_{MN}$: Chunks which will migrate to NAND by τ.

C. Chunk Migration Module

If PRAM is full and new chunks are about to migrate into PRAM, migration of relatively colder chunk need to occur to accept a new hotter chunk as shown in Figure 5. In this case, P-SQLITE selects least recently used (LRU) as victim selection algorithm. To apply the strategy, finding the recently used chunk is essential. P-SQLITE checks the section table which has write count and EWMA in the

ISBN: 978-1-61208-272-1

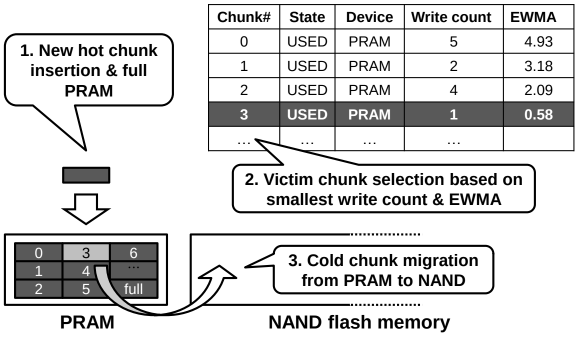

1. New hot chunk insertion & full PRAM
Chunk# | State | Device | Write count | EWMA
0 | USED | PRAM | 5 | 4.93
1 | USED | PRAM | 2 | 3.18
2 | USED | PRAM | 4 | 2.09
3 | USED | PRAM | 1 | 0.58
… | … | … | …
2. Victim chunk selection based on smallest write count & EWMA
0 3 6
1 4 …
2 5 full
3. Cold chunk migration from PRAM to NAND
PRAM
NAND flash memory

Figure 5: Sequence of Chunk Migration

current period to find out LRU chunk in PRAM. Based on these two values, it figures out the victim chunk in PRAM. Small write count and EWMA value of the chunk means that it is not recently used. P-SQLITE firstly selects the chunk which has smaller write count and secondly compares EWMA if there are chunks which have same write counts.

D. Overall Process Sequence of P-SQLITE

To find out hot chunks which are frequently updated in a database file, P-SQLITE firstly record the write count of each chunk per write transaction. Figure 6 shows the recording and assigning process of P-SQLITE. For every write transaction, write count of each chunk is written in the section table. After that, hot chunk decision module evaluates chunks which will be assigned in PRAM based on the section table. In this moment, it inserts mark for selected chunk in the section table, not placing them into PRAM immediately. Those chunks are stored in PRAM only after the actual writing command is requested for them again. This lazy policy reduces unnecessary migration of chunks from NAND flash memory to PRAM to prevent extravagant usage of PRAM capacity.

At the moment of actual writing, it is hard to expect the write performance enhancement by PRAM when it is full and recent hotter chunks are stored in NAND flash memory. To solve this problem, chunk migration module moves relatively colder chunks from PRAM to NAND flash memory.

V. Evaluation

In this section, experimental environment and results are explained. Firstly, the specification of hardware components, software layers, and workload characteristics are introduced. After that, the experimental results for the effect of P-SQLITE compared to previous work are given.

A. Experimental Setup

We developed P-SQLITE in HBE-EMPOS3 SV210 board [13] made by Hanback electronics. It has 800Mhz ARM Conrtex-A8 CPU, 512Mbyte of DDR2 SDRAM and 256Mbyte SLC NAND flash memory. In case of software

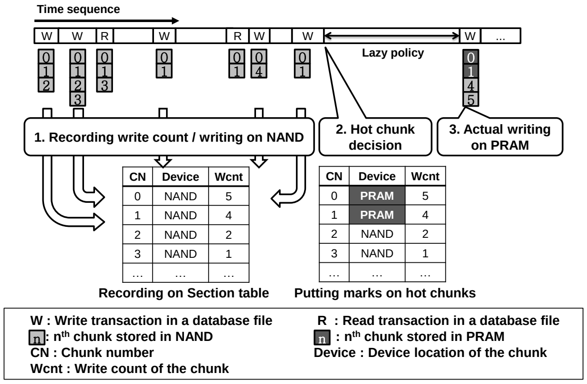

Time sequence
W
W
R
W
R
W
W
W
…
Lazy policy
0 1 2
0 1 2 3
0 1 3
0 1
0 1
0 4
0 1
0 1 4 5
1. Recording write count / writing on NAND
2. Hot chunk decision
3. Actual writing on PRAM
CN | Device | Wcnt
0 | NAND | 5
1 | NAND | 4
2 | NAND | 2
3 | NAND | 1
… | … | …
Recording on Section table
CN | Device | Wcnt
0 | PRAM | 5
1 | PRAM | 4
2 | NAND | 2
3 | NAND | 1
… | … | …
Putting marks on hot chunks
W : Write transaction in a database file
R : Read transaction in a database file
n: n^th chunk stored in NAND
n : n^th chunk stored in PRAM
CN : Chunk number
Device : Device location of the chunk
Wcnt : Write count of the chunk

Figure 6: Process Sequence of P-SQLITE

base, Android 2.2 proyo [7] and Linux Kernel 2.6.32 were used as the main operating systems in the P-SQLITE. Our P-SQLITE is implemented based on the fundamental MDBMS, SQLITE [6] which is basically installed in the Android platform to manage mobile applications. We evaluated P-SQLITE with databases of four practical applications which are web browser (webview.db, webviewcache.db, webviewIcons.db), text message (mmssms.db), phone book (contacts2.db), and alarm (alarms.db).

We selected PRAM as an NVRAM of P-SQLITE where it is widely discussed in the storage research area and it is the closest RAM to being on sale. Because PRAM and its specifications are not released yet, we used the specification of [14]. Also, we emulated the PRAM with SDRAM by inserting additional delays in the file system level since the target board only contains SDRAM. Additionally, we filled up PRAM with sufficiently large database files in each application to consider the capacity limitation problem of PRAM. Table II shows the parameters for NAND flash memory and PRAM.

B. Write Performance Improvement of P-SQLITE

Firstly, we measure the read and write performance of P-SQLITE to analyze overall performance. As shown in the Table II, read latency of PRAM is similar to that with NAND flash memory and PRAM has much higher write latency than read latency. Thus, write performance is critical factor of P-SQLITE performance. In Figure 7, the write

Table II. Parameters for NAND flash memory and PRAM

	NAND Flash	PRAM
Device capacity	256MB	8MB
Page Size	2KB	-
Block Size	128KB	-
Read time	100μs/page	20ns/byte(40μs/2KB)
Write time	800μs/page	100ns/byte(200μs/2KB)
Erase time	1.5ms/block	-

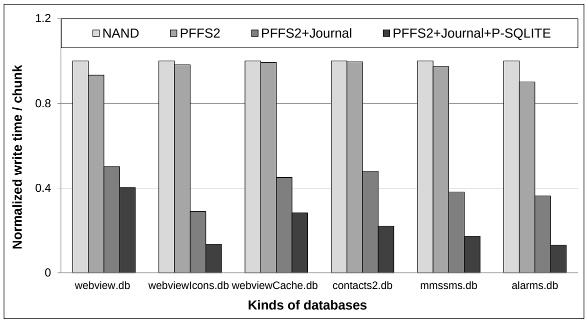
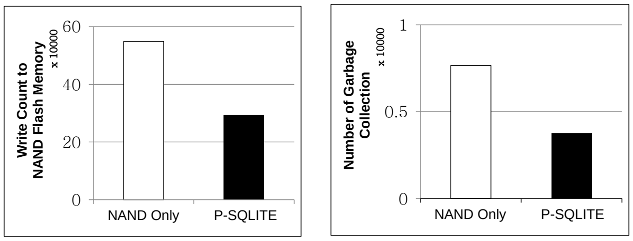


Figure 7: Normalized Write Time of Various Systems



(a) Write Count

(b) Garbage Collection

Figure 8: Write Count and Garbage Collection in NAND Flash Memory

performance of each database with four different approaches including P-SQLITE are evaluated. We combined P-SQLITE with previous works and estimated the write performance of them which were referred in Section II. ”PFFS2” [10] stores metadata in PRAM at the file system level and ”Journal” [11] places the rollback journal in PRAM at database software level. By adopting PFFS2, it gives only 7% write enhancement compared to NAND flash memory. Because MDBMS has only one file to handle the database, the effect of saving metadata in PRAM is comparatively small. Case of rollback journal file, it gives performance enhancement by almost 50% on average compared to NAND flash memory since it reduces the garbage collection. Finally, the average write performance improvement by P-SQLITE is about 47% compared to previous scheme where both previous works are applied. Since P-SQLITE is able to combine with previous works independently, the scheme adopting whole three techniques has 77% write performance improvement compared to only NAND flash memory.

C. Reduction of Write Count and Garbage Collection

We also estimated the total write count and number of garbage collection in NAND flash memory during 5000 write transactions of phone book application. Figure 8 shows that the P-SQLITE brings 46% write count in NAND flash memory and 51% reduction of garbage collection compared to NAND flash memory. It means our work concentrates the frequent write count in PRAM efficiently where it increases the write performance of MDBMS on the hybrid storage system.

D. Discussion

In our P-SQLITE system, additional cost is needed where the space overhead occurs due to the Section Table. However, additional two bytes per 2K chunk for Section Table are negligible compared to total performance.. Also, it causes additional overhead for migration. If write count pattern is flat, performance overhead can become smaller.

VI. Conclusion

In this paper, we propose P-SQLITE which enhances the write performance of the storage system with fine-grained data management. It contains section table which records the information of each chunk of the file. We also design hot data decision module to select the chunks which will be allocated to PRAM dynamically. Finally, P-SQLITE has chunk migration module which migrates colder chunk from PRAM to NAND flash memory to consider the limited size of PRAM. Through this implementation, we improve the write performance of the previous MDBMS in the hybrid storage architecture by 47% on average.

For the further work, we will focus on the write endurance problem of PRAM. Limited lifetime of PRAM is also challenging issue and proper wear-leveling technique for the system should be considered. We are planning to clarify the appropriate wear-leveling policy for mobile database management system.

Acknowledgment

The work presented in this paper was supported by MKE (Ministry of Knowledge Economy, Republic of Korea), Project No. 10035231-2010-01.

References

[1] S. Raoux, G. W. Burr, M. J. Breitwisch, C. T. Rettner, Y.-C. Chen, R. M. Shelby, M. Salinga, D. Krebs, S.-H. Chen, H.-L. Lung, and C. H. Lam, ”Phase-change random access memory: A scalable technology”, IBM Journal of Research and Development, vol. 52. no. 4, July. 2008, pp.465-479.

[2] A. Sheikholeslami and P. G. Gulak, ”A survey of circuit innovations in ferroelectric random-access memories”, Proceedings of the IEEE, vol. 88, no. 5, May. 2000, pp. 667-689.

[3] S. Tehrani, J. Slaughter, E. Chen, M. Durlam, J. Shi, and M. DeHerren, ”Progress and outlook for mram technology”, IEEE Transactions on Magnetics, vol. 35, no. 5, Sep. 1999, pp. 2814-2819.

[4] S. Lee, S. Jung, and Y. Song, ”An efficient use of PRAM for an enhancement in the performance and durability of NAND storage systems”, IEEE Transactions on Consumer Electronics, vol. 58, no. 3, Feb. 2000, pp. 825-833.

[5] H. Lee, ”High-performance NAND and PRAM hybrid storage design for consumer electronics”, IEEE Transactions on Consumer Electronics, vol. 56, no. 1, Feb. 2000, pp. 112-118.

[6] Sqlite official webpage, http://www.sqlite.org/, retrieved: March, 2013.

[7] Android, Software stack for mobile devices that includes an operating system, http://www.android.com/, retrieved: March, 2013.

[8] J.-M. Kim and J.-S. Kim, ”AndroBench: Benchmarking the storage performance of android-based mobile devices”, Frontiers in Computer Education AISC, 2012, pp. 667-674.

[9] Y. Lv, B. Cui, B. He, and X. Chen, ”Operation-aware bufferr management in flash-based systems”, Proc. The 2011 International Conference on Management of Data (SIGMOD’11), 2011, pp. 13-24.

[10] Y. Park and K. H. Park, ”High-performance scalable flash file system using virtual metadata storage with phase-change ram”, IEEE Transactions on Computers, vol. 60, no. 3, Mar. 2011, pp. 321-334.

[11] Y. Park, S. K. Park, and K. H. Park, ”Design of embedded database based on hybrid storage of pram and nand flash memory”, Proc. The 16th International Conference on Database Systems for Advanced Applications (DASFAA’11), 2011, pp. 254-264.

[12] N. Ye, Q. Chen, and C. Borror, ”Ewma forecast of normal system activity for computer intrusion” detection. IEEE Transactions on Reliability, vol. 53, no. 4, Dec. 2004, pp. 557-566.

[13] Hbe-empos3 sv210 board, http://www.hanbak.cn/en/, retrieved: March, 2013.

[14] M. Kryder and C. S. Kim, ”After hard drives: what comes next?”, IEEE Transactions on Magnetics, vol. 45, no. 10, Oct. 2009, pp. 3406-3413.

HMMSched: Hybrid Main Memory-Aware Task Scheduling on Multicore Systems

Woomin Hwang and Kyu Ho Park
Computer Engineering Research Laboratory, KAIST
Daejeon, Republic of Korea
{wmhwang, kpark}@core.kaist.ac.kr

***Abstract*—The strong demand for larger memory capacity with high energy efficiency creates the need for a hybrid main memory of DRAM and NVRAM (Non-Volatile RAM). In an attempt to provide task co-scheduling method on hybrid main memory, we found that the conventional task scheduling algorithms do not reflect the features of different memory mediums into scheduling decisions. Long access latency of the NVRAM make memory bandwidth usage of each task misestimated, thereby incurring unpredictable contention on memory accesses. Different access latencies according to the access type to the NVRAM deteriorates the contention. As a remedy to these problems, we propose HMMSched, which reflects different access latencies of hybrid main memory to the bandwidth estimation. Our scheme prioritizes CPU-intensive tasks, and then co-schedules tasks having complementary bandwidth consumption on different memory medium. Consequently, HMMSched reduces execution time of tasks by reducing memory access contention of co-scheduled tasks on the same memory. The experimental results show that the new scheduler reduces the overall execution time up to 19% than the default Linux scheduler.**

Keywords-Hybrid Main Memory; NVRAM; PRAM; task scheduling; memory contention;

I. Introduction

Increased concurrency of the workload executions brings not only great advances in system performance but limitations. Increasing number of computing cores achieve high instruction per cycle (IPC) by executing multiple instruction streams concurrently on same chip. The more tasks run concurrently, however, the greater the demand of memory because of the size of aggregated working set. It causes problems in the aspects of capacity, energy, and access contention. Enlarging memory size is expected to be limited by the clamped scalability of DRAM [1], [2]. Energy consumption of memory occupying 30-40% of server system energy [3], [4] will grow with the increase of DRAM size. Contention on memory accelerated by the use of many-core processors is a well-known problem on DRAM-based main memory. Cores share parts of the memory hierarchy and compete for resources. It brings delay in memory accesses, which results in degradation of application performance. As solutions, several studies proposed contention-aware scheduling [5], [6], [7], [8], [9], [10], [11], [12] to reduce memory contention. A contention-aware scheduler detects tasks having potentials of competing for memory hierarchies. Found tasks are scheduled at different time and location.

Hybrid main memory of DRAM and NVRAM is a promising architecture to enlarge memory capacity with high energy efficiency and little performance loss [3], [13], [14]. By placing both media at the same level of memory hierarchy, both memory are complementary to each other. Non-volatility of the NVRAM can reduce the total energy consumption of the main memory compared with DRAM-only main memory of the same capacity. DRAM complements the demerits of NVRAM on performance caused by its longer and asymmetric access latencies.

Unfortunately, existing studies of contention-aware task scheduling algorithms focused on DRAM main memory. With DRAM-only main memory, access latencies to the same memory bank are independent of access request type. In the hybrid main memory architecture, on the contrary, access latency varies according to the medium of the target memory, memory access type, and the location of the target memory on NUMA (Non-Uniform Memory Access) architecture.

NVRAM read latency is longer than DRAM access latencies, and NVRAM write latency is much longer than NVRAM read. When we ran hybrid main memory-agnostic contention-aware scheduling algorithms on the target system, we found that they do not aware resource usage well. It results in performance degradation. We have summarized our motivation about the performance degradation.

- **Multiple tasks concurrently accessing same type of memory increases contention:** Hybrid main memory is a combination of DRAM and NVRAM, where larger size of NVRAM replaces DRAM. With them, frequently accessed data are located in DRAM while NVRAM maintains data generating burst accesses [14]. This deteriorates task performance when memory-intensive tasks accessing same memory bank are co-scheduled. Concurrent tasks accessing hot data on reduced size of DRAM generate more contention like shown in Fig. 3. Overlapped burst accesses to NVRAM amplify longer access latencies of the medium.
- **The number of memory transactions does not reflect bandwidth occupancy of a task:** Existing contention-aware scheduling algorithms [8], [9], [10], [15], [16] estimates memory bandwidth usage by counting the

number of memory transactions. In the hybrid main memory environment, however, a write to NVRAM occupies target memory longer than a DRAM read. Type-agnostic counting of memory transactions to the NVRAM makes the value not proportional to the actual memory bandwidth usage. It incurs wrong decision of the task scheduler that tries to avoid memory contention based on the bandwidth usage of each task.

In this paper, we propose HMMSched, a task co-scheduling method for tasks running on hybrid main memory. HMMSched collects memory access information for each task to acquire memory bandwidth usage of each type. We define that Effective Bandwidth (EBW) is the estimated bandwidth considering the different access latencies of each memory medium. It translates bandwidth usage of NVRAM to the same unit of DRAM's not to make the estimated bandwidth fluctuate according to the target medium and access type of the memory access requests.

HMMSched performs following two phases to determine the next task to schedule. On the first phase, HMMSched prioritizes tasks having lower EBW consumption than the predetermined threshold. It is from the fact that a task having lowest EBW have greatest potential to make progress in the CPU. As the second phase, HMMSched separates EBW of each task according to its target medium for the remaining schedulable tasks. The tasks having low complementary EBW on different memory medium are alternately co-scheduled.

Our algorithm is implemented in Linux kernel with simulated per-task performance monitoring unit (PMU) attached to each workload. The preliminary experiment results show that our algorithm outperforms up to 19% better than the default Linux scheduler.

This study is an extension of our previous work [17], which is a part of resource management for the MN-MATE computing platform [18]. In the previous work, we focused on finding problems on task scheduling on hybrid main memory environment with small number of cores. Our objective in this paper, however, is to devise task scheduling policy with more consideration on real multicore execution environment and implementation issues.

The rest of this paper is organized as follows. Section II explains background of this work and Section III discusses related work. Section IV describes our motivation of this paper. Section V presents design of HMMSched and its implementation issues. Section VI provides the experimental results, and we conclude our work in Section VII.

II. Background

In this section, we explain two non-volatile memories as DRAM alternatives. Then we introduce a hybrid main memory, which is the target memory architecture of this paper.

Table I. Access latencies of memory technologies. Data obtained from [20], [21]

Technology	**Latency (ns)**	
	Read	***Write***
DRAM	25	25
STT-RAM	29.5	95
PRAM	67.5	215

A. Non-Volatile Memories

Non-volatile memory (NVRAM) technologies, unlike DRAM, will possibly enable memory chips that are non-volatile, require low-energy, and have density and latency closer to current DRAM chips [19]. Among emerging memories, Phase Change Memory (PRAM) and Spin-Transfer Torque RAM (STT-RAM) is one of the most promising technologies for future memory.

1) Phase-Change Memory (PRAM): PRAM is a byte-addressable, non-volatile memory based on phase change materials that can sustain phase persistently [22]. Persistent sustenance of phase gives non-volatility to PRAM, which makes leakage energy negligibly small. Because the length of the current pulse decides the direction of state change, a PRAM cell can be switched from 0 to 1 and vice-versa without any erase operation. It gives the memory in-place update property. PRAM is argued to be a scalable technology [19], [22], [1] for its high density. Recent works [4], [1] have demonstrated that the scalability will make PRAM a promising DRAM alternative for main memory.

On the other hand, there are several disadvantages. First, both read and write latencies of PRAM are several times slower than DRAM. Table I summarizes read and write latencies of main memory candidates. Second, large write energy mostly consumed by the reset operation is also one of the weak points of the PRAM. It makes the energy usage of PRAM depend on the number of writes. We will not cover the endurance problem of 10^8 rewrite sustainability [23].

2) Spin-Transfer Torque RAM: STT-RAM is another byte-addressable, non-volatile memory. It applies different write mechanism based on spin polarization [24], [25]. Compared with the PRAM access latencies, STT-RAM has very low read and write latencies. Read latency of STT-RAM is as fast as of DRAM according to [20]. Though the write latency is slower than DRAM, it is still very fast compared with the PRAM. It also has better endurance, reaching above 10^{15} cycles [24]. The weak point of STT-RAM is density. Its cell size has less density than current DRAM cells, shown in Table I. Smaller capacity from lower density depletes fast access speed, which is as fast as DRAM.

B. Hybrid Main Memory

Hybrid main memory is a combined architecture of DRAM and NVRAM for main memory [3], [13], [14], where both memory is located at the same level of memory

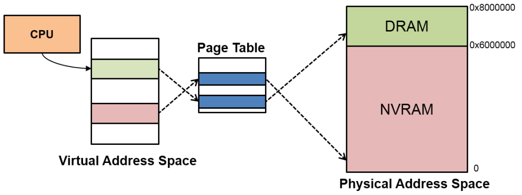


Figure 1: Hybrid main memory architecture. Both NVRAM and DRAM are located at the same level in the memory hierarchy.

hierarchy. It is designed to exploit the strenghs of each memory medium while avoiding their weaknesses as much as possible. By maintaining both media as a main memory at the same level, DRAM and NVRAM complements weak points of each other. Higher density of NVRAM provides main memory scalability, which will be limited by the DRAM. Lower standby power of NVRAM enlarges main memory capacity with lower energy consumption. Furthermore, shorter access latency of DRAM reduces performance loss due to long access speed of NVRAM [4].

In this paper, we target PRAM and STT-RAM as candidates for DRAM alternatives of main memory. Their byte-addressability and in-place update property enable them to be directly attached to the main memory interface, though they need separate memory controller. It makes the memory hierarchy provide both low latency and non-volatility. There are two studies [13], [14] providing kernel supports for the efficient use of the hybrid main memory architecture.

Fig. 1 shows our target structure of hybrid main memory architecture. In this architecture, NVRAM and DRAM are assigned to single physical memory address space. NVRAM has either lower or higher, contiguous physical address. OS can distinguish the medium of the target memory based on their physical address. OS manages all physical page frames and controls data placements based on reference characteristics. Any tasks can use both types of memory through the kernel.

There are some researches [26], [27] to extends the memory capacity with flash memories not attached to the main memory interface. However, they use either DRAM or Flash memory as a data cache or swap storage, which are not our concern in this paper. We motivated from the possibilities with the memory architecture having different read/write latencies. Therefore we only targets byte-addressable NVRAM and DRAM directly attached to the CPU at the same level of memory hierarchy.

III. Related Work

In this section, we explain previous researches about task co-scheduling. Contemporary studies present solutions that enhance system-wide performance in a system with multiple number of cores and DRAM-only main memory.

Memory bandwidth is one of the most popular criteria for memory-aware task scheduling. Koukis and Koziris [5], [6] proposed a scheduling method for SMP clusters. They profile bandwidth usage of each process and calculate average available bandwidth for new task. The scheduler selects the most fitting process whose bandwidth consumption best matches the average available bandwidth. Though it utilizes per task memory bandwidth consumption, it targets SMP clusters, which does not share caches.

In addition, Xu et al. [8] quantified the impact of memory bandwidth fluctuation on overall performance for tasks on multicore system. They found that bandwidth fluctuation measured using very fine time intervals could distorts the parameters of the job scheduler, especially when the total bandwidth usage approaches to peak system bandwidth. They proposed new scheduling criteria maintaining the total bandwidth requirement at a steady level instead of maximizing bandwidth utilization. Though they enhanced performance by reducing fluctuation in bandwidth consumption of the workloads, they targeted tasks running on a system with DRAM main memory. We borrowed some of their idea by not selecting tasks to fill in the available bandwidth. HMMSched co-schedules tasks by prioritizing latency-sensitive tasks followed by matching task consuming largest bandwidth. It doesn't compare bandwidth consumption with the available system bandwidth to fill in.

Miss rate of shared caches among cores is also a popular criterion for task co-scheduling on DRAM main memory. El-moursy et al. [7] tried to select thread pairs on each Simultaneous Multi-Threaded (SMT) processors combined with DRAM main memory. They add new hardware PMU and monitor contention on functional unit, register files, and caches. They define a phase as changing utilization of target resource and tried to find compatible phases that generate less conflicts on the resource. Compatible threads minimizing total cache miss rate are co-scheduled on the same SMT core.

Zhuravlev et al. [9] and Blagodurov et al. [10] also proposed a contention modeling heuristic named Distributed Intensity on DRAM main memory. They found that memory bus, prefetch hardware, and DRAM controller affecting performance degradation rely on cache miss rate. Based on the observation, they assigned threads to caches to even out the miss rate across all the caches. Though it showed good performance enhancement and performance stabilization, we target different execution environment where it utilizes NVRAM as a part of main memory. In addition, recent studies [15], [16] give an insight that miss rates does not reflect memory bandwidth because of the different write latency in PRAM main memory. Therefore, it is hard to apply to systems with hybrid main memory.

Several researches try to reduce energy consumption by scheduling tasks. Merkel and Bellosa [28] proposed memory-aware scheduling for energy efficiency on DRAM.

They also used performance counters to measure memory intensity. After sorting tasks on runqueues of cores, they paired cores. The scheduler selected two tasks from each runqueue, one having smallest memory intensity and the other one having largest memory intensity. Selected tasks are co-scheduled during time calculated by the unit time divided by the number of tasks in the source runqueue. They also applied frequency scaling for energy efficiency. DeVuyst et al. [29] also found that unbalanced number of threads on each core bring less energy consumption.

Suleman et al. [11] and Bhadauria and McKee [12] controlled the number of homogeneous threads to maintain bandwidth usage below the bandwidth limit. They utilized change of concurrency level on scheduling. But it is hard to apply tasks with noncontrollable concurrency. Again, their target systems include DRAM only while our target system includes NVRAM main memory.

Data placement is another issue on hybrid main memory for its effect to access patterns. Park et al. [14] proposed a data placement and migration policy between DRAM and PRAM at the same level in memory hierarchy. Ramos et al. [30] also proposed page placement methods where small-sized DRAM and large size PRAM is combined in a memory system. While they focused on the location of data and has little scheduling features, it affects access pattern of running tasks. Those policies can be complementarily cooperated with our scheduling policy in spite of the changed memory environment.

In this paper, we consider scheduling for manycores with hybrid main memory for objectives that are a composition of both performance and energy.

IV. Motivation

Previous researches concentrate on memory bandwidth usage estimation on an environment where DRAM is used as a main memory. However, these approaches are inappropriate for the hybrid main memory of DRAM and NVRAM. In this section, we explain our motivations in more detail.

A. Co-scheduling contention on Hybrid Main Memory

In the hybrid main memory environment, co-scheduling tasks intensively accessing same type of memory generates contention in several locations.

Fig. 2 shows contention on DRAM of hybrid main memory according to the task type. With 444.namd, which is classified as CPU-intensive task, accessing same bank of memory has little effect on performance due to low memory access intensity. They undergo rather little slowdown. With memory-intensive task such as 429.mcf, however, task performance is greatly affected by the contention on memory. The more memory accesses converge to the same memory bank, it makes execution time of each task longer.

The situation is getting worse with the hybrid main memory of DRAM and NVRAM. With the hybrid main memory, frequently accessing or short-lived data are usually located in DRAM while others are stored in NVRAM [3], [13], [14]. Memory accesses are flocked to the reduced number of DRAM banks. It increases memory access intensity, which is the number of memory accesses during a unit time. NVRAM accesses tend to be burst for the stored data's high spatial locality and low popularity. The performance deteriorates if the number of target memory bank decreases.

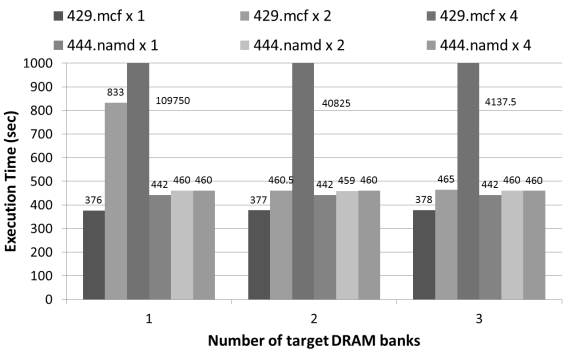


Figure 2: Slowdown of tasks according to the number of target memory banks and the number of concurrent tasks.

B. Inappropriate Bandwidth Estimation

Memory bandwidth (BW) usage has been one of the popular criteria for task scheduling working on DRAM main memory [5], [6], [8], [31]. With DRAM, access time of the memory does not depend on the type of the memory access request. Therefore, BW usage can be represented by the number of memory transactions generated during a unit time, shown in (1).

$$BW \approx \frac{Number\ of\ Memory\ Transactions}{Unit\ Time} \tag{1}$$

Several researches try to relieve the unequal distribution of memory bandwidth usage with the counted memory bandwidth [5], [6], [8]. Basic approaches of previous researches are to select next task consumed memory bandwidth not greater than the available bandwidth. More specifically, they measured total bandwidth usage of running tasks, BW_{Occ}. Based on the peak bandwidth, BW_{Peak}, estimated before, a scheduler calculates available bandwidth BW_{Avail}. It selects a next task that has memory bandwidth usage similar to the BW_{Avail}.

However, these methods are not applicable with the hybrid memory architecture where both DRAM and NVRAM are used as a part of main memory. Equation (5) is a decomposition of (2) according to the memory access type when the target memory is NVRAM. As we describe in Section II-A, access latency to the NVRAM is vary according to the request type.

TABLE II. SYMBOLS TO DESCRIBE (3) TO (5).

Symbol	Description
BW_D	occupied DRAM bandwidth
BW_{NV}	occupied NVRAM bandwidth
$BW(R)$	occupied memory bandwidth by read
$BW(W)$	occupied memory bandwidth by write
$BW_{D,Avail}$	available DRAM bandwidth
$BW_{NV,Avail}$	available NVRAM bandwidth
$BW_{D,Occ}$	occupied DRAM bandwidth
$BW_{NV,Occ}$	occupied NVRAM bandwidth
$BW_{D,Peak}$	peak DRAM bandwidth
$BW_{NV,Peak}$	peak NVRAM bandwidth
$BW_{D,Peak}(R)$	peak DRAM bandwidth by read
$BW_{D,Peak}(W)$	peak DRAM bandwidth by write
$BW_{NV,Peak}(R)$	peak NVRAM bandwidth by read
$BW_{NV,Peak}(W)$	peak NVRAM bandwidth by write

$$BW_{Avail} = BW_{Peak} - BW_{Occ} \tag{2}$$

$$BW_{Avail} = (BW_{D,Avail}, BW_{NV,Avail}) \tag{3}$$

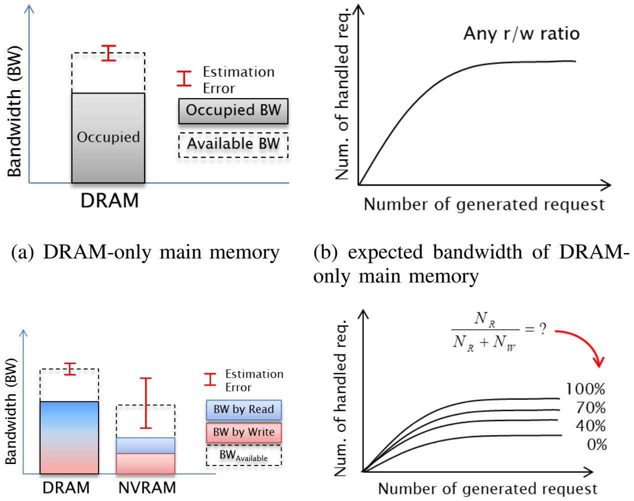
$$BW_{D,Avail} = BW_{D,Peak} - BW_{D,Occ} \tag{4}$$

$$\begin{aligned} BW_{NV,Avail} &= BW_{NV,Peak} - BW_{NV,Occ} \\ &= (BW_{NV,Peak}(R) \\ &+ BW_{NV,Peak}(W)) \\ &- (BW_{NV,Occ}(R) + BW_{NV,Occ}(W)) \end{aligned} \tag{5}$$

An NVRAM access request occupies target memory rank for a specified time according to the type of the request. Different latencies of each memory request type change the number of memory access requests handled during a unit time. Even a read operation consumes more time than the DRAM's, it is hard to represent the bandwidth consumption of a task by the number of memory access requests. It should be consider the target memory medium and the proportion of read and write operation to the number of memory transactions.

Fig. 3 illustrates the effect of different latencies to the bandwidth usage estimation. With DRAM main memory, occupied bandwidth of running tasks can be represented by the number of memory transactions. With hybrid main memory of DRAM and NVRAM, memory access requests can be classified into three categories based on access latencies: DRAM access, NVRAM read, and NVRAM write.

The problem is that estimated bandwidth of running tasks changes over the proportion of write to overall accesses. It makes the number of memory transactions hard to reflect the bandwidth usage of the task. It results in wrong decision about selecting next task satisfying decision criteria. Therefore, bandwidth estimation for NVRAM should reflect type of accesses by differentiating $BW_{NV}(W)$ and $BW_{NV}(R)$ as shown in (5).



(a) DRAM-only main memory (b) expected bandwidth of DRAM-only main memory

(c) Hybrid main memory (d) expected bandwidth of Hybrid main memory

Figure 3: Number of memory transactions handled during a unit time. NVRAM bandwidth usage changes according to the ratio of read and write.

V. HYBRID MAIN MEMORY-AWARE TASK SCHEDULING (HMMSCHED)

A. Basic Design

The primary design goal of the HMMSched is to accelerate task performance by co-scheduling tasks consuming complementary amount of memory bandwidth on different type of memory. To achieve this design goal, HMMSched consists of two phases: 1) per-task memory access monitoring; 2) selection of a next candidate task to be scheduled based on collected memory access statistics during the previous phase. During the running time of each task, hardware PMU collects memory access information for each task. Collected information includes type of target memory medium, memory access type, and the access counts. The HMMSched selects next task on the basis of the proposed policy regarding the collected statistics during the last scheduled period.

1) Per-task memory access monitoring: With HMMSched, memory bandwidth usage is an important criterion to choose next task to be run on an idle core. A software PMU, described in Section VI-A collects segmentalized memory access statistics for each task based on the access types and the target memory type. We have three criteria to classify memory accesses: target memory type, access type, and distance from the core where the target task runs to the target memory.

Target memory type is our primary consideration for classifying memory accesses. We targeted hybrid main memory that DRAM and NVRAM are located at the same level in

TABLE III. VARIABLES TO CALCULATE EFFECTIVE BANDWIDTH IN THE HYBRID MAIN MEMORY ENVIRONMENT

Variable	Description
$EBW(T)$	Effective bandwidth of a task T
EBW_D	Effective bandwidth for DRAM
EBW_{NV}	Effective bandwidth for NVRAM
BW_D	Conventional bandwidth for DRAM
N_{Req}	Number of memory access transactions
T_D	Memory access latency of DRAM
N_R	Number of read transactions to the NVRAM
N_W	Number of write transactions to the NVRAM
γ	NVRAM read latency / DRAM access latency
δ	NVRAM write latency / DRAM access latency

the memory hierarchy. However, each medium has different access latencies. If a task accesses data located in NVRAM, it can access less number of data within a unit time compared with when accessing data in DRAM. It makes the task make progress in the CPU. Therefore, we differentiate NVRAM accesses from DRAM accesses to find tasks having more potentials of making progress. Because the latency of write to NVRAM is much longer than read from the NVRAM, we also differentiate NVRAM writes from NVRAM reads based on the same criteria.

Generally, the DRAM bandwidth usage of a task is measured as the number of memory transactions during a unit time. In case of NVRAM, memory access latencies are different from each other according to the target media and access type. It results in that the number of memory access does not reflect memory bandwidth usage of a task. We therefore use a new metric, Effective Memory Bandwidth (EBW) to translate NVRAM's bandwidth usage and bandwidth usage of remote node memory into the number of DRAM transactions. Table III shows items that the per-task software PMU collects and translates. We can calculate effective bandwidth using (6).

$$\begin{aligned} EBW &= EBW_D + EBW_{NV} && (6)\\ EBW_D &= BW_D \\ &\simeq N_{Req} \times T_D \\ EBW_{NV} &\simeq \gamma \times N_R \times T_D + \delta \times N_W \times T_D && (7) \end{aligned}$$

When a task is about to be scheduled, the software PMU starts monitoring of memory accesses generated from the task. If the task consumes all allocated time slices or is preempted by other task, collected values are stored in the kernel memory. Software PMU then translates measured data to estimated values having same unit using (7). HMMSched use latest EBW value of each task to select next task to be scheduled on an idle core.

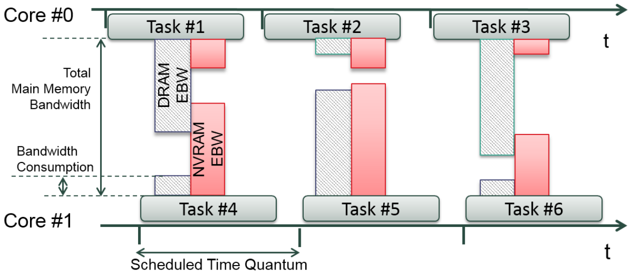


Figure 4: Basic idea of HMMSched. A task consuming complementary EBW to different medium is selected to be scheduled

2) Choosing a next task to be scheduled: Whenever a core becomes idle, HMMSched selects a task from scheduling candidate tasks based on a converted EBW.

As a preliminary work for choosing a next task to be scheduled, HMMSched arranges all schedulable tasks in order of EBW. HMMSched then classify them into two categories based on their latest EBW: latency-sensitive and bandwidth-sensitive. Latency-sensitive tasks consume less amount of memory bandwidth. Bandwidth-sensitive tasks spend more time to access both types of memory.

HMMSched use EBW as a criterion to classify tasks according to the bandwidth usage. Before classification, all tasks are arranged in order of EBW calculated by the (6). Let $Task_i$ indicates a task having *i*th lower EBW and $EBW(Task_i)$ indicates estimated effective bandwidth of $Task_i$. We then calculate $TotalEBW = \sum_{i=1}^{n} EBW(Task_i)$, where *n* is the number of all schedulable tasks. Tasks occupying some of total memory bandwidth greater than a predefined threshold α are classified as latency-sensitive. In other words, *K* tasks satisfying $\sum_{i=1}^{K} EBW(Task_i) < \alpha TotalEBW$ are classified into latency-sensitive tasks. Others are classified as bandwidth-sensitive. Here α is a parameter $0 \leq \alpha \leq 1$, where lower α indicates a stronger threshold. In order to prevent bandwidth-sensitive tasks from being subjected to starvation, α reflects total execution time of each category. HMMSched selects latency-sensitive task first as long as the total execution time of all latency-sensitive tasks is less than a tenth of the total execution time of bandwidth-sensitive tasks.

We applied different management scheme to each categories for ease of candidate selection. In the latency-sensitive category, all tasks are arranged in ascending order of own $EBW(T)$. In the bandwidth-sensitive category, each task belongs to two management lists; DRAM-intensive and NVRAM-intensive. DRAM-intensive list sorts all tasks in ascending order of their $EBW_D(T)$. NVRAM-intensive list arranges all tasks in ascending order of $EBW_{NV}(T)$.

When HMMSched tries to select a task to be scheduled next, the primary rule is that the scheduler always chooses

ISBN: 978-1-61208-272-1

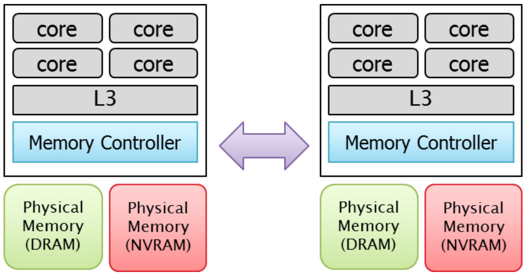


Figure 5: Target hardware architecture of HMMSched. DRAM and NVRAM share same memory controller on each local node of the NUMA system.

a latency-sensitive task prior to selecting a bandwidth-sensitive task. A task consumed lowest EBW has highest priority if there are multiple latency-sensitive tasks.

If there is no latency-sensitive task, HMMSched picks a task from the bandwidth-sensitive category based on the following order:

1) a task with largest EBW_D
2) a task with smallest EBW_{NV}
3) a task with largest EBW_{NV}
4) a task with smallest EBW_D

Fig. 4 illustrates this policy. It first makes the scheduler co-schedule intensively accessing different type of memory together. It then co-schedules tasks having complementary access intensities to the same type of memory.

VI. Evaluation

In this section, we evaluate the performance of HMMSched described in the previous section. We used two experiment systems for evaluation with Intel I7 960 processor running at 3.2GHz, 6GB of RAM, 2GB per DIMM. The operating system is Linux 2.6.38.2. Simulated NVRAM and PMU supporting per-task memory access monitoring is described in the next section. In this paper, we targeted a manycore system combined with hybrid main memory of DRAM and NVRAM, shown in Fig. 5. DRAM and NVRAM in a local node share same memory controller.

A. Modeling customized PMU and NVRAM

To evaluate our idea, we need two items: NVRAM main memory module and the hardware performance monitoring unit (PMU), which counts per-task memory accesses. Unfortunately, there is no released NVRAM module for main memory and commercial CPUs with PMU counting per-task memory accesses. We use Intel Pin [32] to perform a detailed simulation of the system's main memory augmented with NVRAM and the per-task memory access monitoring support that HMMSched requires. The memory simulator adds designated delay between a read or a write to the NVRAM and the operations that follow, allowing it to accurately model the longer read and write times of NVRAM. For NVRAM, we use the performance model of PRAM from [21], which gives an access latency ratio shown in Table I.

As a software PMU, the simulator monitors main memory accesses for each task and counts the number of following requests separately: the number of DRAM accesses, the number of NVRAM reads, and the number of NVRAM writes. Fig. 6(a) illustrates a simulation environment where each task runs with own per-task simulation module. Each simulation module acts like a PMU for single task. To estimate the number of memory accesses, each simulation module has own cache hierarchy model. Each simulation module shares its last-level cache model with other modules when their target tasks run on the cores sharing same last-level cache. Collected memory access information is transferred to the task scheduler via shared memory.

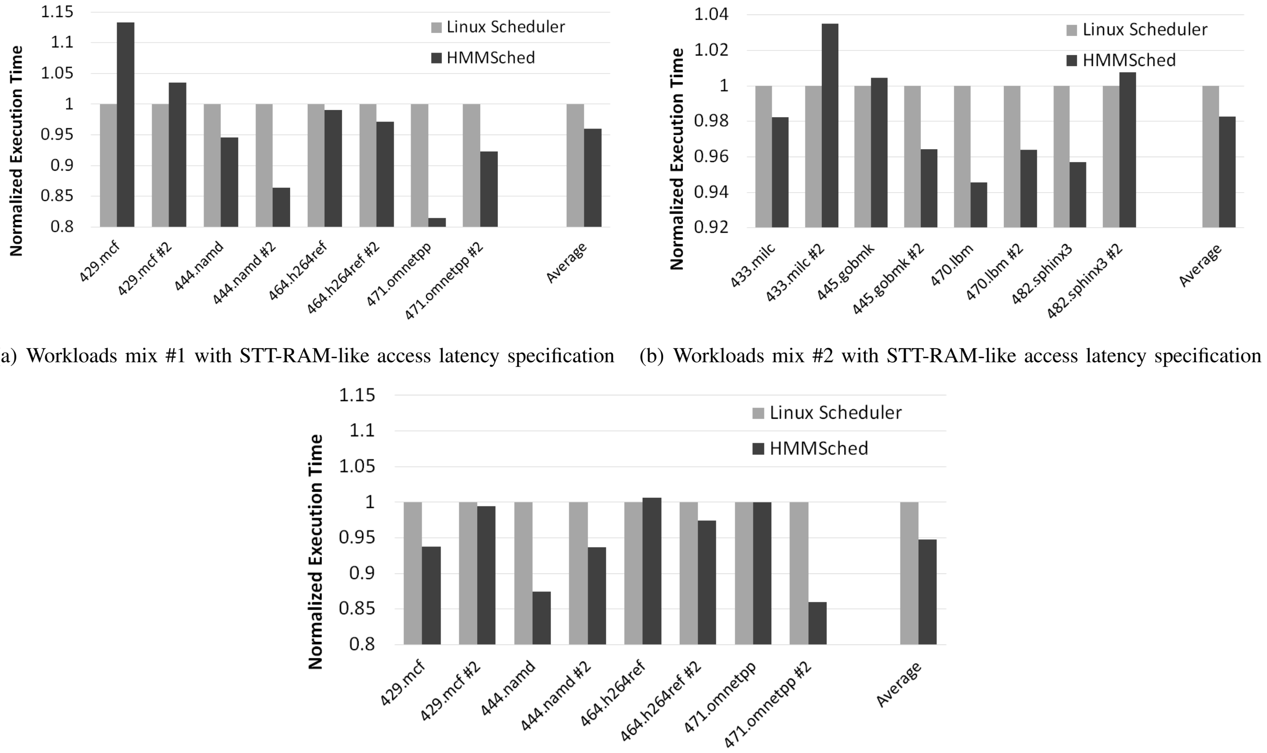
Note that collected information from the simulation module should reflect actual memory access information generated by the combination of the target task and the attached simulation module. It is because of the fact that the simulated PMU monitors memory accesses of the target task while the scheduler schedules the target task running with the simulation module. We calibrated the simulated PMU by adjusting miss rates of the shared last level cache. As a result, the software PMU reflects the memory access characteristics generated from the combination of the target task and the simulation module.

B. Workloads

We use the SPEC CPU2006 benchmarks [33] for evaluation. We compiled each benchmark using gcc 4.4.3 with default optimizations of the benchmark, and executes on Linux 2.6.38.2 kernel. Two workload set is used to evaluate the effectiveness of HMMSched. Each workload set consists of four applications, where each application has two instances. In the first set, half of the applications are memory-intensive tasks while others are CPU-intensive tasks. Second set has six memory-intensive tasks and two CPU-intensive tasks. We configured them to check the effect of prioritization to the latency-sensitive tasks and the effect of various memory access intensities.

C. Preliminary Result

Fig. 7 shows preliminary experimental results of the HMMSched compared with the default Linux scheduler. Y-axis of the graphs indicates normalized execution time of the benchmarks. In this paper, we add experiment results with local node sharing same memory controller. We leave experiments related to NUMA architecture as our further work. Here we used following NVRAM access latency specifications. First set is DRAM : NVRAM(read) : NVRAM(write) = 1 : 1 : 3, which is similar to the specification of STT-RAM shown in Table I. Second set is DRAM : NVRAM(read) : NVRAM(write) = 1 : 3 : 8, which is also similar to the specification of PRAM in

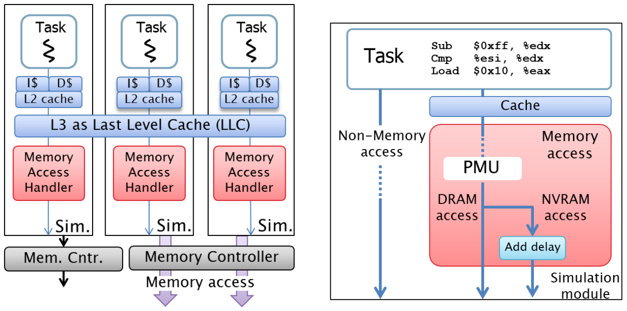


(a) Workloads mix #1 with STT-RAM-like access latency specification

(b) Workloads mix #2 with STT-RAM-like access latency specification

(c) Workloads mix #1 with PRAM-like read/write access latency specification

Figure 7: Elapsed time of selected benchmark sets under HMMSched compared with default Linux scheduler.



(a) For multiple tasks

(b) Internal flow of a simulation module

Figure 6: Overall simulation environment running multiple tasks. Each task runs with per-task simulation module.

Table I. We also composed two benchmark set to analyze the effect of the priority change of the scheduler when the latency-sensitive tasks are run with bandwidth-sensitive tasks. Benchmark mix #1 consists of four memory-intensive tasks and four CPU-intensive tasks. Benchmark mix #2 contains six memory-intensive tasks and two CPU-intensive tasks. On each experiment, all benchmarks of the target set run concurrently.

As we can see in the two experiment results, prioritizing latency-sensitive tasks and utilizing EBW on co-scheduling bandwidth-sensitive tasks can effectively reduce completion time of them. An instance of 471.omnetpp completed by 19% faster than under the default Linux scheduler. In addition, we can reduce average running time of all workloads by 5% under HMMSched even though it hurts the performance of 429.mcf in Fig. 7(a) With the workloads mix #2, we can also see improvements with smaller performance degradation of 433.milc #2. If we target PRAM as an alternative of main memory, our scheduling policy affects more on concurrently running application performance. Figure 7(c) shows experiment result with PRAM specification for NVRAM part of main memory. Because of the long read access latency of PRAM compared with DRAM, co-scheduling bandwidth-sensitive tasks utilizing EBW shows better performance than with the default linux scheduler. With STT-RAM specification, tasks generating more NVRAM write get performance penalty. With PRAM specification, however, read latency also affects the effective bandwidth usage of tasks. It leads to leveling-off of each task's EBW usage so that they divide up the penalty from scheduling delay. Consequently, though several tasks are slightly hurt their performance, we can get more balanced performance gain.

VII. Conclusion and Further Work

As manycore increases the need for larger memory capacity, contention on memory has become a key issue. Although hybrid main memory of DRAM and NVRAM can enlarge memory capacity with high energy efficiency, hybrid main memory-agnostic contention-aware scheduling degrades task performance. Different access latencies to the NVRAM compared with DRAM prevent bandwidth consumption of each task hard to measure. In this paper, we propose HMMSched, a novel task co-scheduling method for manycore and a hybrid main memory of DRAM and NVRAM. We define effective bandwidth to reflect different access latencies according to the access type to the NVRAM. We then prioritize CPU-intensive tasks to make more progress in the CPU. Tasks consuming complementary EBW on different memory are alternately selected to be co-scheduled with running tasks. Consequently, HMMSched co-schedule tasks consuming different memory type with complementary access intensity, thereby reducing delay from contention on the memory. The experimental results show that our scheme outperforms default Linux scheduler by up to 19% in terms of time efficiency.

As further works, we will add hybrid main memory management features proposed in [14] to analyze effect of data migration policy on the HMMSched scheduling. We will also add task co-scheduling policy when the target memory hierarchy includes NUMA architecture. Finally, we will carefully analyze the effect of cache hit ratio according to the ratio of read and write to the NVRAM main memory.

References

[1] B. C. Lee, E. Ipek, O. Mutlu, and D. Burger, “Architecting phase change memory as a scalable dram alternative,” in ISCA ’09: Proceedings of the 36th annual international symposium on Computer architecture. New York, NY, USA: ACM, 2009, pp. 2–13.

[2] P. Zhou, B. Zhao, J. Yang, and Y. Zhang, “A durable and energy efficient main memory using phase change memory technology,” in ISCA ’09: Proceedings of the 36th annual international symposium on Computer architecture. New York, NY, USA: ACM, 2009, pp. 14–23.

[3] G. Dhiman, R. Ayoub, and T. Rosing, “Pdram: a hybrid pram and dram main memory system,” in DAC ’09: Proceedings of the 46th Annual Design Automation Conference. New York, NY, USA: ACM, 2009, pp. 664–469.

[4] M. K. Qureshi, V. Srinivasan, and J. A. Rivers, “Scalable high performance main memory system using phase-change memory technology,” in ISCA ’09: Proceedings of the 36th annual international symposium on Computer architecture. New York, NY, USA: ACM, 2009, pp. 24–33.

[5] E. Koukis and N. Koziris, “Memory bandwidth aware scheduling for smp cluster nodes,” in PDP ’05: Proceedings of the 13th Euromicro Conference on Parallel, Distributed and Network-Based Processing. Washington, DC, USA: IEEE Computer Society, 2005, pp. 187–196.

[6] ——, “Memory and network bandwidth aware scheduling of multiprogrammed workloads on clusters of smps,” in Proceedings of the 12th International Conference on Parallel and Distributed Systems - Volume 1, ser. ICPADS ’06. Washington, DC, USA: IEEE Computer Society, 2006, pp. 345–354.

[7] A. El-Moursy, R. Garg, D. H. Albonesi, and S. Dwarkadas, “Compatible phase co-scheduling on a cmp of multi-threaded processors,” in Proceedings of the 20th international conference on Parallel and distributed processing, ser. IPDPS’06. Washington, DC, USA: IEEE Computer Society, 2006, pp. 10–19.

[8] D. Xu, C. Wu, and P.-C. Yew, “On mitigating memory bandwidth contention through bandwidth-aware scheduling,” in PACT ’10: Proceedings of the 19th international conference on Parallel architectures and compilation techniques. New York, NY, USA: ACM, 2010, pp. 237–248.

[9] S. Zhuravlev, S. Blagodurov, and A. Fedorova, “Addressing shared resource contention in multicore processors via scheduling,” in ASPLOS ’10: Proceedings of the fifteenth edition of ASPLOS on Architectural support for programming languages and operating systems. New York, NY, USA: ACM, 2010, pp. 129–142.

[10] S. Blagodurov, S. Zhuravlev, A. Fedorova, and A. Kamali, “A case for numa-aware contention management on multicore systems,” in Proceedings of the 19th international conference on Parallel architectures and compilation techniques, ser. PACT ’10. New York, NY, USA: ACM, 2010, pp. 557–558.

[11] M. A. Suleman, M. K. Qureshi, and Y. N. Patt, “Feedback-driven threading: power-efficient and high-performance execution of multi-threaded workloads on cmps,” in Proceedings of the 13th international conference on Architectural support for programming languages and operating systems, ser. ASPLOS XIII. New York, NY, USA: ACM, 2008, pp. 277–286.

[12] M. Bhadauria and S. A. McKee, “An approach to resource-aware co-scheduling for cmps,” in ICS ’10: Proceedings of the 24th ACM International Conference on Supercomputing. New York, NY, USA: ACM, 2010, pp. 189–199.

[13] J. C. Mogul, E. Argollo, M. Shah, and P. Faraboschi, “Operating system support for nvm+dram hybrid main memory,” in Proceedings of the 12th conference on Hot topics in operating systems, ser. HotOS’09. Berkeley, CA, USA: USENIX Association, 2009, pp. 14–18.

[14] Y. Park, S. K. Park, and K. H. Park, “Linux kernel support to exploit phase change memory,” in Linux Symposium, 2010, pp. 217–224.

[15] S. Lee, H. Bahn, and S. Noh, “Characterizing memory write references for efficient management of hybrid pcm and dram memory,” in Modeling, Analysis Simulation of Computer and Telecommunication Systems (MASCOTS), 2011 IEEE 19th International Symposium on, july 2011, pp. 168 –175.

[16] M. Zhou, Y. Du, B. Childers, R. Melhem, and D. Mossé, “Writeback-aware partitioning and replacement for last-level caches in phase change main memory systems,” ACM Trans. Archit. Code Optim., vol. 8, no. 4, Jan. 2012, pp. 1–21.

[17] K. H. Park, S. K. Park, W. Hwang, H. Seok, D.-J. Shin, and K.-W. Park, “Resource management of manycores with a hierarchical and a hybrid main memory for mn-mate cloud node,” in Services (SERVICES), 2012 IEEE Eighth World Congress on. IEEE, 2012, pp. 301–308.

[18] K. H. Park, Y. Park, W. Hwang, and K.-W. Park, “Mn-mate: Resource management of manycores with dram and nonvolatile memories,” in High Performance Computing and Communications (HPCC), 2010 12th IEEE International Conference on. IEEE, 2010, pp. 24–34.

[19] G. W. Burr, B. N. Kurdi, J. C. Scott, C. H. Lam, K. Gopalakrishnan, and R. S. Shenoy, “Overview of candidate device technologies for storage-class memory,” IBM J. Res. Dev., vol. 52, July 2008, pp. 449–464.

[20] A. M. Caulfield, J. Coburn, T. Mollov, A. De, A. Akel, J. He, A. Jagatheesan, R. K. Gupta, A. Snavely, and S. Swanson, “Understanding the impact of emerging non-volatile memories on high-performance, io-intensive computing,” in Proceedings of the 2010 ACM/IEEE International Conference for High Performance Computing, Networking, Storage and Analysis, ser. SC ’10. Washington, DC, USA: IEEE Computer Society, 2010, pp. 1–11.

[21] J. Coburn, A. M. Caulfield, A. Akel, L. M. Grupp, R. K. Gupta, R. Jhala, and S. Swanson, “Nv-heaps: making persistent objects fast and safe with next-generation, non-volatile memories,” in Proceedings of the sixteenth international conference on Architectural support for programming languages and operating systems, ser. ASPLOS ’11. New York, NY, USA: ACM, 2011, pp. 105–118.

[22] S. Raoux, G. W. Burr, M. J. Breitwisch, C. T. Rettner, Y.-C. Chen, R. M. Shelby, M. Salinga, D. Krebs, S.-H. Chen, H.-L. Lung, and C. H. Lam, “Phase-change random access memory: A scalable technology,” IBM Journal of Research and Development, vol. 52, no. 4.5, july 2008, pp. 465 –479.

[23] S. Eilert, M. Leinwander, and G. Crisenza, “Phase change memory: A new memory enables new memory usage models,” in Memory Workshop, 2009. IMW ’09. IEEE International, may 2009, pp. 1–2.

[24] H. Li and Y. Chen, “An overview of non-volatile memory technology and the implication for tools and architectures,” in Design, Automation Test in Europe Conference Exhibition, 2009. DATE ’09., april 2009, pp. 731 –736.

[25] B. Dieny, R. Sousa, G. Prenat, and U. Ebels, “Spin-dependent phenomena and their implementation in spintronic devices,” in VLSI Technology, Systems and Applications, 2008. VLSI-TSA 2008. International Symposium on, april 2008, pp. 70 –71.

[26] M. Saxena and M. M. Swift, “Flashvm: virtual memory management on flash,” in Proceedings of the 2010 USENIX conference on USENIX annual technical conference. USENIX Association, 2010, pp. 14–27.

[27] A. Badam and V. S. Pai, “Ssdalloc: hybrid ssd/ram memory management made easy,” in Proceedings of the 8th USENIX conference on Networked systems design and implementation, ser. NSDI’11. Berkeley, CA, USA: USENIX Association, 2011, pp. 16–29.

[28] A. Merkel and F. Bellosa, “Memory-aware scheduling for energy efficiency on multicore processors,” in HotPower’08: Proceedings of the 2008 conference on Power aware computing and systems. Berkeley, CA, USA: USENIX Association, 2008, pp. 1–5.

[29] M. DeVuyst, R. Kumar, and D. M. Tullsen, “Exploiting unbalanced thread scheduling for energy and performance on a cmp of smt processors,” in Proceedings of the 20th international conference on Parallel and distributed processing, ser. IPDPS’06. Washington, DC, USA: IEEE Computer Society, 2006, pp. 140–149.

[30] L. E. Ramos, E. Gorbatov, and R. Bianchini, “Page placement in hybrid memory systems,” in Proceedings of the international conference on Supercomputing, ser. ICS ’11. New York, NY, USA: ACM, 2011, pp. 85–95.

[31] Y. Kim, M. Papamichael, O. Mutlu, and M. Harchol-Balter, “Thread cluster memory scheduling: Exploiting differences in memory access behavior,” in Proceedings of the 2010 43rd Annual IEEE/ACM International Symposium on Microarchitecture, ser. MICRO ’43. Washington, DC, USA: IEEE Computer Society, 2010, pp. 65–76.

[32] C.-K. Luk, R. Cohn, R. Muth, H. Patil, A. Klauser, G. Lowney, S. Wallace, V. J. Reddi, and K. Hazelwood, “Pin: building customized program analysis tools with dynamic instrumentation,” SIGPLAN Not., vol. 40, June 2005, pp. 190–200.

[33] “SPEC CPU 2006 Benchmark Suite.” [Online]. Available: http://www.spec.org/cpu2006

Optimal Beacon and Superframe Orders in WSNs

Marwa Salayma, Wail Mardini, Yaser Khamayseh, Muneer Bani Yasin
Department of Computer Science
Jordan University of Science and Technology (JUST)
Irbid 22110, Jordan
{mksalayma, mardini, yaser, masadeh}@just.edu.jo

Abstract— **Most of Wireless Sensor Network (WSN) applications aim to utilize low data rates, consume very low energy, and operate in short range areas at low costs. IEEE 802.15.4 standard is proposed in order to achieve such needs by putting standards for physical and Medium Access Control (MAC) layers. MAC operates in either beacon enabled or beaconless modes. Performance of the standard beacon enabled mode is basically affected by beacon frame parameters, which are Beacon Order (BO) and Superframe Order (SO). These two parameters determine node's active and inactive periods, they also determine node's duty cycle. BO and SO values must be chosen carefully when the standard MAC is parameterized since some combinations may degrade standard performance dramatically. Finding the optimal (BO, SO) combination is an application-based issue since diverse WSN applications work through different arrival rates at different duty cycles. This paper investigates the standard beacon enabled mode behavior through intensive simulation and revels the optimal range of (BO, SO) combinations for specific star topology with different number of nodes. The investigated application is evaluated in terms of energy consumption, average end to end delay and throughput. Moreover, this paper proposes an adaptive algorithm that converges to the network current performance and improves network performance accordingly, irrespective of the duty cycle. The performance of the new adaptive algorithm is compared to that of the original MAC algorithm using QualNet 5.2 simulator. The new adaptive algorithm outperforms the original MAC algorithm as it reduces energy consumption up to 7%, decreases average delay by 26%, and increases throughput by 16%.**

Keywords-Wireless sensor networks; IEEE802.15.4; Beacon enabled; Superframe structure; Energy consumption.

I. INTRODUCTION

Recently, most wired sensors are replaced with wireless ones forming the emerging era of Wireless Sensor Networks (WSNs). WSNs consist of sensing devices that can communicate with each other and with the surrounding environment via wireless communication medium [1] [2]. Huge number of sensor nodes are often scattered in unreachable areas, and WSNs are often battery powered and cannot be easily recharged. Moreover, it is reasonable to force sensor nodes to track and monitor phenomena for months or even years [2].

Energy conservation is the main concern for researches in the area of WSN, these studies focus on designing WSN energy efficient algorithms and standards, one of which is the IEEE 802.15.4 [3]. IEEE 802.15.4 refers to Wireless Personal Area Network (WPAN) standard proposed by the Institute of Electrical and Electronic Engineers Task Group 4 (IEEE TG4) to support Low Rate (LR) applications; it is often referred to as LR-WPAN IEEE 802.15.4 standard. LR-WPAN is designed for home, building and industrial automated applications [4].

IEEE802.15.4 standard supports both the physical and the Media Access Control (MAC) layers. IEEE802.15.4 MAC supports two types of devices; Full Functional Devices (FFDs) and Reduced Functional Devices (RFDs). FFD acts as a regular coordinator and/or as a sink node. If both features are taken, the node typically referred to as PAN coordinator (PANc). However, RFD acts as an ordinary end device [4-6]. Both FFDs and RFDs communicate with each other forming two types of topologies: star and peer to peer topologies. Peer to peer topology can be classified to either a mesh or a cluster tree topology [6-8].

IEEE802.15.4 standard operates in three different Radio Frequency (RF) bands and it supports different data rates [4]. The 2.4 GHz RF band is widely used in IEEE802.14.4 based applications due to the following: Firstly, there is no license required to operate in this RF, and secondly, it offers the highest data rate amongst the 16 operating channels [4].

IEEE802.15.4 MAC operates either in beacon enabled or beaconless modes. In the beacon enabled mode, FFD broadcasts regular beacon frames in order to advertise itself to the other nodes The beacon frame includes information that enables the nodes to synchronise with each other when they need to access the channel [9][10]. Furthermore, beacon frame includes information that indicates whether there is some pending data for some nodes. The time between two successive beacons is referred to as the Beacon Interval (BI), which divided virtually into 16 equal sized slots. BI duration is specified by the Beacon Order parameter (BO) according to the following formula [9]:

$$BI = aBaseSuperframeDuration * 2^{BO} \quad (1)$$

Nodes can use the channel during the whole BI period or can sleep for some time portions; the parameter which decides that is the Superframe Order (SO), the SO decides

ISBN: 978-1-61208-272-1

the Superframe Duration (SD) active session according to the following formula [9]:

$SD = aBaseSuperframeDuration * 2^{SO}$ (2)

where $0 \leq SO \leq BO \leq 14$

aBaseSuperframeSuration value depends on the slot duration according to the following formula:

aBaseSuperframeDuration= aBaseslotDuration * total number of slots (3)

Typically, time durations are expressed in term of a general time unit, that is, the symbol. The value of one symbol in seconds depends on the chosen RF band. However, the 2.4 Ghz RF band works in 62500 symbol/s at which one symbol brings out 16 µs, and as each slot duration (aBaseslotDuration) equals about 60 symbols, then the total 16 slots contribute in 960 aBaseSuperframeDuration symbols are equal to 15.36 ms. From those equations, node can infer the duration of its sleep period. All those concepts can be indicated through one concept which is the duty cycle (D). It is the percentage of time the node is awake from the whole time between the two successive beacons. D is mathematically expressed in (4) [9][10]:

$D=SD/BI * 100\%$ (4)

When a node need to access the medium, it has to locate the beginning of the next time slot in order to compete for the channel, thus, it follows the contention based algorithm followed by the standard, that is, the slotted Carrier Sense Multiple Access/Collision Avoidance algorithm (CSMA/CA); this is why this time portion is referred to as Contention Access Period (CAP) [9][10]. Furthermore, the standard empowers PANc with the authority to assign some slots excessively for some nodes during which they can utilize the channel alone. This is why such time slots are referred to as Guaranteed Time Slots (GTS).

The optional period which includes those slots is referred to as the Contention Free period (CFP) and it include maximum of seven GTS which are preserved optionally after the CAP period. CAP and the optional CFP together are referred to as the Active Period. Active period is the time during which nodes can be active and are able to use the medium. The duration of this period is often referred to as the SuperFrame Duration [9][10]. More precisely, every time the node needs to access the channel, it needs to locate the boundary of what is called the slotted/un-slotted CSMA backoff period. The Backoff period unit is indicated through the aUnitBackoffPeriod which equals to 20 symbols or 0.32 ms [10]. The lengths of the discussed periods are assigned through the beacon frame which is transmitted in the first time slot (slot 0) [4].

Obviously, improving the beacon enabled standard performance is directly related to the chosen BO and SO values. How to decide the optimal BO and SO values that achieve the best performance is an application related issue. For example, an application may have packets ready for transmission every second but needs to be active for 30 minutes and sleep for 30 minutes. Some applications spend most of the time inactive, thus, i.e., low duty cycles; others need to work through full duty cycles, while many of applications need to sleep for some time portions. Hence, each application has its own special case that has much to do in the decision of BO and SO values, keeping in mind that the basic building block of any network topology consists of seven nodes (piconet). Therefore, we need to find a mechanism that is general enough for beacon enabled MAC coordinator to regularly examine PAN status and performance to adapt the superframe parameters and durations.

This paper is organized as follows. Section II summarizes some of literature work which is closely related to the paper topic, while section III investigates the problem and illustrates the followed methodology. Section IV clarifies the proposed algorithm. The algorithm performance is then evaluated in Section V. Finally the whole paper topic, analysis and work are concluded in the last section along with some ideas and work proposed to be adopted in the future.

II. RELATED WORK

Since the launch of IEEE 802.15.4 standard, many researches took place that analyse the standard performance either mathematically through analysis models or virtually through simulation or both. Those researches are primarily aimed at finding ways for enhancing the standard performance especially when it comes to energy consumption. Such interests aimed at choosing the best standard parameters values that suits the applications to work for the longest time possible. f those studies is listed below.

In [11], the IEEE 802.15.4 standard performance is evaluated in terms of throughput and packet delivery ratio. The study focused in the quality of service (QoS) for real time sensor applications and provides an enhancement to the current IEEE 802.15.4 beacon enabled standard by dynamically allocating the already existed GTS. The standard performance metrics were evaluated through varying both BO and SO values while preserving the one dynamically allocated GTS. The study considered both 100% and 50% duty cycles. And, the maximum SO and BO values tested were 6 due to the association latency that may result from choosing higher values that are not suitable for WSN applications. Other QoS property examined was the collision probability which was evaluated through varying number of nodes. Simulation run through NS2 simulator and applied on a star topology. Results showed that high values of BO increase throughput due to the decreased possibility of packets drops. Moreover, results revealed that collision probability increases as the number of nodes increases which will degrade the successful use of the channel and hence achieve poor throughput.

In [12], the performance of beacon enabled IEEE 802.15.4 is evaluated in term of energy consumption in a large scale clustered tree network. Analysis of the IEEE 802.15.4 MAC were performed on a real ZigBee nodes applied on home network areas by varying BO values between 6 and 10 while fixing SO value to 0. High fraction of packets transmitted is sacrificed for the aim of minimizing the power consumption by allowing nodes to stay active for only 15.36 ms and turn the transceiver off else after. Results revealed that power consumption keeps on decreasing by increasing BO to some value (approximately 10) after which it is started to increase again. However, the study considered only very low duty cycles due to the small fraction of CAP and did not consider the effect of SO on the standard performance at all.

In [13], performance of the slotted CSMA/CA is investigated through studying the effects of SO, BO and Backoff Exponent (BE). However, the same study took place in [3], which considered other criteria such as the number of nodes and the data frame size. Simulation experiments were done for 13 different values of BO and SO which contributes in a 100% duty cycles. Those experiments intended to reach up the best range of traffic load offered that achieves the optimal performance metrics values. Metrics which were evaluated are the throughput, average delay and network reliability. The best range of offered load that achieved the optimal trade-off between throughput and average delay utility was found to be between 35% and 60%. This study did not concern parameters behaviour with sleep period enabled.

However, Shu et al. [6] proposes an optimization problem in order to achieve the minimum energy consumed under the packet delivery reliability constrain. The objective function was achieved after finding optimal values for the two decision variables which are BO and SO. Experiments run through a C implemented simulator and applied on a star topology. Simulation results revealed that for a network where packets generated under Poisson processes and where the number of nodes varies from 5 to 35, the optimal value of BO was found to be 7 while that for SO was found to be 1, that's just in case that number of nodes is less than 15 and 2. However, choosing optimal values for BO and SO depends on the quality of service constraints chosen.

The authors in [14] propose an algorithm that reconfigures only the BO parameter of the IEEE 802.15.4 superframe structure. The Beacon Order Adaptive Algorithm (BOAA) was investigated and applied on a star topology. Changing BO depends on the inter-arrival rate which reflects the frequency of communication. Adjusting the value of BO changes the length of the duty cycle due to the dynamic changing of the beacon interval. Experimental results showed that increasing the value of BO contributes in saving power due to the increased inactive period. However, this power saving improvement would be at the expense of the delay because increasing BI would cause nodes wait more time for the next beacon which determines their new configuration. Throughput was not taken under consideration because only small numbers of nodes were allowed to send a light traffic. This makes BOAA suitable only for simple applications which need no real-time or complex configuration.

In [15], IEEE 802.15.4 standard performance is investigated in terms of throughput, energy consumption and reliability by applying the standard on ideal and non-ideal star topologies. The focus was on changing nodes number while varying some of IEEE 802.15.4 standard configurations such as the availability of synchronization, BO and SO. According to the results achieved, some recommendations were suggested that aid in configuring the standard, configuring applications that follow the standard and how to improve the standard. However, such recommendations can only be taken under consideration when applying the standard on the same topologies tested where only the behaviour of BO=SO (100% duty cycle) considered . Moreover, most of the suggestions provided were based on theoretical ideas and not on practical achieved results. However, all such researches are very application specific and none gives a general enhancement that can be applied on all WSN applications.

III. METHODOLGY AND PROBLEM ANALYSIS

In order for the current transaction to complete during the current superframe active session, the remaining CAP backoff periods must be sufficient enough to accommodate the whole CSMA/CA operations; otherwise, nodes shall wait for the next superframe. If this situation continues to happen, network performance will be adversely affected. Actually, PAN performance is adversely affected by high and low values of both BO and SO parameters. This section investigates the effect of all possible (BO, SO) combination values on IEEE 802.14.4 standard performance through simulation. The Simulation study is conducted using QualNet 5.2 simulator. Simulation parameters are summarized in Table 1.

PANs consist of one FFD that is the PANc while the other devices are RFDs. RFDs transmit a 50 byte Constant Bit Rate (CBR) packet every one second through the simulation period. Standard performance is investigated in terms of energy consumption, average end to end delay and throughput. Results are categorized according to the duty cycle. In other words, those (BO, SO) combinations that achieve the same duty cycle are depicted in one figure for each performance metric. Duty cycle concept is expressed according to the following formula:

$$D= 2^{SO-BO} * 100\% \qquad (5)$$

As SO value is always lower than the BO value, D can be expressed as:

$$D= 1/2^{BO-SO} *100\% \qquad (6)$$

Hence, $D \in$ {100%, 50%, 25%, 12.5%, 6.24%, 3.13%, 1.56%, 0.78%, 0.39%}.

For each duty cycle in the set, the standard performance is investigated for the seven PANs in order to study the effect of the number of nodes in one piconet, so that we can generalize the results achieved for larger network sizes as the basic building block for any network is seven nodes. However, due to space consideration, we just depict results for 100% and 50% duty cycles and which reflect energy consumption and average end to end delay behaviour.

TABLE 1. QUALNET SIMULATION PARAMETERS FOR SEVEN PAN SCENARIOS

Parameter	Value
Simulator	QualNet 5.2
Physical and MAC	IEEE 802.15.4
Area	80 m * 80 m
Number of nodes	2-8
Transmission range	10 m

Simulation time	1000 s
Channel Frequency	2.4 GHz
Energy model	MICAZ
Antenna Height	0.08
Traffic	CBR
Payload size	50 byte
Arrival Rate	1 second
BO and SO values	1-14

A. *Total Energy Consumption Results (mWh)*

Fig. 1 and Fig. 2 depict total energy consumed in different PAN sizes. PANs work in the 100% and 50% duty cycles which are determined by all possible (BO, SO) beacon frames combinations sent by the PANc. We note that despite the value of SO, combinations with BO values greater than 3 increase BI value. This result in longer time between two successive beacons and thus decreases beacon overhead which contributes in conserving energy. Unfortunately, this is not always the case, since very high SO values starting from 9 increase the possibility for idle listening due to the longer time nodes spend doing nothing which will dissipate energy.

However, increasing the duty cycle by either fixing BO while increasing SO values or fixing SO while decreasing BO values, decreases sleep time portion as SO becomes closer to BO. Unfortunately, this may increase energy consumption at combinations with high SO values (starting from 9) due to the idle listening . However, increasing BO while fixing SO values or decreasing SO while fixing BO values leads to an increased BItherefore, it leads to lower duty cycles. This, on one hand, offers to nodes more time to sleep between active periods, thus helping in conserving energy while on the other hand decreases the need for frequent beacon frames transmission which consequently saves energy. Furthermore, very short CAP compared to the total overall BI will cause frequent CCA deference; hence, nodes shall try to transmit altogether at the beginning of the next superframe. This leads to bothsevere collision and packets re-transmission which consequently increases energy consumption.

It can be noticed that for the 1 s arrival rate, as the duty cycle decreases, there is not much big difference in energy consumption trend. We can say that the dominating energy consumption factor is the idle listening. Or in other words, to conserve energy in a piconet, it is necessary to let PAN to work through a reasonable CAP in a reasonable BI by setting reasonable BO and SO values keeping in mind that we need to avoid all combinations with BO greater than 8. Sleep mode somehow contributes in energy saving but not to that extent. The positive effect of sleep mode in energy consumption is much noticeable in inactive networks which work in very low arrival rates where there is infrequent packets transmission (every hour for example). Actually, for the 1 s arrival rate, inactive period length has much to do with the delay and the throughput behavior but it is not a big energy conserving factor. This will be more clarified in the following sections.

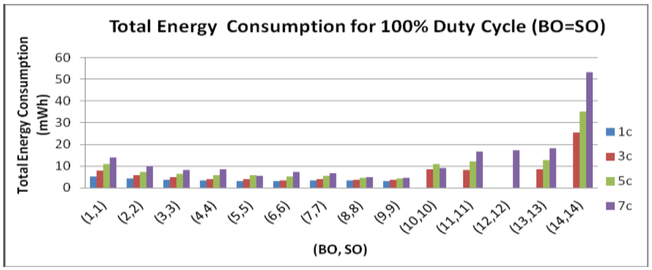


Figure 1. Total energy consumption in PAN works at 100% duty cycle

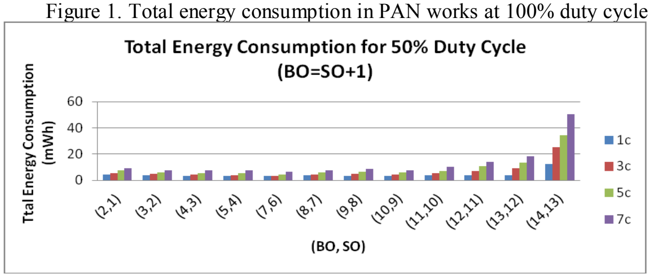


Figure 2. Total energy consumption in PAN works at 50% duty cycle

B. *Average End to End Delay Results(s)*

The second performance metric considered is the delay each received packet suffers during its journey from the source node to the PANc which is why it is called end to end delay. Fig. 3 and Fig. 4 depict average end-to-end delay in different PAN densities which work in 100% and 50% duty cycles determined by the whole possible (BO,SO) beacon frames combinations sent by the PANc. Results revealed that as (BO, SO) values increase, the time during which nodes may access the medium increases, and if there are more than one node want to use the medium, the average delays significantly increase because nodes will go into additional and higher backoff delays, since the backoff exponent should be higher which increases as number of nodes increases.

However, despite the value of SO, as BO increases, nodes that haven't finish their work in the current superframe shall wait until the next beacon frame initiates the active period in order to accomplish the work in the next superrframe. This case is obvious in Fig. 4 where the average delay increases dramatically if it compared to that of Fig. 3. Despite the duty cycle, all combinations with BO and SO values lower than (6, 6) contributes in very low delay (close to 0) because of the short BI which if it increases, delay increases accordingly. Unfortunately, all those combinations with BO=SO values starting from 11 results in a very long BI which on one hand delays the association process, while on the other hand may cause nodes to loose synchronization with the coordinator. Those cases result in a bad throughput as they lessen the number of the successfully transmitted packets. This explains why average delay at such combinations has values close to 0. However, at (14, 14), the first successful association occurred after 3000 s whereas the simulation period occupies 1000s; thus, no PAN activities shall take place during this time which explains the 0 delay at this combination.

In short, whatever is the duty cycle, the average delay behavior is consistent. Increasing the duty cycle on one hand decreases sleep time portion as SO becomes closer to BO. This shall decrease delay, since node shall have enough time to achieve its work during the current CAP and hence will wait less time for the next superframe in order to continue or renew transmitting its packets.

However, increasing BO while fixing SO or decreasing SO while fixing BO values leads to an increased BI and therefore leads to lower duty cycles. This offers nodes more time to sleep between active periods at the expense of completing their work in the current superframe, thus, node shall need to wait for the next CAP which definitely will increase delay as the duty cycle decreases.

In short, for 1 s arrival rate, high (BO, SO) values increase backoff delay which is proportional to the number of nodes, whereas small SO values increase the waiting delay as node shall need to wait for the next CAP to accomplish its work. Despite that the inactive period has not much effect on energy consumption, long sleep time portions will increase delay. However, this may not be the case in inactive applications which work through very low arrival rates, because in the time that low SO values decrease energy consumption dramatically, delay may increase which would obviously occur if node receives packets at the end of its active session. It can be said that for applications that work through low duty cycles, energy can be saved if delay is sacrificed.

C. *Summary*

For the examined CBR application that works through 1s arrival rate, it can be noticed that the closer the values of SO to BO (high duty cycles); the higher the throughput and the lower the delay and nodes percentage of collision. However, this has not much to do with energy consumption. Keeping in mind that very high values of BO on one hand will delay the association time which will adversely affect the throughput, while on the other hand will increase the delay significantly besides increasing energy due to the idle listening. Meanwhile, as number of nodes increases, collision increases, which can be noticed in short CAP, thereby, in order to decrease collision rate, CAP should be increased.

Regarding to the results achieved for the whole 9 possible duty cycles, it is revealed that for 1s arrival rate application, the Rang of combinations that is possible to achieve the optimal performance is aligned between {(6, 6) and (8, 8)}. If the standard is allowed to work in 100% duty cycle for example, its optimal performance achieved at (6, 6). However, for those applications that work in 50% duty cycle, it needs to work at (7, 6), whereas for applications that work in 25% duty cycle, it is preferred for them to work through (8, 6). Thus, if our aim is to allow the 1s arrival rate applications to work in its near optimal performance, it is preferred to allow them to work in 25%, 50% or 100% duty cycles. Unfortunately, lower duty cycles achieve bad performance in terms of the three metrics irrespective of the (BO, SO) combination, this means that unless it is necessary to follow, it is preferred to avoid such duty cycles. More generally, irrespective of the duty cycle, all those combinations with BO greater than 8 must be unconsidered when we need to implement any application that works in 1s arrival rate. Hence, (8,8) can be expressed as the cut-off value after which PAN performance drops dramatically and which may varies according the arrival rate the application may work with.

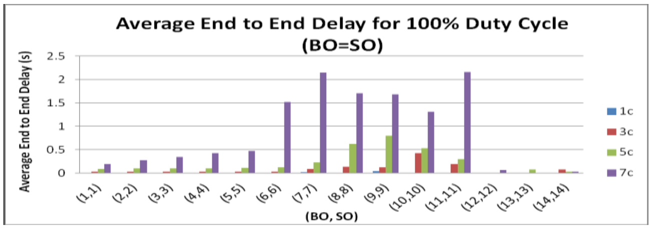


Figure 3. Average end to end delay for PAN works at 100% duty cycle.

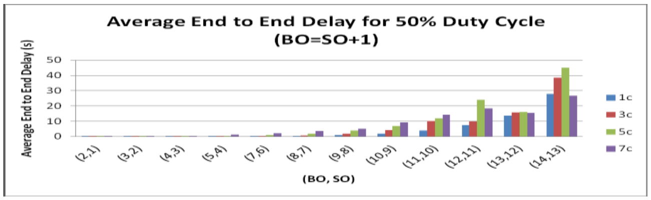


Figure 4. Average end to end delay for PAN works at 50% duty cycle

IV. PROPOSED ALGORITHM

Previous analysis indicates that in all PAN scenarios, despite that decreasing the duty cycle affects applications total energy differently, both delay and throughput behaviors are consistence, that is, as the duty cycle decreases by either fixing BO while decreasing SO or fixing SO while decreasing SO, both metrics are adversely affected. This result is two-folded; first, if we want to improve performance, we need to avoid low duty cycles by having both BO and SO values close to each other's as much as possible. In other words, we have to increase the duty cycle; this can be achieved by fixing BO while increasing SO. Moreover, enhancing delay behavior through increasing duty cycle shall improve throughput behavior accordingly. This note presents the base from where we start our new algorithm.

Following the new algorithm, PANc regularly estimates total end to end delay each node suffered so far along with the number of packets received from that node at that moment. PANc can then estimate nodes average end to end delay which will be its performance criteria according to which SO value shall dynamically change. Experiments revealed that the most reasonable number of packets after which the checking process is done by PANc is 5; hence, checking process is done every 5 packets. According to the results achieved, PANc decides if it should increase SO. In other words, if the new estimated average delay is checked to be worse than that of the previous calculated one, then PANc shall increase SO value, otherwise, do nothing. If SO increased, its value should not exceed that of BO value, and if so, it shall reset to the original SO value and restart the overall process again. This adaptive algorithm is illustrated in Fig. 5.

V. PERFORMANCE EVALUATION

The new proposed adaptive algorithm performance is evaluated along with that of the MAC original algorithm. Simulation took place on a PAN that consists of five nodes where four CBR applications are sent by the four nodes to the PANc every 1s. Recall that for 1s arrival rate application, simulation analysis investigated revealed that the (BO, SO) combination that achieves the near optimal performance in terms of the three metrics and after which the performance is adversely affected is (8, 8). Hence, our scope is limited to all possible (BO, SO) combinations which are less than and including (8, 8). Selected combinations then dynamically change according to the network performance. Other simulation parameters are identical to those presented in Table 1. Fig. 6, Fig. 7 and Fig. 8 depict performance evaluation results for the proposed algorithm compared with the original MAC algorithm in terms of the three metrics starting from average end to end delay behavior.

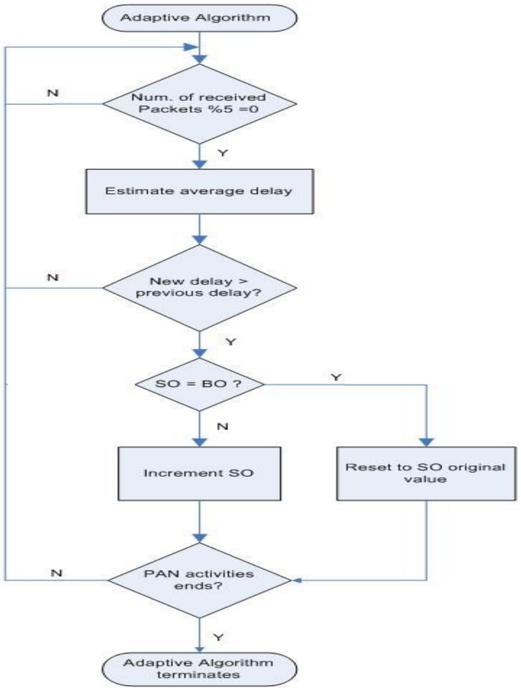

Adaptive Algorithm
N
Num. of received Packets %5 =0
Y
Estimate average delay
N
New delay > previous delay?
Y
SO = BO ?
Y
N
Increment SO
Reset to SO original value
N
PAN activities ends?
Y
Adaptive Algorithm terminates

Figure 5. The new adaptive algorithm

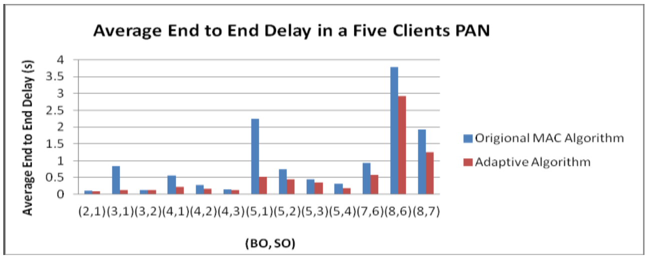


Figure 6. Average delay for the proposed and the original algorithm

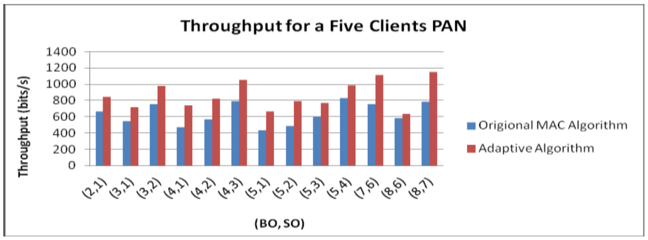


Figure 7. Throughput for the proposed and the original algorithm

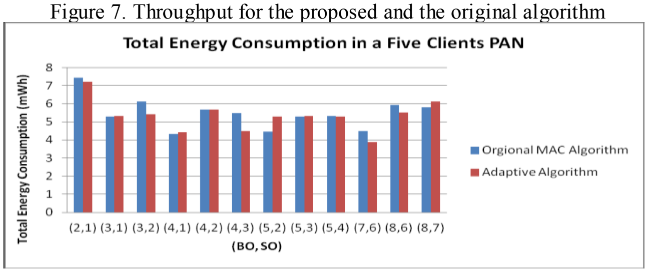


Figure 8. Total energy following the proposed algorithm along with origional algorithm

A. *Average End to End Delay (s)*

Fig. 6 depicts that the adaptive algorithm achieves less delay than the original MAC algorithm. This is simply because CAP increased gradually offering more time for nodes to accomplish their work in the current superframe, thereby, there will be less possibility for nodes to wait for the next beacon frame in order to accomplish the uncompleted work, thus, delay decreased, keeping in mind that BI remains constant as BO remains fixed.

B. *Throughput (bits/s)*

As it is expected, Fig. 7 depicts that the throughput for new adaptive algorithm outperforms that of the original one. In the adaptive algorithm, nodes have more time to perform activities and send packets in the current active session as SO increases gradually; thus, CAP increases accordingly. This gives nodes more opportunities to finish their work in the current active session which will also decrease collision rate and hence increases the throughput.

C. *Total Energy Consumption (mWh)*

It is noticed from Fig. 8 that the adaptive algorithm outperforms the original MAC algorithm at most (BO, SO) tested combinations, especially at some combinations which contribute in 50% duty cycle. At such combinations, delay is somehow moderate which will partially cause increment in SO value; thus, it allows PAN to work most of the time in 50% duty cycle; this offers to nodes more opportunities to sleep half of the period, while limiting idle listening. However, for combinations which contribute in very low duty cycles, such as (5, 2), delay is initially estimated to be very bad, hence, SO shall enhance delay accordingly, this will decrease sleep time opportunities while increasing idle listening which in turn increases energy consumption; this explains why energy consumption at such values unfortunately increases. Thereby, the proposed adaptive algorithm performance proved to outperform the original MAC algorithm in terms of average delay and throughput at all the tested (BO, SO) combinations while decreasing energy consumption at most of them. The new adaptive algorithm reduces energy consumption up to 7%, decreases average delay by 26% while increasing throughput by 16%.

VI. CONCLUSION AND FUTURE WORK

The chosen (BO, SO) combination has a direct impact on the performance of WSN applications that follows the IEEE802.15.4. Low SO values, compared to that of BO, serves low duty cycle applications whereas the closer SO to BO the higher the duty cycle. Thus, IEEE802.15.4 standard can support up to 9 duty cycles through just manipulating BO and SO parameters. In order to achieve optimal performance, different types of applications have to be aware of which (BO, SO) combination to use to achieve the best performance. For 1 s arrival rate application, the rang of combinations that is possible to achieve the best performance is aligned between {(6,6) and (8,8)}. The stranded can achieve its optimality if it is allowed to support 25%, 50% and 100% duty cycles applications. Lower duty cycles achieve bad performance irrespective of the (BO, SO) combination, which means that unless it is necessary to follow, it is preferred to avoid such duty cycles. (8, 8) can be expressed as the cut-off combination because other combinations with higher BOs drop the PAN performance dramatically.

The experimental work conducted for the 1 s arrival rate application revealed that improving some metrics performance may be scarified in order to allow others achieved. However, our goal is to achieve high PAN performance in term of all metrics and most importantly the power consumption irrespective of the duty cycle or the arrival rate. The proposed adaptive algorithm proved to improve the standard in terms of the average delay and the throughput at all tested (BO, SO) combinations while decreasing energy consumption at most of them. The new adaptive algorithm reduces energy consumption up to 7%, decreases average delay by 26% while increasing throughput by 16%.

As a future work, increasing the duty cycle will be achieved through following different techniques which manipulates BO values. The same work that is conducted for 1s arrival rate applications, will be investigated for other higher and lower arrival rate applicants in order to decide both the optimal and the cut-off (BO, SO) values. Then the new adaptive algorithm behavior is going to be evaluated for such arrival rates applications. Other than average end to end delay criteria, PANc shall check PAN performance according to energy consumption metric and according to which the new adaptive algorithm shall be controlled.

REFERENCES

[1] IF. Akyildiz, W. Su, Y. Sankarasubramaniam, and A. Cayirci: A survey on sensor networks. Communications Magazine 2002. Atlanta, GA, USA, vol. 40(8), pp. 102-114, 2002.

[2] L. Selavo, A. Wood, Q. Cao, T. Sookoor, H. Liu, A. Srinivasan, and J. Porter," wireless sensor network for environmental research", Proc. the 5th international conference on Embedded networked sensor systems. Sydney, Australia Nov. 2007, pp. 103-116.

[3] A. Koubaa," Promoting Quality of Service in Wireless Sensor Networks", (Submitted for receiving Habilitation Qualification in Computer Science) National School of Engineering, Sfax, Tunisia, 2011.

[4] SC. Ergen, " ZigBee/IEEE 802.15. 4 (Summary)", [Online][accessed April 2013], Available from URL http://pages.cs.wisc.edu/~suman/courses/838/papers/zigbee.pdf.

[5] P. Park, C. Fischione, and KH. Johansson, "Adaptive IEEE 802.15. 4 protocol for energy efficient, reliable and timely communications", Proc. the 9th ACM/IEEE international conference on information processing in sensor networks. Stockholm , April 2010, pp. 327-338.

[6] F. Shu, T. Sakurai, HL. Vu, and M. Zukerman, "Optimizing the IEEE 802.15. 4 MAC", Proc. IEEE Region 10 Conference (TENCON). Hong Kong,Nov. 2006, pp. 1-4.

[7] P. Patro, M. Raina, V. Ganapathy, M. Shamaiah, and C. Thejaswi , "Analysis and improvement of contention access protocol in IEEE 802.15. 4 star network", Proc. Mobile Adhoc and Sensor Systems (MASS 07), IEEE Internatonal Conference. Piza, Italy, Oct. 2007, pp. 1-8.

[8] H. Deng, J. Shen, B. Zhang, J. Zheng, J. Ma, and H. Liu, "Performance Analysis for Optimal Hybrid Medium Access Control in Wireless Sensor Networks", Proc. Global Telecommunications Conference (GLOBECOM 08). LA, USA, Nov. 2008, pp 1-5.

[9] E. Casilari and J.M. Cano-Garcí, "Impact of the Parameterization of IEEE 802.15. 4 Medium Access Layer on the Consumption of ZigBee Sensor Motes", Proc. The Fourth International Conference on Mobile Ubiquitous Computing Systems, Services and Technologies (UBICOMM 2010). Florence, Italy, Oct. 2010, pp. 117-123.

[10] X. Li, CJ. Bleakley, and W. Bober, "Enhanced Beacon-Enabled Mode for improved IEEE 802.15. 4 low data rate performance", Wireless Networks 2012, vol. 18, pp. 59-74.

[11] F. Charfi, and M. Bouyahi, "Performance evaluation of beacon enabled IEEE 802.15.4 under NS2", arXiv preprint arXiv 2012, pp. 1204.1495.

[12] SA. Khan and FA. Khan, "Performance analysis of a zigbee beacon enabled cluster tree network", Proc. Third International Conference on Electrical Engineering (ICEE'09). LahoreApril 2009, pp. 1-6.

[13] A. Koubaa and M. Alves, E. Tovar, "A comprehensive simulation study of slotted CSMA/CA for IEEE 802.15. 4 wireless sensor networks", IEEE WFCS, 2006, pp.63-70.

[14] M. Neugebauer, J. Plonnigs, and K. Kabitzsch, "A new beacon order adaptation algorithm for IEEE 802.15. 4 networks", Proc. The Second European Workshop on Wireless Sensor Networks. Ghent, Belgium2005, pp. 302-311.

[15] J. Hoffert, K. Klues, and O. Orijh "Configuring the IEEE 802.15. 4 MAC Layer for Single-sink Wireless Sensor", Washington University in St. Louis, 2005.

Twitter Data Preprocessing for Spam Detection

Myungsook Klassen
Computer Science Dept
California Lutheran University
Thousand Oaks, USA
e-mail:mklassen@clunet.edu

Abstract—**Detecting Twitter spammer accounts using various classification machines learning algorithms was explored from an aspect of data preprocessing techniques. Data normalization, discretization and transformation were methods used for preprocessing in our study. Additionally, attribute reduction was performed by computing correlation coefficients among attributes and by other attribute selection methods to obtain high classification rates with classifiers,such as Support Vector Machine, Neural Networks, J4.8, and Random Forests. When top 24 attributes were selected and used for these classifiers, the overall classification rates obtained were very close in range 84.30% and 89%. There was no unique subset of attributes which performed the best, and there were various different sets of attributes playing important roles.**

Keywords-data preprocessing;spam detection; social network; classification.

I. INTRODUCTION

Twitter6 [17] was started in 2006 by Jack Dorsey as an online social networking and microblogging service for users to send and receive short messages (called tweets) of up to 140 characters. Tweeter's 140-character limit on a message serves modern day busy people's trend of acquiring information in a short and quick way. There is much less mindless minutia to read through short tweets. People can spend 5 to 10 minutes on Twitter to find out fast what is happening in the world.

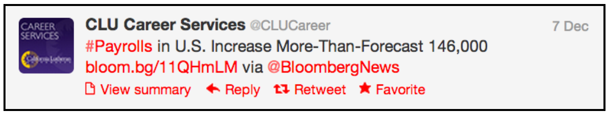
As a result, within the last few years, Twitter has grown to be one of the most popular social network sites with a half billion daily tweets as of October 2012, up from 140 million per day in early 2011. Along with Twitter's growth, spam activities in Twitter have increased and have become a problem. Spamming has been around since the birth of internet and emails, and is not a unique problem with Twitter, but Twitter simply introduces new kinds of spam behavior. Unlike popular social networking service Facebook, or MySpace, anyone can read tweets without a Twitter account, but must register to post tweets. The fact that most accounts are public and can be followed without the user's consent provides spammers with opportunities to easily follow legitimate users.

A recent spamming activity took place during the Russian parliament election December 4, 2011 [8]. For two days after the election, Twitter users posted over 800,000 tweets containing a hashtag related to elections. It turned out nearly half the tweets were spams with unrelated contents, and spam tweets were sent out through fraudulent accounts purchased by a single person in an attempt to disrupt political conversations following the announcement of the election results.

Twitter currently blocks malware by in-house-built heuristics rules using Google's Safebrowsing application programming interface (API) [18] to filter spam activities described in the "Twitter Rules" posted in its web site. Some of spam definitions in the rule are such as an excessive account creation in a short time period, excessive requests to befriend other users, posting misleading links, and posting unrelated updates to a topic using a hashtag "#". Twitter also checks twitter contents with uniform resource locators (URLs) to see if they are on its known harmful sites blacklist database. Harmful sites can be "phishing" sites, sites that download malicious software onto users' computers, or spam sites that request personal information. However, C. Grier et al. [2] show that it takes a few weeks for URLs posted in Twitter to be on its blacklist. In addition to the fact that Twitter itself does to prevent spamming, Twitter relies on users to report spam. Once a report is filed, Twitter investigates it to decide to suspend an account or not. Currently, much research is going on to find a method to detect Twitter spamming in an efficient and automated way. After all, it is not very reliable for the Twitter community to depend on users to identify spams manually based on previous spam activities.

An example of a tweet is shown in Figure 1. It shows a tweet content of a Twitter user "CLU Career Services" with a Twitter ID "CLUCareer". When a Twitter User name or a Twitter user ID is clicked, its public profile page shows a full name, a location, a web page, a short bio, along with tweet contents, the number of tweets, the total number of followers and their Twitter user names, the total number of people the user is following, and Twitter user names of people the user is following. The tweet content in Figure 1 contains a shortened link URL "bloom.bg/11QHmLM" which points to an article page at [19]. Such shortened URLs allow users to post a message within 140 characters, but hide the source URLs, thus providing an easy opportunity for malicious users to phish and spam. This tweet message also contains a topic "Payrolls" which is identified with hashtag "#" in front of it and a *mention* to "BloomerbergNews" user with "@" symbol in front of it.

All this information can be gathered using Twitter API by crawling the Twitter web site. Using these collected "raw" data, different attributes, either content attributes or user behavior attributes [13] can be created.



CLU Career Services @CLUCareer 7 Dec

#Payrolls in U.S. Increase More-Than-Forecast 146,000 bloom.bg/11QHmLM via @BloombergNews

Figure 1. Tweet example

The number of followers is an example of user behavior attributes while the number of URLs in tweets is a content attribute.

To identify spam tweets, classification machine learning methods which consist of a training (or learning) process and a testing process can be applied. During the training process, learning takes place to generalize information from a given data set which contains a large number of attributes. Often, using too many attributes may cause overfitting of data during training which can hinder classifiers from classifying "new" data correctly. Using too few attributes may not be powerful enough to generalize characteristics of data.

It is critical to use important and relevant attributes and remove redundant and irrelevant ones for a chosen machine learning algorithm to obtain high classification rates. Detecting spam tweets correctly not only provides a better Twitter social network environment in general, but also reduces the chance to anger legitimate users by mistakenly labeling them otherwise.

A desirable case is to use a small number of attributes which can distinguish data in one class from those in another class. Data with a small number of attributes executes fast during a training process and ultimately allows the machine learning system to make a fast classification decision with new data. This is critical in a real time application situation. It would be desirable to have an automated machine learning system which can detect Twitter spams in real time at a fast speed and alert the Twitter authority.

Normalization and discretization of numeric attribute data are widely used preprocessing methods. Discretization eliminates small data observation variations or errors while normalization is to avoid attributes in greater numeric ranges dominating those in smaller numeric ranges [6], thus potentially increasing the performance of classifiers. Data transformation is to map data value into another using a mathematical linear or nonlinear function to capture relationships, if any, between attributes.

In this paper, we analyze an impact of preprocessing of Twitter data for a spam detection task. More specifically the purpose is

- to evaluate the impact of using different attributes of Twitter data on different classifiers.
- to evaluate an impact of using a small number of attributes and a large number of attributes.
- to evaluate if preprocessing steps, such as discretization, normalization and transformation with Twitter data may increase classification rates.
- to evaluate if all these steps are consistently needed for all classifiers used.

The rest of the paper is organized as follows. Section II describes related work on spam detection. Section III describes data used, classifier methods, and evaluation methods. Section IV presents actual experiments and results, followed by the last section, Section V, to summarize the study results and future work.

II. RELATED WORK

In this section, we first discuss previous studies on Twitter social media in general to get insight of Twitter data characteristics, then discuss previous studies on machine learning algorithms approaches for detecting and measuring Twitter spam.

Thomas et al. [3] analyzed over one million suspended Twitter accounts to characterize the behavior and lifetime of spam accounts, to understand how spammers abuse legitimate web services such as URL shortening services by exploring spam affiliate programs and market place of illegitimate programs run by spammers. They report that 77% of spam accounts are suspended within the first day of their first tweet and 92% of accounts within three days. Less than 9% of accounts form follower/following relationships with regular users, 52% of spam accounts use unsolicited *mentions*, and 17% used hashtags in the messages with unrelated contents for trend *topic* search. They also report that 89% of spam accounts have fewer than 10 followers.

Link is an important feature to detect spams or to infer user opinions in sentiment analysis [12]. The computed rank depends on the user's connectivity in the social graph. The more followers a user has, the more likely his/her tweets are to be ranked high. Spammers attempt to use this ranking score by acquiring links in Twitter-they follow other users and try to get others to follow them as a courtesy of "social etiquette". Cha et al. [11] and Ghosh et al. [5] studied links in Twitter and reported that popular users who have many followers are not necessarily influential in terms of spawning retweets or mentions. Kwak et al. [4] collected 41.7 million users to study follower-following topology of Twitter and reported that influence inferred from the number of followers and from the popularity of one's tweet do not match. Their findings also show that ranking users by the number of followers matches with results computed by PageRank while ranking by retweets differs from PageRank and from the number of followers and any retweeted tweet reached an average 1000 users regardless of the number of followers of the original tweet account.

Previous study at U.C. Berkeley [14] shows that 45% of users on a social network site readily click on URLs without doubt. Grier et al. [7] collected over 400 public tweets and reported that 8% of 25 million unique URLs posted to Twitter point to phishing, malware, and scam. They reported that the click through rate is 0.13% which is almost twice higher than the email spam click through rate previously published and 80% of clicks occur within the first day of a spam URL appearing on Twitter.

Alex Wang [13] crawled Twitter and collected 29847 users with around 500K tweets and 49M follower/friends relationships. He manually labeled each tweet either as spam or non spam and found that only 1% is spam account. A graph based attribute *reputation* and content based attributes such as existence of duplicate tweets, the number of HTTP links, the number of replies/mentions, the number of tweets with trending topics are used with Bayesian classifier for spam detection and 89% overall classification rate was reported.

McCord and Chuah [14] collected 1000 Twitter user accounts and extracted the following attributes: distribution of Tweets over 24 hours, the number of friends, the number of followers, the number of URLs, the number of replies/mentions, weighted keywords, the number of retweets, and the number of hashtags and ran 4 different classifiers with theses attribute values. They reported that the Random Forest performs the best among 4 classifiers with an overall precision value of 0.957.

Benevenuto et al. [10] gathered a large Twitter data set related to three trend topics and extracted 39 contents attributes and 23 user behaviors attributes which were used with Support Vector Machine (SVM) classifier to detect Twitter spammers. Further description of data can be found at Section III, since our study was conducted using this set of data. They reported classification rates of 70.1% and 96.4% for spam class and non spam class, respectively.

Twitter spammers are known to employ automation to publish tweets. Zhang et al. [16] presented a technique to detect automated twitter content updates. They tested 19436 accounts and reported that 16% exhibit highly automated behavior and verified accounts, most-followed accounts, and followers of the most followed account all have lower automation rates of 6.9%, 12% and 4.2%, respectively.

III. EXPERIMENTAL SETUP

A. *Data set*

Data from Benevenuto et al. [10] was used as a basis for this study. In his work, Twitter was crawled to collect tweets with three most trendy topics at the time in August 2009 and 1065 legitimate accounts and 355 spam accounts were used for his study. Data contains thirty nine content attributes and twenty three user behaviors attributes, all numeric values, from the raw tweet information. Content attributes are a fraction of followings: tweets replied, tweets with spam words, tweets with URLs, along with the mean, median, min, and max of the followings: the number of hashtags per words on each tweet, URLs per word on each tweet, characters per tweet, hashtags per tweet, mentions per tweet, numeric characters per tweet, URLs per tweet, words per tweet and times a tweet is retweeted.

Two additional attributes which are computed from existing attributes are added to the data for our study. One is reputation defined by Wang [13] as a ratio of the number of followers to the sum of the number of followers and the number of followees. The second is *Influence factor*, which is defined for this study as a ratio of the sum of a number of times mentioned and a number of times a user was mentioned to the sum of a number of times mentioned , a number of times a user mentioned, and a number of times a user replied.

B. *Methods*

Four classifiers, SVM, random forest (RF), a multi layer back propagation neural networks and J4.8 decision tree implemented in the open source data mining suites WEKA were used in our experiments. WEKA [20] is a data mining software developed at the University of Waikato, New Zealand. For SVM, a program *grid.py* from the libSVM implementation site [6] was used to select two important parameters, C a penalty parameter of an error term and gamma a RBF kernel function coefficient. These values are used for SVM in WEKA.

C. *Evaluations*

The ten cross validation is used to measure the generalization performance of classifiers used in this research. The method first partitions data into 10 equal sized segments and in each iteration, 9 different segments are used for training and 1 remaining segment is used for testing. This repeats 10 times and an average of 10 results from testing segment is computed.

Classifier performance results are discussed using values derived from a confusion matrix. TABLE I shows a confusion matrix of two classes.

TABLE I: CONFUSION MATRIX

	Predicted Class1	Predicted Class2
Actual Class1	a	b
Actual Class2	c	d

True Positive (TP) for class 1 is a/(a+c) and False Positive(FP) for class 1 is c/(c+d). Precision for class 1 P= a/(a+c) is the ratio of the number of data predicted correctly to the total predicted as class 1. Recall for class 1 R= a/(a+b) is the ratio of the number of data correctly predicted to the number of data in class 1. TP, FP, P and R for class 2 are similarly defined. A classification rate or an average weighted TP rate in WEKA is defined as the ratio of the number of correctly predicted data to the total number of data in both classes, (a+d)/(a+b+c+d). F-measure is a weighted average of the precision P and recall R to measure of a test's accuracy and is defined as 2* P*R/(P+R).

IV. EXPERIMENTS AND RESULTS

A. *Normalization*

The original data set has a vast range of attribute values. Seventeen attributes such as a *fraction of tweets replied* are in a range between 0 and 1 while most of content based features such as the number of followers are in a range 0 and over 40000. The age of an account and an elapsed time between tweets are measured in seconds so values range between 0 and 87,000,000. We investigated if attributes in greater numeric data ranges dominate their significance in learning and produce inaccurate classification. Experiments without normalization and with normalization in a range -1 and 1 were performed with libSVM and results shown in TABLE II. It also shows that without normalizing data, the spam class was predicted very poorly with only 6.8% TP rate and the overall classification rate of 68.9%. With data normalization, not only the classification rate went up significantly to 88.3%, but also more importantly the spam class TP values went up to 75.2%! When data is normalized between 0 and 1, similar results to those with normalization between -1 and 1 are obtained.

Our next experiment is to evaluate sensitivity of classifiers with data normalization to obtain high classification rates. Classification rates obtained with 4 classifiers using both original data set and normalized data set and results are presented below in TABLE III. Without data normalization, both the multi-layer neural networks and libSVM show a low performance with 68.68% and 70.42% classification rates respectively, while J48 and Random Forests consistently perform well regardless of data normalization.

B. *Manual atttributes selection*

A manual attribute selection process used for this study is based on the notion that an attribute with a high correlation with the "class" attribute, but with a low correlation with other attributes is a "good" attribute. Correlation between the "class" attribute and an attribute being reviewed is computed for all attributes. For instance, the "*existence of spam words in the screen name"* attribute has -0.01085 correlation values with the "class" attribute, so this attribute is considered not very useful and is eliminated from the attribute list. Attributes *min of the number of URLs per tweet* and *max of the number of URLs per tweet* showed a similar trend and as a result, they are eliminated from the attribute list. Similar steps were taken with all other attributes for selecting good attributes.

Redundancy of attributes is evaluated by their correlation values which are shown in TABLE III. When there is high correlation between two attributes, one attribute is eliminated and if there is no strong correlation between two attributes, both are kept. In the case of the number of words per tweet and the number of characters per tweet, there is a little correlation so both are kept. After this manual process, 24 attributes are selected.

C. *Using WEKA attribute selection methods*

Chi Squared Attribute selection, Filtered Attribute selection, Info Gain Attribute evaluation, Gain Ratio Attribute Eval and oneR AttributeEval with Ranker selection method were used to rank original 62 attributes. Top 10 ranking from Filtered Attribute Evaluation and Info Gain Attribute Evaluation are quite similar, but are quite Different from those obtained by 3 others and top 10 rankings from these 3 methods have only a few attributes in common. Simply there is not a group of top 10 attributes common for all selection methods. TABLE V shows three attributes, *the number of followers per followees, fraction of tweets replied, and the number of times the user replied* are on the top 10 ranking attributes for all 5 selection methods. When a comparison is made for top 24 attributes, a situation is worse-there is less percentage of common attributes selected by all 5 methods.

TABLE II. BETTER PERFORMANCE OF NORMALIZED DATA WITH SVM

Without scaling				
	TP rate	FP rate	Precision	Recall
No spam class	1	0.932	0.682	1
Spam class	0.068	0	1	0.068
Classification rate	0.689	0.622	0.788	0.689
Data scaled [-1,1]				
	TP rate	FP rate	Precision	Recall
No spam class	0.947	0.248	0.885	0.947
Spam class	0.752	0.053	0.876	0.752
Classification rate	0.883	0.183	0.882	0.883

TABLE III. PAIRED-T-TEST OF F MEASURE OF CLASSIFICATION RATES

	libSVM	**MultiLayer NN**	**J48**	**Random Forest**
Original data	68.68	70.42	83.22	88.55
Normalized data	87.59	87.47	85.54	88.16

TABLE IV. CORRELATION COEFFICIENTS OF SOME PAIRED ATTRIBUTES

Two Attributes selected for correlation inspection	**Correlation coefficients**
number of hashtags per word on each tweet(mean). number of hashtags per tweet(mean).	0.886909
number of posted tweets per day(mean). number of posted tweets per week(mean).	0.742563
number of followees of a user's followers. number of followees.	0.903385
number of followees of a user's followers. number of followers.	0.862586
number of words per tweet (mean). number of characters per tweet(mean).	0.349

However, despite different rankings of attributes, when top 24 attributes were used with classifiers, all of them showed compatible classification rates, slightly lower than results obtained when manually selected 24 attributes were used as shown in TABLE V.

D. Data discretization

This process reduces the number of possible values for attributes with continuous values. Equal-interval binning was performed. An objective is to measure an effectiveness of data discretization on classifier performance. With the original data set of 62 attributes, attributes with a very large data range such as the number of followees, the number of followers, number of tweets, and age of the user account are discretized into 10 bins and results such as TP, P and F values of spam class and no spam class along with the overall classification rate are as shown in TABLE VII. Classifier performances of all classifiers are compatible to those obtained from data normalization.

V. CONCLUSION AND FUTURE WORK

In this paper, we explore attribute reduction and data preprocessing such as data normalization and discretization from the aspect of Twitter spam detection using various machine learning algorithms. Four classifiers SVM, back propagation multilayer neural network, decision tree J4.8 and random forest.

TABLE V. COMMON TOP TEN RANKING ATTRIBUTES SELECTED

	attributes
Attributes ranked in a top 10 by all selection methods.	• Number of followers per followees. • Fraction of tweets replied. • Number of times the user replied.
Attributes ranked in a top 10 by 4 selection methods.	• Fraction of tweets with URLs. • Average number of URLs on each tweet. • Number of times user replied. • age of the user account.
Attributes ranked in a top 10 by 3 selection methods.	• Average number of hashtags per word on each tweet.

TABLE VI. CLASSIFICATION WITH DIFFERENT ATTRIBUTES OF DIFFERENT CLASSIFIERS

	libSVM	**MultiLayer NN**	**J48**	**Random Forest**
Manually picked 24 attributes	87.90	87.72	85.98	88.39
ChiSquare	84.57	87.83	86.19	87.95
GainRatioAttributeEval	86.83	87.71	86.39	87.90
InfoGainAttributeEval	86.12	87.23	86.12	88.91
OneRAttributeEval	86.23	87.72	86.35	87.42

Using correlation coefficients, good attributes are selected manually and are compared to those obtained with 5 WEKA attribute selection methods. With correlation coefficient and along with Twitter data structure information, a total of 24 attributes were selected and results obtained with 4 classifiers were very close, or slightly higher values obtained with attributes selected by WEKA attribute selection methods.

TABLE VII. CLASSIFICATION RESULTS (%) FROM DATA DISCRETIZATION

	Overall TP	**Nospam TP**	**Spam TP**	**Nospam P**	**Spam P**	**Nospam F**	**Spam F**
libSVM	87.32	0.962	0.696	0.863	0.901	0.91	0.785
ML NN	87.32	0.942	0.735	0.877	0.864	0.908	0.795
J4.8	86.29	0.944	0.701	0.863	0.862	0.902	0.773
RF	88.26	0.965	0.718	0.873	0.911	0.916	0.803

The newly introduced attribute *influence factor* has a much higher correlation coefficient (> 0.35) with the ‘class” attribute while its component attributes, *the number of times mentioned, the number of times the user was replied, the number of times the user replied,* has correlation coefficient values 0.11449, 0.149, and 0.256 respectively with the class attribute. This means the better attribute “influence factor” was created through a linear data transformation of existing three attributes to boost the classifier performance.

When the original data was used initially, the overall classification rate obtained with libSVM was 68.9% with TP value of no spam class= 1 and TP value of spam class =0.068 which is extremely low. This is a much skewed classification result that most of spam accounts were misclassified as non spam while non spam accounts were 100% correctly classified. The probable explanation for this is that attributes which represent characteristics of the spam class have small data ranges compared to other attributes and thus can’t contribute to correctly classify the spam class. With back propagation multi layer neural network, the similar results and explanation can be applied. Both libSVM and multilayer back propagation neural network consider and add up all attributes and as a result, scaling becomes important. In tree decision algorithm such as J48 and Random Forest, each attribute is individually considered for information gain, so normalization is not an important factor and such is the case with twitter data.

Data discretization produced similar results to those obtained by data normalization process, classification rates of libSVM and multi layer neural networks increased significantly and TP value of the spam class increased dramatically over to 0.7 from 0.068. We also report that when the number of bins was increased from 10 to 15, classification performance changed less than 5%.

Regardless of all methods applied, the highest TP of the spam class is relatively low with a value of 0.752 while the highest TP obtained in our experiments for the non spam class is quite high with a value of 0.964. This is a slightly higher value than 0.701 reported by Benevenuto [10] where data for our study came from.

A close comparison with two other works [13][14], which used similar attributes and reported higher spam class TP rates is further discussed to hopefully understand why our data shows low non spam class TP rates and to continue this study in the future work.

McCord [14] collected data at random without considering trendy topics while Benevenuto did with three specific popular topics at the time Twitter data was gathered. The two unique attribute used in McCord's work are the *word weight metric* which is a difference between the weight of spam words and the weight of legitimate words in tweets. The sum of all weights is used as a "word weight metric" attribute. This weight parameter controls a probability that a word can be in a list of spam words and a probability that it can be in the regular word list. And the other unique attribute used for their study is the time of a day a tweet was posted. The rationale of this attribute is that spammers work at night.

Wang [13] reported TP value 0.89 of the spam class and the overall classification rate 91.7% and the recall value R 0.917 with Naïve Bayesian classifier. With SVM, neural networks, and J48 decision tree, 100%, 100%, and 66.7% overall classification rates respectively reported. But the reported recall rates for three classifiers are very low, 0.333, 0.417, 0.25 for decision tree, neural network, SVM respectively. In his work, like McCord's work, random Twitter accounts were collected without considering trendy topics. And the unique attributes used in his work are reputation which we used for our study and the total number of duplicate tweets which is computed by using Levenshtein distance. The rationale is that spammers use different user names to post the same contents. An observation worth mentioning is that the number of hashtags for the spam class in his study is lower than those for the non spam class. Many spam accounts have less than or equal to 2 average hashtags while non spam accounts have anywhere from 0 to 20 (an estimate of an average number of hashtags by a quick visual inspection of Figure 7d in his work is about 7).

This is quite contrast to what Benevenuto's data shows regarding the hashtags that spammers post much higher fraction of hashtags per tweet. This contrast may come from the fact that data crawled from Twitter by Benevenuto et al. was using trendy topics while Wang's was gathered at random without using any trendy topic. This comparison suggests that spammers use more hashtags to capture legitimate users attention when there are hot trendy topics being discussed but when there is little hot trendy topics being discussed among Twitter users, usage of hashtags among spammers and among non spammers is not much different .

In conclusion, our study shows that normalization, transformation and discretization improve Twitter spam/no spam classification rates, especially the spam class when libSVM and back propagation multi layer neural networks were used. And our study demonstrates that when using a smaller number of attributes selected manually using correlation coefficient, equally high classification rates were obtained. This is an important finding for a real time detection of spams. Further investigation of Twitter characteristics is needed to understand why the spam class TP value is not as high as we hope to.

REFERENCES

[1] D. Terdiman, "Report: Twitter hits half a billion tweets per day," http://news.cnet.com, December 2012. Retrieved: January, 2013.

[2] C. Grier, K. Thomas, V. Paxson, and M. Zhang, "@spam: The underground on 140 characters or less," Proceedings of CCS' 10 Conference, pp. 27-37, October 2010.

[3] K. Thomas, C. Grier, V. Paxson, and D. Song, "Suspended Accounts in Retrospect: An analysis of Twitter spam," Proceedings of IMC conference, pp. 243-256, November 2011.

[4] H. Kwak, C. Lee, H. Park, and S. Moon, "What is Twitter,a social network or a news media?" Proceedings of WWW2010 conference, pp. 591-600, April 2010.

[5] S. Ghosh, B. Viswanath, F. Kooti, N. Sharma, G. Korlam, F. Benevenuto, N. Ganguly, and K. Gummadi, "Understanding and combating link farming in the twitter social network," Proceedings of WWW2012 conference, pp. 61- 70, April 2012.

[6] C. Hsu, C. Chuan, and C. Lin, "A Practical guide to support vector classification," http://www.csie.ntu.edu.tw/~cjlin. Retrieved: January, 2013.

[7] C. Grier, K. Thomas, V. Paxson, and M. Zhang, "@spam:the underground on 140 characters or less," Proceedings of CCS' 10 conference, pp. 27-37, October 2010.

[8] http://www.icsi.berkeley.edu/icsi/gazette/2012/05/twitter-spam. Retrieved: December, 2012.

[9] W.S. Sarle. Neural Network FAQ, 1997, ftp://ftp.sas.com/pub/neural/FAQ.html. Retrieved: December, 2012.

[10] F. Benevenuto, G. Magno, T. Rodrigues, and V. Almeida, "Detecting Spammers on Twitter," Proceedings of CEAS 2010 conference, pp. 21-30, July 2010.

[11] M. Cha, H. Haddadi, F. Benevenuto, and K. Gummadi, "Measuring user influence in twitter: The million follower fallacy, "Proceedings of ICWSM 2010 conference, pp. 10-17, May 2010.

[12] J. Rabelo, R. Prudencio, and F. Barros, "Using link structure to infer opinions in social networks," Proceedings of SMC conference, pp. 681- 685, October 2012.

[13] A. Wang, "Don't follow me: spam detection in twitter," Proceedings of 5th international conference on security and cryptography, pp. 142-151, July 2010.

[14] M. McCord, and M. Chuah, "Spam detection on twitter using traditional classifiers," Proceedings of international conference on autonomic and trusted computing (ATC) conference, pp. 175-186, September 2011.

[15] U. C. Berkeley. Twitter Spam. International computer science institute (ICSI) ICSI Gazette, May 2012.

[16] C. Zhang, and V. Paxson, "Detecting and analyzing automated activity on twitter," Proceedings of PAM conference, pp. 102- 111, March 2011.

[17] http://www.twitter.com. Retrieved: December, 2012.

[18] https://developers.google.com/safe-browsing/. Retrieved: January, 2013.

[19] http://www.bloomberg.com/news/2012-12-07/. Retrieved: December, 2012.

[20] http://www.cs.waikato.ac.nz/ml/weka/. Retrieved: December, 2012.

Exploring HADOOP as a Platform for Distributed Association Rule Mining

Shravanth Oruganti, Qin Ding, Nasseh Tabrizi
Department of Computer Science,
East Carolina University,
Greenville, North Carolina, USA
o.shravanth@alumni.ecu.edu, dingq@ecu.edu, tabrizim@ecu.edu

Abstract **- Association rule mining is one of the important data mining techniques. Association rule mining is used to discover associations between different items in large datasets. The Apriori algorithm for association rule mining forms the basis for most other association rule mining algorithms. The original Apriori algorithm runs on a single node or computer. This limits the algorithm's capability to run on large datasets due to the limited computational resources available. There have been various studies for parallelizing the algorithm. In this paper, Apache Hadoop was chosen as the distributed framework to implement the Apriori algorithm and to evaluate the performance of the algorithm on Hadoop. The Apriori algorithm was modified to be run on Hadoop. Performance analysis shows that Hadoop is a promising platform for distributed association rule mining.**

Keywords - Cloud computing; association rule mining; data mining; Hadoop.

I. INTRODUCTION

Business Intelligence has become an integral part of many successful organizations. Analyzing data and making decisions based upon the analysis is very important for an organization's growth. Data mining techniques help analyze the substantial data available to assist in decision-making. Among the most frequently used data mining techniques, association rule mining is a very important one. For example, in market analysis, association rule mining helps identify what items are purchased together by customers and generate interesting rules (depending on the measure of interestingness selected) based on the transactional data. Many approaches [2,3,10,12,14] have been proposed to mine association rules but a majority of them depend on the Apriori algorithm as the basis [2]. Due to huge size of the datasets, running these algorithms and mining association rules on a single computer is not very efficient because it is limited by the processor capacity, RAM, storage, and various other factors. Hence, it is necessary to develop distributed algorithms to perform association rule mining. Performing large-scale computing and data mining is one of the issues in future computing.

It was observed that most algorithms for association rule mining ran out of memory even for small datasets and a small support count. There are some parallelized algorithms for association rule mining, but all of them handle the communication in the network, load balancing or other distributed tasks [1,4,5,6,9,13,15,17,18]. The goal of this work was to see if there would be a distributed framework where data intensive tasks could be performed without the overhead of the programmer managing the distributed part of the system. Apache Hadoop was shown to be a good framework for this task [16]. Very little research has been done on implementing association rule mining algorithms on Hadoop, although there are a few studies on implementing association rule mining using cloud computing [7,11,16]. For this study, Apache Hadoop was chosen as the distributed framework to implement the Apriori algorithm [2] and to evaluate the performance of the algorithm on Hadoop.

The rest of the paper is organized as follows. Section II introduces the background related to association rule mining and Hadoop. Section III details the proposed framework, algorithm, and implementation. Section IV presents evaluation followed by the conclusion and future work in Section IV.

II. BACKGROUND

This section introduces the concepts of association rule mining, the original Apriori algorithm for association rule mining, previous approaches to parallelize Apriori algorithm, the MapReduce framework, and Hadoop.

A. Association Rule Mining

Association rule mining was originally proposed for Market Basket Analysis [2]. By searching for frequent patterns in transactional data sets, interesting associations and correlations between item sets in transactional and relational databases may be discovered.

Market Basket Analysis is a modeling technique based upon the theory that if you purchase a particular group of items, you are more likely to purchase another group of items. For example, if a customer purchases some pizza dough, it is more likely he/she will also purchase some pizza sauce. Placing them at the same aisle in the store or even bundling them together could help increase profits. Just the knowledge that customers often purchase certain items in groups could open up new possibilities in businesses.

In the aforementioned example, the set of items a customer purchases is referred to as an itemset, and market basket analysis seeks to find relationships between itemsets. Typically the relationship will be in the form of a rule, such as: customer purchases pizza dough → customer purchases pizza sauce.

Copyright (c) IARIA, 2013. ISBN: 978-1-61208-272-1

There are two measures, support and confidence, which can be used to measure the interestingness of the rules. Support indicates the frequency of the occurring pattern while confidence measures the strength of the association.

B. Apriori Algorithm

The Apriori algorithm [2] has been the basis for many other algorithms in association rule mining. Subsequently, there were many modifications and other algorithms to mine frequent patterns in data [3,10,12,14]. A brief summary of the Apriori algorithm is given below.

The Apriori algorithm is an iterative approach to generate frequent itemsets based upon a user provided minimum support and confidence. Candidate itemsets of size k are generated based upon frequent itemsets of size k-1. The basic algorithm works as follows (Fig. 1):

```
$C_k$: Candidate itemset of size k
$L_k$: frequent itemset of size k

$L_1$= {frequent items of size 1};
for (k = 1; $L_k$! =∅; k++) do begin
    $C_{k+1}$= candidates generated from $L_k$;
    for each transaction t in database do
        Increment the count of all candidates in $C_{k+1}$ that are
            contained in t
    $L_{k+1}$= candidates in $C_{k+1}$ with min_support
end
return $\cup_k L_k$;
```

Figure 1. Apriori algorithm

A major drawback of this algorithm is the high I/O costs. The database needs to be scanned during each iteration, which is expensive. With huge datasets, this may consume significant system resources in order to scan and hold the transactions in memory.

C. Parallel versions of Apriori

The authors of the Apriori algorithm also proposed three parallelized versions of the algorithm to run on multiple nodes [1], including Count Distribution, Data Distribution, and Candidate Distribution algorithms.

The Count Distribution algorithm uses a simple principle of allowing redundant computations in parallel on otherwise idle processors to avoid communication. The basic idea is that each processor will scan the local data asynchronously in parallel to do the local counting, but the candidate itemsets that each processor counts are identical except during the first pass. At the end of each pass, they must synchronize to calculate the global counts. Additional details may be found in [1].

The main disadvantage of the Count Distribution algorithm is that it does not exploit the aggregate memory of the cluster. The Data Distribution algorithm attempts to address this issue. Unlike the Count Distribution algorithm, the Data Distribution algorithm counts mutually exclusive candidate items. Hence, as the number of nodes increases, more candidate itemsets may be counted in one pass. The major downside of this algorithm is the additional overhead of transmitting the local data along with the candidate itemsets.

The Candidate Distribution algorithm does not require data or candidate itemset communication between processors. This addresses some of the problems associated with count and data distribution methods. In the aforementioned two approaches, dependencies exist between the processors; if the load balancing and synchronization is not performed properly, the algorithm slows down considerably. Candidate Distribution eliminates this by removing the node communication. The basic idea is to divide the candidate itemsets and data among the nodes in such a way that each processor or node can perform the counting of candidates independently of the other processors. This division is performed after a certain number of passes, which is determined heuristically during the iterations.

During the tests performed by the authors, they found that the Count Distribution algorithm is the best way to parallelize the Apriori algorithm.

D. MapReduce Model

MapReduce [8] is a distributed software framework introduced by Google in 2004 to support distributed computing on large datasets on clusters of computers. MapReduce was derived from the mapreduce feature of functional programming languages but was adapted to distributed computing. It consists of two steps: Map and Reduce.

The Map phase consists of a Mapper class that receives an input (key, value) pair and produces a set of intermediate key/value pairs. The MapReduce library groups together all intermediate values associated with the same intermediate key and passes them to the Reduce function. It essentially emits all the key/value pairs containing the same key to the reducer.

The Reducer class contains the reduce function that accepts an intermediate key and a set of values for that key. It merges together these values to form a possibly smaller set of values and writes them to the specified output.

The basic MapReduce architecture is shown in Fig. 2.

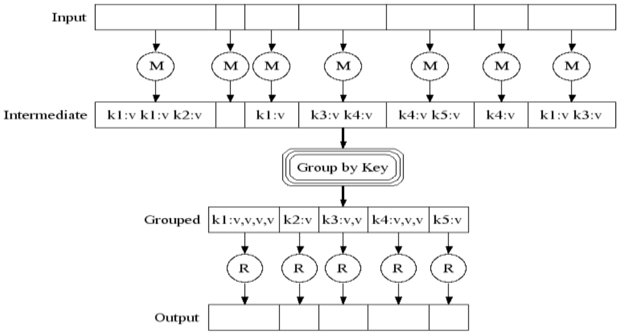


Figure 2. MapReduce architecture

To use the MapReduce programming model, each task must be represented as some computation on a list of key-value pairs.

E. Apache Hadoop

The following is a description of Hadoop from the official Apache website [20]: "Hadoop MapReduce is an open source software framework for writing applications which process vast amounts of data (multi-terabyte data-sets) in-parallel on large clusters (thousands of nodes) of commodity hardware in a reliable, fault-tolerant manner."

Hadoop MapReduce makes use of the Hadoop File System (HDFS) as the underlying distributed file system architecture. HDFS is responsible for distributing the data across multiple nodes and the number of replications, etc. There are many parameters which can be configured based on the requirement from the configuration files provided by the Hadoop distribution.

Minimally, applications specify the input/output locations and supply *map* and *reduce* functions via implementations of appropriate interfaces and/or abstract-classes. Further details may be found on the Apache Hadoop website [20].

Hadoop has three modes of operation: Standalone mode, Pseudo-Distributed mode, and Fully Distributed mode.

In Standalone mode, also known as local mode, there are no daemons running and everything runs within a single JVM (Java Virtual Machine). This is the default mode of operation and is suitable for running MapReduce programs during development, since it is easy to test and debug. In this mode, a single Hadoop node is created but does not use the Hadoop Distributed File System (HDFS). This means that all input and output files are read from and written to the underlying OS file system; thus, there appear to be no benefits in using HDFS.

In Pseudo-distributed mode, all of the Hadoop daemons run on the local machine, thus simulating a cluster on a single machine. Other than where the Hadoop processes are running (one machine vs. one machine per node), this mode is the same as distributed mode. In this mode, Hadoop starts all of the processes for all of the configured nodes on the same machine. This mode is useful because it allows the architect to observe how applications respond to running on a Hadoop cluster, but without the overhead of setting up the individual machines for the nodes of the cluster. While the task is much easier for a software architect using a cloud and Hadoop, there is still some overhead involved. Because the HDFS is used by default, benefits may be gained from using it.

In Fully distributed mode, the Hadoop daemons run on a cluster of machines. Each Hadoop node is started on the specified machine. As with pseudo-distributed, HDFS is used.

III. IMPLEMENTATION

The Count Distribution strategy was chosen for implementing the Apriori algorithm on a Hadoop cluster. Hadoop is not suited to implement the Data Distribution strategy since the data distribution cannot be controlled on Hadoop. It does not make sense in the MapReduce world to transmit data to other nodes. The Candidate Distribution strategy does not work either because a particular data block cannot be assigned to a particular node in Hadoop.

A very high-level view of the modified algorithm is provided below in Fig. 3.

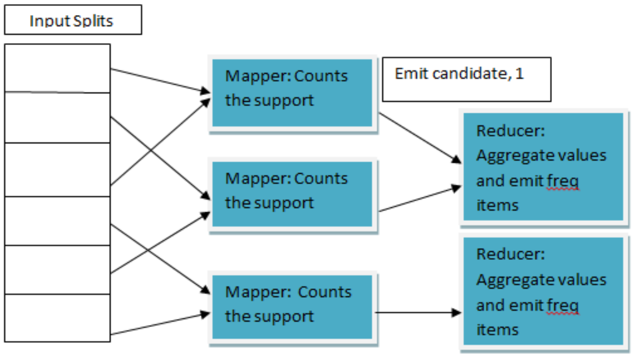

Input Splits
Mapper: Counts the support
Emit candidate, 1
Mapper: Counts the support
Mapper: Counts the support
Reducer: Aggregate values and emit freq items
Reducer: Aggregate values and emit freq items

Figure 3. Design of the algorithm

The algorithm works as follows. By using the HDFS file system and configuring it, Hadoop can automatically split the data files into smaller chunks and distribute them over the Hadoop cluster. The replication factor can be specified to replicate the data on multiple nodes so that the algorithm works even if a node does not work during the execution. The output files containing the frequent itemsets are stored in the output folder of the HDFS. These files need to be retrieved to the local system to generate subsequent candidates.

For performing the counting in parallel, each map job is configured to have the candidate itemset list and each node in the cluster has certain data blocks. The jobs are run multiple times for each frequent itemset having the minimum support. Each Map task that runs on a node reads the chunks present locally and emits the candidate, a count of 1 for each time the candidate is found in the file. The reduce function then aggregates all these counts for a particular key and writes the candidate to the output file only if it satisfies the minimum count. The output file is then read and the candidate list for the next iteration is generated. Thus, the counting step is parallelized.

IV. EVALUATION

To evaluate how well the design of the algorithm scales on multiple nodes and whether Hadoop as a framework is a good fit to perform association rule mining, some analyses were performed. The following section details how the data were generated and various analyses performed.

ISBN: 978-1-61208-272-1

A. Datasets used and Data Generation

We generated many datasets with varying parameters, but for the analysis we used four of them. We used a synthetic data generator to generate the transactions to perform frequent pattern mining. But the generated data would not be a perfect way to test the system as it was arbitrarily generated and may not really reflect real world data. Hence, we also used a dataset from the FIMI repository [19]. The dataset and a brief description of the dataset are given below.

1) Dataset: Accidents.dat

This dataset of traffic accidents was originally obtained from the National Institute of Statistics (NIS) for the region of Flanders (Belgium) for the period from 1991 to 2000. In total, 340,184 traffic accident records are included in the dataset with 572 different attribute values. On average, 45 attributes are filled out for each accident. More details about the attributes can be found in the FIMI repository.

2) Synthetically generated Datasets

The authors of the Apriori algorithm at IBM have developed their own tool to generate datasets to test their algorithm. They used different parameters to generate the dataset. The major factors are:

- Average size of the transaction (Ts)
- Number of transactions (Tn)
- Average size of maximal potentially frequent itemsets (P)
- Number of maximally potentially frequent itemsets (L)
- Number of items (N)

Many other studies used the aforementioned IBM Data generator to generate synthetic data generator. However the tool is now obsolete. Hence, ARtool[21], another open source tool package, was chosen for generating the synthetic data. The tool is an open source tool developed at the University of Massachusetts at Boston and it provides options to mimic the parameters and datasets. By using this tool, four datasets were generated. They are summarized in Table I:

TABLE I. DATASETS

Dataset	Tn	Ts	L	P	N
200K_f50	200000	50	2000	6	1000
200K_f30	200000	30	2000	6	1000
200K_f20	200000	20	2000	6	1000
100K_50	100000	50	2000	6	1000

B. Cluster Setup

Hadoop, as explained earlier, runs in three different modes. All three modes of operation were used while developing and testing the algorithm.

Standalone mode: When the Hadoop and MapReduce framework were tested, many test programs had to be written to test Mappers and Reducers performing tasks in parallel. For the purpose of testing the basics of Hadoop, the NetBeans IDE with a plug-in called Karmasphere was used to test Hadoop applications that had been written. Karmasphere simulates Hadoop and provides Mappers and Reducers within a single node. It uses only the local file system and no distribution of data occurs. The first basic version of Apriori was developed according to MapReduce framework in this mode and was tested on small datasets to determine if the core algorithm was generating correct results.

Pseudo-Distributed Mode: This mode uses the HDFS system and distributes data albeit in a single node. It simulates the Hadoop environment on a single node. Essentially it is a cluster with just one node in it. The Mapper and Reducer run as separate daemons in the system. This mode was used to test whether the program was reading and writing files correctly from and to the HDFS and local file system and vice versa.

Fully Distributed Mode: This is the mode where we have a full Hadoop cluster with multiple nodes running the program. Initially a Hadoop cluster was built with two nodes, i.e., a system and a VM within the system to test the application. We used this to test if the cluster was setup correctly and to understand the complexity of setting up a Hadoop cluster. The IBM smart cloud was then chosen to test the algorithm and the framework. IBM smart cloud is a PAAS (Platform as a Service) that provides Hadoop master and data nodes. The programmer does not need to create the cluster nor perform installation and maintenance of the cluster. This is automatically performed by the cloud provider. With access to 5 nodes in the cloud, multiple tests were performed to generate some interesting and promising results. The following sections describe those results.

C. Analysis on Number of Nodes vs. Running Time

In this analysis, the goal was to test if there would be an improvement in the running time as the number of nodes increased. The accident.dat from the FIMI repository was used as a dataset and the program was run on 2, 3, 4, and 5 nodes. The results are shown below in Table II and Fig. 4.

TABLE II. NUMBER OF NODES VS. RUNNING TIME

Dataset	Support	Nodes	Time(min)
Accidents.dat	0.8%	2	197
Accidents.dat	0.8%	3	152
Accidents.dat	0.8%	4	107
Accidents.dat	0.8%	5	92

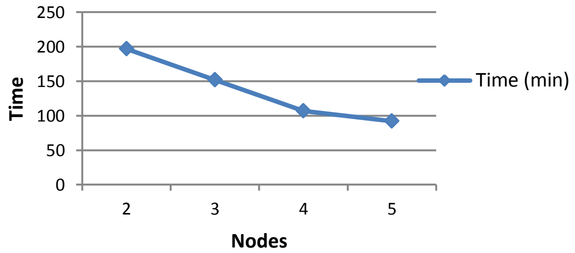


Figure 4. Number of nodes vs. running time

Copyright (c) IARIA, 2013. ISBN: 978-1-61208-272-1

As is evidenced, as the number of nodes increases, the running time of the algorithm decreases drastically. But the decrease in time is not always at the same rate. This depends upon the size of the data, and more importantly, the split factor of the data. When the files are transferred from the local file system to HDFS, a block size needs to be specified for the file to be split into. Each block is then assigned to a node and all the operations on that block are performed by only that node. If the number of nodes exceeds the number of blocks, it is a waste of resources. So, at some point, increasing the number of nodes would not lead to the decrease in time. This point could not be reached in this analysis as there were a limited number of nodes in the cluster.

The important conclusion from this analysis is that running the Apriori algorithm parallel on individual data blocks on Hadoop does lead to a significant performance increase and Hadoop as a distributed framework is a good platform to use to perform data intensive computations. Another observation that should be noted is the advantage of using a cloud platform to achieve this. Both the individual cluster and the IBM smart cloud were utilized and clearly, in terms of adding new nodes to the cluster and scalability, using a cloud is much more advantageous than building one's own cluster. Substantial maintenance costs could be associated with maintaining one's own Hadoop cluster as typically such mining tasks are not run every day but rather once in two weeks or once a month. There are many cloud providers offering a Hadoop cluster and using these cloud vendors is much less expensive and also much more efficient than running it on one's own cluster.

D. Analysis on Running Time Comparison Based on Dataset

In this analysis, the goal was to test the performance of the algorithm on different datasets. The percentage decrease in time was used as a measure for testing the performance. The two datasets were synthetically generated, i.e., 100k_50 and 200K_50, and run on the Hadoop cluster with varying number of nodes. The algorithm was run on both datasets on 2, 3, 4 and 5 nodes in order to compare the decrease in time needed to run the algorithm. The results obtained are shown in Table III and Fig. 5.

Fig. 5 provides some really interesting insights into Hadoop and the algorithm in general. The running time on two nodes was used as a reference to compare the decrease in time when running the algorithm on the two datasets on different number of nodes. As shown in Fig. 5, as the size of the dataset increases, the time gained by running the algorithm on multiple nodes increases considerably. Consider the case of running the program on the 200K transactions dataset. The percentage decrease in time by running it on 5 nodes is close to 64% as compared to running it on 2 nodes. This is a significant increase in the efficiency of the algorithm.

TABLE III. RUNNING TIME ON DIFFERENT DATASETS

Nodes	Percentage drop in time	
	100K	200K
2	0	0
3	25.9	32.9
4	29.84	36.68
5	27.8	64.37

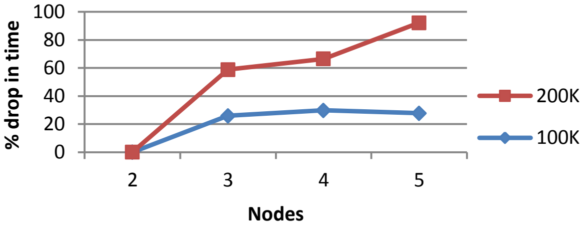


Figure 5. Comparison of running time on different datasets

Another interesting result observed is that the program scales well for larger datasets. As is evidenced, in almost every instance the time saved by running the larger dataset on multiple nodes is higher than the time saved by running the program on smaller dataset. This can be attributed to the fact that when running the algorithm in parallel on individual data blocks, the amount of time needed to perform the counting decreases. In Hadoop, we use HDFS as the distributed file system. HDFS splits the entire file into a number of blocks. So, if the number of nodes working on individual data blocks simultaneously is increased, the time is reduced. Thus, it is very important to choose the cluster size and file split size depending upon the data.

Another important observation relates to the 100K transaction. When running the algorithm on 4 nodes and then on 5 nodes, instead of time decrease which was expected, there was actually an increase in time. The job specifics were studied in order to understand this result and it was found that even though there were 5 nodes in the cluster, as a result of the division of the file, only 4 nodes were participating in the Map and Reduce phases. This unnecessarily caused some overhead in communication. So, it is very important to fix the file split size appropriately depending on the cluster. This is the same reason why there was a spike in the decrease in percentage of time as we added one more node to 4 nodes in the case of 200K transactions. The JobTracker was again studied to see what caused such a spike and noticed that when there were 5 nodes in the cluster, each Map task worked on an individual data block. A Map task is nothing more than a running instance of the Map class on the client node. Whereas in the case of 4 nodes, there were some pending map tasks which had to be computed again. Hence, it caused the steep increase in performance of the program. This reiterates how important the factors controlling the Hadoop environment are. There are many other factors to affect how Hadoop may be configured, such as setting the number of map tasks, reduce tasks, defining our own split function, etc.

E. Analysis on Running Time vs. Transaction Length

In this analysis, the goal was to test how the running time varies based on the length of the transaction. The tests confirmed what was expected. A 200K transaction database generated synthetically was chosen. The average lengths of the transactions were chosen to be 20, 30, and 50. The results are shown in Table IV and Fig. 6.

TABLE IV. RUNNING TIME VS. TRANSACTION LENGTH

Data	Time(min)
200k_50	28
200k_30	16
200k_20	4.5

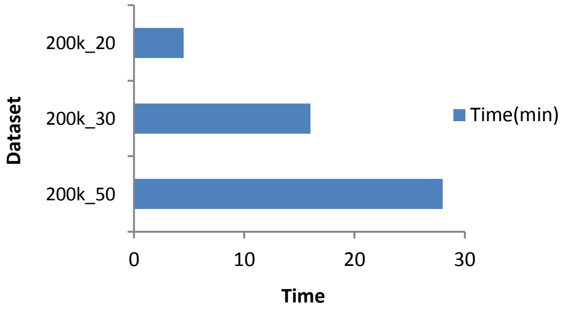


Figure 6. Running time vs. transaction length

The results for this analysis are straight-forward. When the transaction size is small, there is fast processing to count the candidate itemsets; hence, less time.

V. CONCLUSION

In this paper, the design, implementation, and evaluation of using Hadoop as a distributed framework for association rule mining was presented. This proves that a parallelized version of the Apriori could be very efficient and easy to port to Hadoop. The IBM smart cloud was also used and it shows the huge potential that cloud computing has to offer to organizations performing data mining tasks. The cloud provides an easy-to-use interface and web consoles to launch the cluster with pre-installed software and network connectivity. This is particularly useful as the programmer can instead concentrate on writing efficient algorithms and optimizing the data analysis rather than worrying about the maintenance of the Hadoop cluster.

For this paper, the main goal was not to optimize the Apriori algorithm but to test whether it can be implemented on Hadoop with satisfactory results. A modified version of the algorithm was designed and provided very good results with respect to the platform; yet it is currently slow. The speed of the algorithm may be improved to a large extent by using Tries and other alternative implementations for counting the candidates.

REFERENCE

[1] R. Agrawal and J. Shafer. “Parallel Mining of Association Rules,” IEEE Trans. Knowledge and Data Eng., Vol. 8, No. 6, 1996, pp. 962–969.

[2] R. Agrawal and R. Srikant, “Fast algorithms for Mining Association Rules,” Proc. Int’l Conf. Very Large Database, 1994, pp. 487-499.

[3] S. Brin, R. Motwani, J. D. Ullman, S. Tsur, “Dynamic Itemset Counting and Implication Rules for Market Basket Data,” Proc. ACM Conf. Management of Data, 1997, pp. 255-264.

[4] D. Cheung et al., “A Fast Distributed Algorithm for Mining Association Rules,” Proc. Int’l Conf. Parallel and Distributed Information Systems, 1996, pp. 31–42.

[5] D. Cheung and Y. Xiao, “Effect of Data Skewness in Parallel Mining of Association Rules,” Proc. Pacific-Asia Conf. Knowledge Discovery and Data Mining, Lecture Notes in Computer Science, Vol. 1394, Springer-Verlag, 1998, pp. 48–60.

[6] F. Coenen and P. Leng, “Partitioning Strategies for Distributed Association Rule Mining,” The Knowledge Engineering Review, Vol. 21, No. 1, 2006, pp. 25 - 47.

[7] J. Cryans, S. Ratté, R. Champagne, “Adaptation of Apriori to MapReduce to Build a Warehouse of Relations between Named Entities across the Web,” Proc. Int’l Conf. Advances in Databases, Knowledge, and Data Applications, 2010, pp. 185-189.

[8] J. Dean and S. Ghemawat, “Mapreduce: Simplified data processing on large clusters,” Communications of the ACM, Vol. 51, No. 1, 2008, pp. 107-113.

[9] E.H. Han, G. Karypis, V. Kumar, “Scalable Parallel Data Mining for Association Rules,” ACM Conf. Management of Data, 1997, pp. 277–288.

[10] J. Han, J. Pei, Y. Yin, “Mining Frequent Patterns without Candidate Generation,” Proc. Conf. Management of Data, 2000, pp. 1-12.

[11] L. Li and M. Zhang, “The Strategy of Mining Association Rule Based on Cloud Computing,” Proc. Int’l Conf. Business Computing and Global Informatization, 2011, pp. 475-478.

[12] J.S. Park, M. Chen, P. S. Yu, “An effective Hash-Based Algorithm for Mining Association Rules,” Proc. ACM Conf. Management of Data, 1995, pp. 175-186.

[13] J.S. Park, M. Chen, P.S. Yu, “Efficient Parallel Data Mining for Association Rules,” Proc. Int’l Conf. Information and Knowledge Management, 1995, pp. 31–36.

[14] A. Savasere, E. Omiecinski, S. Navathe, “An Efficient Algorithm for Mining Association Rules in Large Databases,” Proc. Int’l Conf. Very Large Database, 1995, pp. 432-444.

[15] T. Shintani and M. Kitsuregawa, “Hash Based Parallel Algorithms for Mining Association Rules,” Proc. Int’l Conf. Parallel and Distributed Information Systems, 1996, pp. 19–30.

[16] X. Yang, Z. Liu, Y. Fu, “MapReduce as a programming model for association rules algorithm on Hadoop,” Proc. Int’l Conf. Information Sciences and Interaction Sciences, 2010, pp. 99-102.

[17] M.J. Zaki, S. Parthasarathy, M. Ogihara , W. Li, “Parallel Algorithms for Fast Discovery of Association Rules,” Data Mining and Knowledge Discovery: An Int’l J., Vol. 1, No. 4, 1997, pp. 343-373.

[18] M. J. Zaki. Parallel and Distributed Association Rule Mining: A Survey. IEEE Concurrency, Vol. 7, No. 4, 1999, pp. 14-25.

[19] http://fimi.ua.ac.be

[20] http://hadoop.apache.org

[21] http://www.cs.umb.edu/~laur/ARtool/index.html

ISBN: 978-1-61208-272-1

An Alternative Archiving Technique for Evolutionary Polygonal Approximation

José Luis Guerrero, Antonio Berlanga and José Manuel Molina
Computer Science Department, Group of Applied Artificial Intelligence
University Carlos III of Madrid
Colmenarejo. Spain
jguerrer@inf.uc3m.es, aberlan@ia.uc3m.es, molina@ia.uc3m.es

***Abstract*—Archiving procedures are a key parameter for Multi-objective evolutionary algorithms, since they guarantee the algorithm convergence and the good spread of the obtained solutions in the final Pareto front. For many practical applications, the cost of the algorithm is clearly dominated by the computational cost of the underlying fitness functions, allowing complex processes to be incorporated into the archiving procedure. This work presents a study of the archiving technique for evolutionary polygonal approximation (the division of a given curve into a set of n segments represented by a linear model) based on the epsilon-glitch concept, highlighting the cost of the technique compared to the fitness computation, and proposing a novel alternative archiving procedure, which yields statistically significant better results compared to available approaches.**

Keywords-Archiving; Polygonal approximation; Segmentation; Multi-objective evolutionary algorithms.

I. Introduction

Multi-objective evolutionary algorithms (MOEA) have become a proficient tool dealing with multi-objective problems [1]. This can be shown by the growing number of real world applications related to their use in the last years [2], with examples such as marine vehicle design [3], groundwater monitoring applications [4], or architectural design [5].

The phases of a MOEA can be stated as follows [1]: initialization, dominated individuals removal, use of density estimators, evolutionary operations, next generation selection, check termination criterion, and archive updating. Archive handling is a phase which is also very common for a wide range of metaheuristics [6]. Archiving can be defined as the process of storing individuals from a population to preserve them from the usual cycles of the evolutionary process applied to the consecutive generations. Some of the initial MOEAs including explicit archiving techniques were SPEA [7], PAES [8] and MOMGA [9].

The initial concern regarding archiving procedures was to guarantee a good distribution of the Pareto optimal solutions, since convergence was considered guaranteed [10]. However, as established in [11], under standard Pareto-based selection schemes, deterioration may occur, and, thus, convergence is no longer guaranteed. Examples of algorithms suffering this effect are PAES [8] or SPEA [7].

One of the most successful MOEAs introduced up to date, regarding its impact in the research community and extended use, is probably SPEA2 [12]. This algorithm introduced a novel archiving technique based on an environmental selection process, according to the concept of strength: for every individual, how many individuals dominate and are dominated by it. This process provided the algorithm with statistically significant improvements versus relevant coetaneous algorithms, such as NSGA-II [13]. It must be noted, though, that these comparisons measured the computational cost for the stopping criterion in terms of function evaluations, disregarding the cost of the algorithms procedures (including the archiving technique).

Polygonal approximation techniques [14] are segmentation processes (the division of a given series of data into *n* segments, usually represented by a linear model) applied to closed curves as an offline process. The multi-objective nature of these processes has been explicitly faced in recent works [15], [16], as opposed to traditional heuristic, single objective approaches [17].

In [16], a MOEA for polygonal approximation is presented based on the SPEA2 algorithm. Its results proved to be statistically significant compared to a set of traditional heuristic techniques in terms of quality. This work will present an alternative archiving strategy comparing the cost of the environmental selection process versus a specific archiving technique based on the discrete nature of one of the objectives, inspired by the ε-dominance technique introduced in [11]. ε-dominance describes the ε value required for solution to dominate another one (and can be presented in additive or multiplicative terms). The presented proposal can be considered on terms of a general novel archiving procedure and also as a specific proposal for the multi-objective polygonal segmentation domain, which benefits from its improvements.

The remaining sections are organized according to the following structure: Section II focuses on the general presentation of the novel archiving procedure, analyzing some of the literature regarding archiving techniques and highlighting the basis for the presented proposal. Section III will present the MOEA approach for polygonal approximation, which is the specific problem than inspires the presented technique and also the one it is applied to in order to verify its results, followed by Section IV, where the proposed archiving technique is presented and detailed. Finally, Section V presents the experimental results obtained, and Section VI the obtained conclusions from these results and possible future lines for the developed research.

II. ARCHIVING TECHNIQUES AND THE ENVIRONMENTAL SELECTION PROCESS

In [11], Laumanns et al. proved that many MOEAs based on standard Pareto-based selection schemes could suffer deterioration, not guaranteeing convergence. Deterioration occurred when elements of a solution set at a given time were dominated by a solution set which the algorithm maintained some time before. Based on these observations, they presented new archiving strategies based on the ε-dominance concept, attempting to provide both convergence and good distribution properties.

However, in [18], some of the issues with this approach were highlighted, mainly the choice of the ε initial parameter. This parameter could be chosen by either a preset value or by an adaptative procedure. In the former case, the number of points in the archive is bounded by a function of the (possibly unknown) objective space ranges. In the latter case, ε may become arbitrarily large, providing a poor final archived set compared to the sequence of points presented to the archiving algorithm.

SPEA algorithm family, both SPEA [7] and SPEA2 [12], relies on the concept of strength for their archiving strategy: originally proportional to the number of solutions which an individual dominated (in the SPEA algorithm), it was improved in the SPEA2 algorithm by also including the number of solutions which dominate it. This led to the environmental selection update mechanism for the archive.

The original archive update mechanism was based on a clustering technique. This mechanism tended to lose boundary solutions when the archive size was too small for the required number of non-dominated solutions. The truncation technique presented in SPEA2 is an iterative process that eliminates at each stage the individual with the minimum distance to another individual (considering the following distances to the second, third... closest individuals in case of ties). This process continues until the maximum number of individuals, according to the archive size parameter, has been introduced.

The archive size in SPEA2 is fixed. If the number of non-dominated individuals is not sufficient to fill it, dominated ones are inserted. Also, the environmental selection mechanism dominates the complexity of the whole algorithm, with a worst case complexity of $O(M^3)$, where M is the population size plus the archive size. On average, that complexity is reduced to $O(M^2 log M)$.

III. MULTI-OBJECTIVE EVOLUTIONARY POLYGONAL APPROXIMATION

Polygonal approximation is the process of dividing a given closed curve into a set of *n* segments, represented each of them by a linear model. If we formalize a general closed curve according to (1), the segmentation process can be formalized with (2).

$$t = \{\vec{p}_i\}, \vec{p}_i = (x_i, y_i, i), i = 1,...,n \tag{1}$$

$$S(t) = \{B_m\}, B_m = \{\vec{p}_i\}, i = k_{\min},...,k_{\max} \quad m \in [1,...,q], \mathrm{q} < \mathrm{n} \tag{2}$$

In (2), *t* represents the closed curve, $\vec{p}_i$ each of its points, x_i and y_i its two dimensional components, *i* an ordering value, *S(t)* the result of the segmentation process, B_m a segment resultant of that process and, for each of these resultant segments, $k_{\min}$ and $k_{\max}$ represent the points at its edges.

Two different problems have been traditionally associated with this process: Min-# and Min-ε problems. Min-# is based on the optimization of the representation error for a previously set value of *q*. Min-ε, on the other hand, finds the minimum number of segments such that the final representation error does not exceed a previously established error ε. Both problems rely on the two objectives of a possible multi-objective formalization for the problem, presented in (3).

$$\min\{E(S(t),t),q\} \quad s.that\left\{\begin{array}{c} E(S(t),t \le total\_error \\ \forall m, E(S(B_m),B_m) \le \max\_seg\_error \end{array}\right\} \tag{3}$$

This formalization presents the two functions to be minimized jointly: the representation error *E(S(t), t)* and the cost of that representation, its number of segments *q*, which must be lower than its original number of points *n*, as established in (2). Traditionally, restrictions have been considered for this issue [17], being these restrictions represented by the overall representation error (*total_error)* and the maximum representation error for each individual segment (*max_seg_error)*

In [16], this problem is faced using a SPEA2 MOEA algorithm. This proposal allows dealing with both minimization problems jointly. Also, as explained in [16], there is a great degree of possible information sharing between different solutions with different numbers of segments, which allows obtaining a Pareto front of possible solutions in an efficient way. Finally, Min-ε and Min-# problems require user configuration parameters, which may be hard to establish a-priori.

The archiving procedure of this algorithm has been thoroughly described in Section II. Also, as commented in the introduction, the performance comparison between different MOEAs does usually only take into account the number of function evaluations (or, similarly, the number of generations), under the consideration that the computational complexity of these function evaluations exceeds that of the algorithm itself. Recent procedures [19] have considerably reduced the computational complexity of fitness evaluations in evolutionary polygonal approximation.

Figure 1 shows a comparison of the running time of the different procedures of the algorithm using this enhanced fitness evaluation. These results have been obtained using the JMetal [20] environment with the general configuration established in [16] using a population size of 200 and the

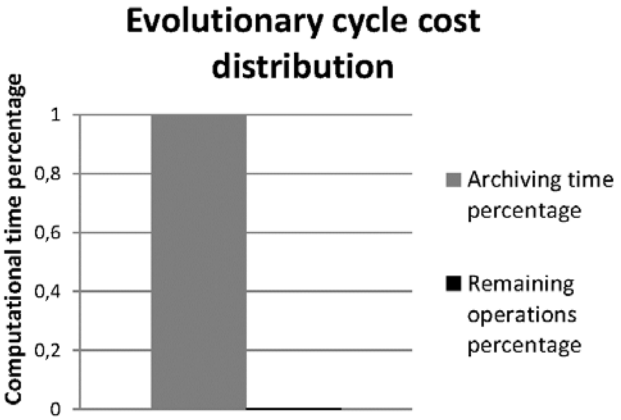


Figure 1. Computational cost distribution in the SPEA2 algorithm for a polygonal approximation problem instance

enhanced fitness computation presented in [19]. The specific problem instance used is the chromosome figure, even though similar results are obtained for the different instances considered in the dataset. These instances will be presented in Section V, along with the experimental results.

As Figure 1 clearly shows, the archiving procedure is not only dominating the algorithm running time, but also the enhanced fitness computation (archiving implies more the 99% of the whole running time, including fitness computation). This huge effort to guarantee a well distributed Pareto front seems unacceptable. The following section will present an alternative archiving technique based on the characteristics of the problem.

IV. ALTERNATIVE ARCHIVING PROCEDURE FOR EVOLUTIONARY POLYGONAL APPROXIMATION

The polygonal approximation process has a set of very specific characteristics, mainly its bi-objective nature with a very high degree of conflict between them and the fact that one of these objectives is discrete. This nature of the problem has been used, for instance, to provide computationally efficient initializing methods [21]. Figure 2 shows the result of an initialization process prior to the application of Pareto dominance selection to highlight these characteristics.

Some of the issues related to the costly archiving results exhibited by SPEA2 in this problem, as shown in Figure 1, are related to the multi-objective proposal: the algorithm must be able to store, ideally, one individual per each compression level. This implies that the archive size can get to be really large (and the computational complexity of the environmental selection depends heavily on that archive size Initialization example for a MOEA polygonal approximation problem instance.

A vector $\vec{u} = (u_1,...,u_k)$ is said to ε-dominate another vector $\vec{v} = (v_1,...,v_k)$ for some ε>0 iff $\forall i \in 1,...,k, (1-\varepsilon)u_i \le v_i$. The idea presented in [11] was, according to ε-dominance, to draw ε -boxes such that at most one element is contained in each box. From the characteristics of the problem presented, the idea for the alternative archiving presented is to use a technique similar to these ε -boxes, considering a box for each of the possible individuals according to the number of segments objective. Figure 3 represents these boxes over the results in Figure 2.

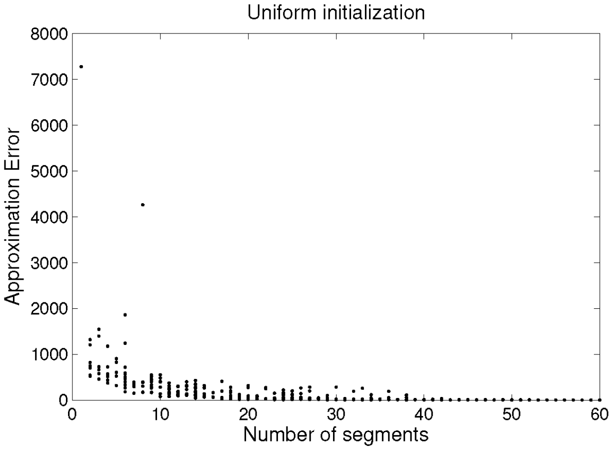


Figure 2. Initialization example for a MOEA polygonal approximation problem instance

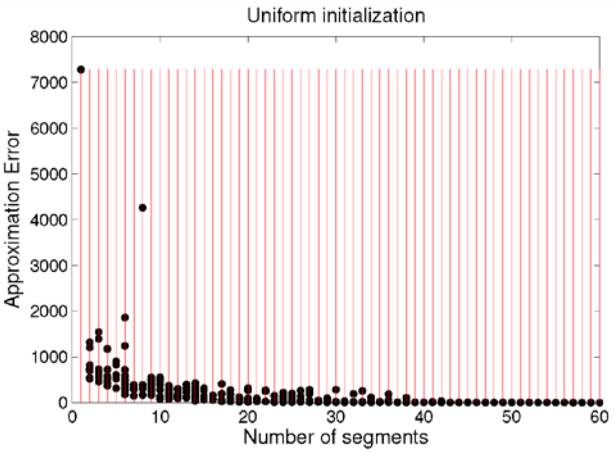


Figure 3. Initialization example for a MOEA polygonal approximation problem instance

As shown in Figure 3, these boxes are infinitely thin in one objective (the discrete objective representing the number of segments, where they only cover one value) and infinitely long in the other (the objective representing the representation error). These particular instances of ε-boxes are similar to the glitches from signal processing theory, and so they have been named according to this resemblance. Pareto-dominance is only checked within the ε-glitch which an individual belongs to, not among different glitches. This implies that the complexity of this process is now constant, and the complexity of the whole archiving mechanism is reduced to *O(n)*, where *n* is the population size. Figure 4

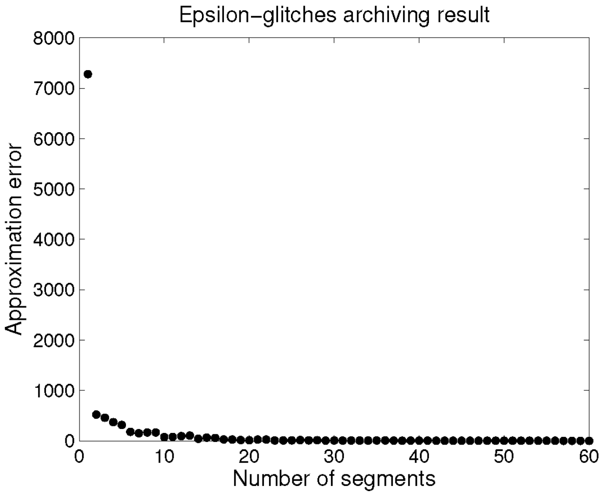


Figure 4. ε-glitches archiving procedure example

shows the result of the ε-glitches archiving procedure over the population presented in Figure 2.

It must be noted that one of the issues related to the environmental selection process, the fact that its complexity included the archive size, has been overcome, an especially important achievement for this problem, since the required number of individuals in the archive can be very large for some problem instances, as explained in previous sections.

The traditional evolutionary cycle where one full generation is produced at each step is no longer required, since every individual is compared to the correspondent one in its glitch already stored in the archive. With this approach, the evolutionary cycle implies parent selection, children obtaining through the transformation operators and the invocation of the archiving procedure individually

Even though implementation issues are not the focus of this work, the appropriate management of the data structure for the archive is important for the computational cost reduction. Since the archive size is fixed (with an exception detailed in the following paragraph), a fixed size, constant time access data structure is suggested (such as a traditional array). For the initialization of this array, we suggest the generation of an initial random individual for one of the archive boundaries (either of them) and the application of directed mutations to obtain an individual for each of the ε-glitches. This directed mutations (and, thus, the archive initialization process) imply a very low computational cost using the fitness computations detailed in [19].

One final improvement is introduced in the archiving procedure. For MOEA approaches for polygonal segmentation, the initial archive size, as explained in [16], is set to the number of points in the curve. However, very commonly, through the evolutionary process, new individuals with a perfect segmentation (zero error) are found, requiring a lower number of segments than that initial boundary. With the explained archiving mechanism, this would not be taken into account, since it covers Pareto dominance between two individuals which belong to two different ε-glitches. To cover this special case, when a new individual is added to the archive with zero representation error, the archive size is reduced to its number of segments.

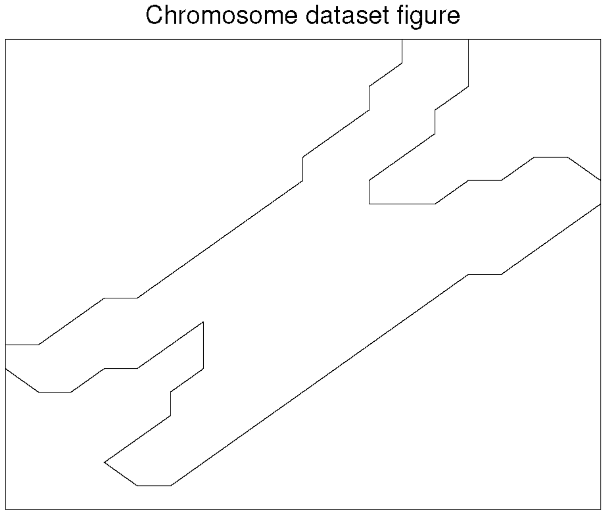


Figure 5. Chromosome dataset figure

A final overview of the detailed archiving process can be detailed in the following steps:

- Initialize archive with size *n* and fill with individuals obtained with applied directed mutations from an individual with either, the highest or the lowest possible number of segments, obtaining one individual for each possible number of segments
- While the stopping criterion is not met
    - Select parents
    - Apply transformation operators to parents
    - For each children produced:
        - Compare to archive individual with the same number of segments, and update as required
        - If the updated individual has zero representation error, update archive size if required (the number of segments of the individual is smaller than archive size)
- Output archive results

V. EXPERIMENTAL EVALUATION

The dataset used for the experimental evaluation of the proposed technique is based on three traditional figures for the polygonal approximation domain, usually named chromosome, leaf and semicircle [14]. As discussed in [19], these figures, while representative and appropriate for technique comparison, are not very complex. Thus, the procedure presented in that paper will be used to generate three additional figures for the dataset, containing ten linked copies of the previous three, with the names chromosome10, leaf10 and semicircle10. Figure 5 presents the chromosome curve, while Figure 6 shows the chromosome10 curve.

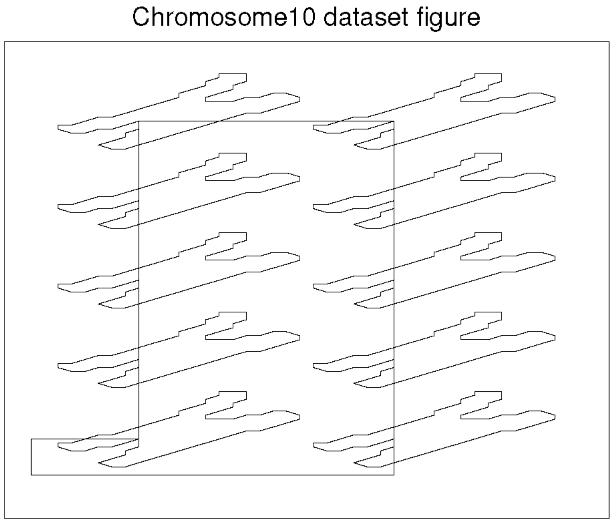


Figure 6. Chromosome10 dataset figure, built using ten different instances of the chromosome figure

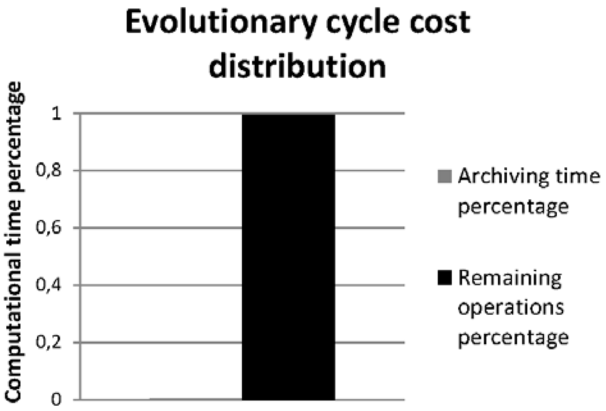


Figure 7. Computational cost distribution in the proposed algorithm with the ε-glitches archiving technique

TABLE I. FINAL POPULATIONS COMPARISON

Figure	Epsilon Hyp. results		SPEA2 Hyp. results		Best
	Mean	*Std*	*Mean*	*Std*	
chrom	0,99000	0,00006	0,99001	0,00005	-
leaf	0,99533	0,00001	0,99532	0,00001	-
semi	0,99409	0,00008	0,99412	0,00003	-
chrom10	0,99924	0,00002	0,99858	0,00093	eps.
leaf10	0,99961	0,00001	0,99812	0,00139	eps.
semi10	0,99944	0,00002	0,99814	0,00175	eps.

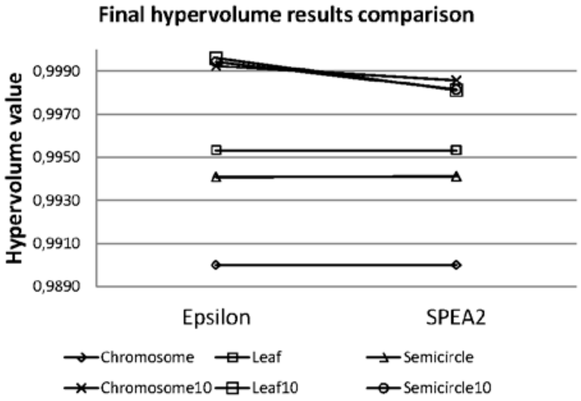


Figure 8. Computational cost distribution in the proposed algorithm with the ε-glitches archiving technique

The first relevant comparison, according the considerations presented in Section II, is the computational time which is now used by the archiving procedure. The original results for the SPEA2 technique have been presented in Figure 1. The results with the presented technique are shown in Figure 7. These results are general for all the different figures in the dataset. As this figure shows, the percentages have been swapped, spending now more than 99% percent of the available computational time in the evolutionary search instead of the archiving technique.

The experimental configuration is based on thirty different executions of the SPEA2 algorithm as described in [16]. The population size used is 200, and the algorithm is left to run for 200 generations. The running time used for each of this independent executions is measured, and afterwards thirty different runs of the proposed technique are performed, each of them according to an individual previously measured running time as its stopping criterion. The hypervolume indicator values [22] are built afterwards according to these results and the statistical significance of the differences are tested according to Wilcoxon statistical testing [23] with a 95% confidence interval. The results of these procedures are shown in Table I. Figure 8 summarizes these results.

The results for the three initial figures do not show statistically significant differences between the two techniques, while the results for the three harder ones are clearly dominated by the epsilon-glitches based technique. The explanation for this fact is clear: when the final solution can be easily reached, the improved distribution of the solutions provided by the environmental selection technique allows SPEA2 to obtain solutions of similar quality, even though the computational effort spent in the proper search process is smaller (as seen in the comparison of Figures 1 and 7). As the problem instances become harder and the extent of search required to reach a reasonable Pareto front grows, the focus on the search process of the epsilon-glitches technique pays off for the poorer solution distribution, providing it with substantial better results in terms of final hypervolume.

VI. CONCLUSIONS AND FUTURE WORK

This work has presented a study over the archiving procedure considerations for evolutionary polygonal approximation. Available approaches based on standard, well established multi-objective evolutionary approaches, spend most of their computational time on their archiving procedures (when combined with novel improved fitness computation algorithms). This leads to a waste of computational resources, particularly for hard problem instances. A novel archiving procedure, based on a reduced Pareto-dominance application, has been presented, and its results tested over a set of six different functions with growing difficulty.

The epsilon-glitch technique is applied to the multi-objective polygonal segmentation issue, applying a simplified version of previous epsilon-dominance proposals to this bi-objective problem, focusing on the archive organization and direct comparisons to provide computational cost enhancements. The results show that the techniques, for easier problems, do not yield statistically different results. However, as the difficulty of the problem instances is increased, the novel epsilon-glitch archiving procedure provides clearly better results, since it spends most of its computational effort in the search of the best solutions, instead of the archiving procedure. Future lines include the study of mixed approaches, where full Pareto-dominance can be introduced at certain steps of the presented approach, and the extension of this approach to different sets of problems with similar characteristics.

ACKNOWLEDGMENT

This work was supported in part by Projects MINECO TEC2012-37832-C02-01, CICYT TEC2011-28626-C02-02, CAM CONTEXTS (S2009/TIC-1485)

REFERENCES

[1] C. Coello, G. Lamont, and D. Van Veldhuizen, Evolutionary algorithms for solving multi-objective problems. Springer, 2007

[2] C. Coello and G. Lamont, Applications of multi-objective evolutionary algorithms. World Scientific Publishing Company Incorporated, 2004.

[3] D. Lee, “Multiobjective design of a marine vehicle with aid of design knowledge,” International journal for numerical methods in engineering, vol. 40, no. 14, 1997, pp. 2665–2677.

[4] P. Reed, B. Minsker, and D. Goldberg, “A multiobjective approach to cost effective long-term groundwater monitoring using an elitist nondominated sorted genetic algorithm with historical data,” Journal of Hydroinformatics, vol. 3, 2001, pp. 71–89.

[5] J. Byrne, M. Fenton, E. Hemberg, J. McDermott, M. ONeill, E. Shotton, and C. Nally, “Combining structural analysis and multi-objective criteria for evolutionary architectural design,” Applications of Evolutionary Computation, 2011, pp. 204–213.

[6] E. Talbi, Metaheuristics: from design to implementation. Jonh Wiley and Sons Inc., 2009.

[7] E. Zitzler and L. Thiele, “Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach”, IEEE Transactions on Evolutionary Computation, vol. 3, no. 4, 1999, pp. 257–271.

[8] J. Knowles and D. Corne, “Approximating the nondominated front using the pareto archived evolution strategy,” Evolutionary computation, vol. 8, no. 2, 2000, pp. 149–172.

[9] D. Van Veldhuizen and G. Lamont, “Multiobjective optimization with messy genetic algorithms,” in Proceedings of the 2000 ACM symposium on Applied computing-Volume 1, 2000, pp. 470–476.

[10] G. Rudolph and A. Agapie, “Convergence properties of some multi-objective evolutionary algorithms,” Proceedings of the 2000 Congress on Evolutionary Computation, 2000., vol. 2. IEEE, 2000, pp. 1010–1016.

[11] M. Laumanns, L. Thiele, K. Deb, and E. Zitzler, “Combining convergence and diversity in evolutionary multiobjective optimization,” Evolutionary computation, vol. 10, no. 3, 2002, pp. 263–282.

[12] E. Zitzler, M. Laumanns, and L. Thiele, “SPEA2: Improving the Strength Pareto Evolutionary Algorithm,” Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems, K. Giannakoglou, D. Tsahalis, J. Periaux, P. Papailou, and T. Fogarty, Eds. Athens, Greece: International Center for Numerical Methods in Engineering (CIMNE), 2001, pp. 95–100.

[13] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, “A fast and elitist multiobjective genetic algorithm: Nsga-ii,” Evolutionary Computation, IEEE Transactions on, vol. 6, no. 2, 2002, pp. 182–197.

[14] M. Sarfraz, “Linear capture of digital curves,” in Interactive Curve Modeling. Springer London, 2008, pp. 241–265.

[15] A. Kolesnikov, P. Franti, and X. Wu, “Multiresolution polygonal approximation of digital curves,”. Proceedings of the 17th International Conference on Pattern Recognition, vol. 2. IEEE, 2004, pp. 855–858.

[16] J. Guerrero, A. Berlanga, and J. Molina, “A multi-objective approach for the segmentation issue,” Engineering Optimization, vol. 44, no. 3, 2012, pp. 267–287.

[17] E. Keogh, S. Chu, D. Hart, and M. Pazzani, “Segmenting time series: A survey and novel approach,” Data mining in time series databases, vol 57, 2004, pp. 1–21.

[18] J. Knowles and D. Corne, “Properties of an adaptive archiving algorithm for storing nondominated vectors,” IEEE Transactions on Evolutionary Computation, , vol. 7, no. 2, 2003, pp. 100–116.

[19] J. Guerrero, A. Berlanga, and J. Molina, “Fitness-aware operators for evolutionary polygonal approximation,” in Applied Computing. IADIS, 2012, pp. 283–290.

[20] J. J. Durillo and A. J. Nebro, “jmetal: A java framework for multi-objective optimization,” Advances in Engineering Software, vol. 42, no. 11, 2011, pp. 760–771.

[21] J. Guerrero, A. Berlanga, and J. Molina, “Initialization procedures for multiobjective evolutionary approaches to the segmentation issue,” Hybrid Artificial Intelligent Systems, 2012, pp. 452–463.

[22] E. Zitzler, L. Thiele, M. Laumanns, C. Fonseca, and V. Da Fonseca, “Performance assessment of multiobjective optimizers: An analysis and review,” Evolutionary Computation, IEEE Transactions on, vol. 7, no. 2, 2003, pp. 117–132.

[23] G. Corder and D. Foreman, Nonparametric statistics for nonstatisticians: a step-by-step approach. West Sussex, England: John Wiley & Sons Inc, 2009.

Future Irregular Computing with Memory Accelerators

Noboru Tanabe
Toshiba Corporation
Kawasaki, Japan
noboru.tanabe@toshiba.co.jp

Junko Kogou, Sonoko Tomimori, Masami Takata, Kazuki Joe
Nara Women's University
Nara, Japan
{vega, tomimori-sonoko0826, takata, joe}@ics.nara-wu.ac.jp

***Abstract*—Effective memory bandwidth for irregular applications such as a CG (Conjugate Gradient) solver and graph processing for larger sparse matrices must be accelerated with lower power in the future. Since the total performance of such HPC (High Performance Computing) applications is limited by memory bandwidth, smart memory is a possible lower power accelerator than GPUs (Graphics Processing Units). In this paper, we propose a GPU based HPC system using memory accelerators with gather functions and a HMC (Hybrid Memory Cube) interface. We implemented CG solver for it. The memory accelerator converts indirect accesses, which are unsuitable for cache and device memory, into direct accesses using gather functions. This paper presents the performance of the proposed memory architecture with University of Florida Sparse Matrix Collection. The result shows 1.01 to 1.20 times acceleration by the memory accelerator against the texture cache, even in the case of small matrices that take advantage of texture cache effects. The ratio will dramatically increase when the gap of the cache capacity and the matrices size increases. The scalability of the proposed method is guaranteed by the scalable broadcast thorough interconnection network.**

Keywords-high performance computing; memory architecture; smart memory; irregular processing; CG solver

I. INTRODUCTION

The computation ability of vector processor based supercomputers can be substituted with COTS (Commercial Off-The-Shelf) CPUs or GPUs in many cases. The computation ability of a high-end GPU reaches to one Tera FLOPS (FLoating point Operation Per Second) to be widely used for various applications known as GPGPU (General Purpose computing on Graphics Processing Units). The peak performance of GPUs seems to continuously and steadily increase in FLOPS according to the Moore's law [1]. If such performance progress is not given with sufficient device memory bandwidth for the peak performance, the memory wall problem gets more serious year by year. Hierarchical memory systems, typified by cache memory, do not address the problem when data reusability is small. However, in the case of some applications such as solving a system of linear equations consisting of kernels of SpMV (sparse matrix-vector multiplication), data reusability cannot be exploited so much. When matrices are small, such as a benchmark collection for a sparse matrix [2], the problem is not critical because GPU's cache works well for the small size matrices. The larger the target sparse matrices are, the more frequent and inefficient accesses to the external memory the cache needs to make, thus degrading performance. When large enough sparse matrices are to get regular accesses, most accesses can be converted to coalesced accesses of GPUs so that each cache miss takes a cache line with possible data to be accessed later. On the other hand, when the same sparse matrices are to get irregular accesses because the cache line size or the shortest burst length of GDDR5(Graphics Double Data Rate 5) memory is 128 bytes, it is reported that a cache miss exhausts the memory bandwidth 16 times (double precision) or 32 times (single precision) [3]. The line size of the last level cache (L2 cache) on a new generation GPU is larger than that of texture cache on older GPUs. Therefore, effective bandwidth degradation for huge SpMV (i.e. in the situation with low cache hit rates) of the newer GPU will be larger than that of the older GPUs.

To solve the above problem, an extended large capacity functional memory (memory accelerator) system with PCI (Peripheral Component Interconnect) express based interface and a set of scalable SpMV algorithms for GPUs are proposed in [3]. Although the experiment results of the SpMV algorithms and the functional memory system show four times performance improvement at a maximum, the contribution of the algorithms and the functional memory for the performance improvement is not described separately. In the meantime, they show that the bottleneck of the proposed algorithms and the functional memory lies in the PCI express.

The main contributions of this paper are summarized below:

- We propose new interfaces for the functional memory on a future GPU cluster to avoid the PCI express bottleneck, where Hybrid Memory Cube ports are promising.
- We analyze the relation between the cache hit rate and the matrix size for an SpMV executed on two kinds of GPUs. A tendency of hit rate degradation of texture cache and L1 cache is observed when row vector size increases.
- We implement a CG (Conjugate Gradient) solver including SpMVs for the proposed memory system to evaluate the performance improvement against a cache based system. Since we use the same algorithms for the evaluation, the improvement of memory system is estimated separately.
- During the execution of the CG solver on the proposed memory system, we get its breakdowns to detect hidden problems.

- To solve the above new problems, we improve the implementation based on CUBLAS, which is a numerical calculation library by Nvidia.

The rest of this paper is organized as follows: In Section 2, we introduce related works. In Section 3, we explain the architecture of our proposed memory system. In Section 4, the target workload (i.e. CG solver) on the proposed architecture is explained. We present the performance evaluation results in Section 5. In Section 6, we conclude the work and present future considerations.

II. RELATED WORKS

Recently, there are many research results that SpMV on GPU is accelerated to take the advantage of larger memory bandwidth of GPU rather than CPU [3]-[12]. So far, some sparse matrix libraries are already open to the public, such as in Nvidia [13][14]. While most studies are about storage formats for sparse matrix [3]-[12], some studies make use of GPU's texture cache for high-speed access to column vectors of the sparse matrix [4]-[12]. Although GPU's texture memory is read-only memory from the GPU, it can be used for storing column vectors, which are reusable, by the GPU with dedicated functions such as tex1Dfetch and tex2D to access the column vectors. Since the texture memory is cached on the texture cache, accesses to the device memory would be reduced if appropriate disposition of non-zero elements of sparse matrices are given.

Latest GPUs (e.g., Fermi architecture GPUs in the case of Nvidia) contain general purpose L1 and L2 caches for global memory on the device memory as well as texture memory. Using such general purpose caches, the column vectors can be cached without any dedicated functions to reduce device memory accesses.

Among many studies for accelerating SpMV, there are very few studies for accessing huge sparse matrices from many GPUs. In such studies, the huge sparse matrices should be decomposed for each device memory on the GPUs. Unless smart decomposition methods are used, considerable numbers of fine grain random communications, which degrade the scalability, are generated. In [5], the reduction of inter-GPU communication by hyper graph partitioning is reported, but the efficiency heavily depends on the matrix shape and/or the number of GPUs.

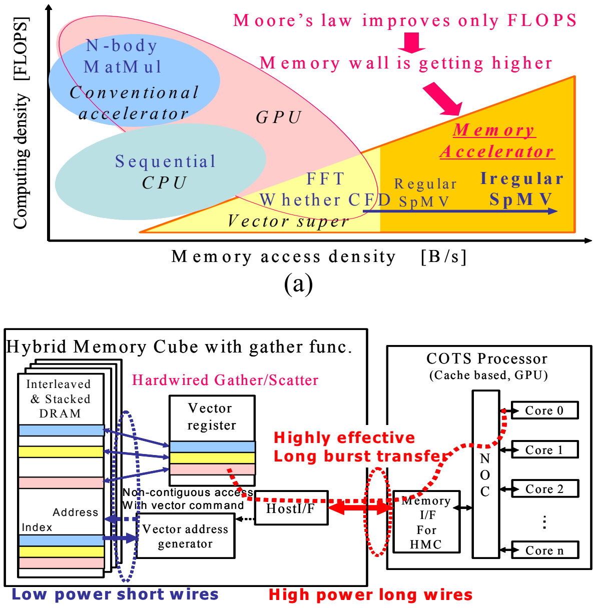
III. MEMORY ACCELERATOR

A. Basic concept

Figure 1(a) illustrates the mapping of applications and their suitable hardware accelerators categorized by the density of memory access and computation. The memory accelerator is hardware for the fast execution of memory bandwidth intensive applications such as irregular SpMVs that are difficult to be optimized for existing hardware accelerators. When a series of cache misses are issued, inefficient memory accesses where only four or eight bytes out of a 128 byte cache line are valid would be repeated. In such the case, the accesses to column vectors, which occupy one third of the total accesses, require the memory bandwidth 32 times for single precision and 16 times for double precision.

The memory accelerator has a memory controller with hardwired scatter/gather functions on the memory-side, i.e. between a block of external memory chips and a network-on-chip (NoC). The hardwired scatter/gather functions on the memory-side have been implemented in DIMMnet-2[15]. Figure 1(b) gives the concept of our proposed system including memory chips connected by many memory channels with a small number of wires for random memory accesses.

Recently, a memory subsystem that supports gather/ scatter capabilities is announced as a focus area [16] by IAA which is an organization for an Exa-FLOPS machine of the United States.



(a)

Hybrid Memory Cube with gather func.
Interleaved & Stacked DRAM
Hardwired Gather/Scatter
Vector register
Address
Index
Non-contiguous access With vector command
Vector address generator
HostI/F
Highly effective Long burst transfer
COTS Processor (Cache based, GPU)
Memory I/F For HMC
NOC
Core 0
Core 1
Core 2
Core n
Low power short wires
High power long wires

(b)

Figure 1. The concept of Memory accelerator: (a) Pourpose, (b) Proposed architecture.

B. Architectural improvement

In this paper, we propose an architectural improvement to the architecture of [3], where the interface of the memory accelerator is PCI express. Although it is reported in [3] that four times acceleration is observed with the combinatorial use of memory accelerator and pre-processing algorithms, it turned out that the bottleneck lies in the PCI express. As a short-term solution, we have adopted PCI express [3] as the host interface for the memory accelerator. For the middle and long term solutions, we are to adopt the GDDR5 device memory interface and the HMC (Hybrid Memory Cube) [17] interface, respectively. The current GDDR5 DRAM (Dynamic Random Access Memory) provides the bandwidth of 28GB/s.

We propose that the memory accelerator is implemented with 3D stacking as an HMC to integrate its hardwired scatter/gather functions inside the logic base chip of the HMC as shown in Fig. 1(b). Since each host interface of the

HMC provides 160 to 320GB/s bandwidth, the bottleneck problem would be resolved. The HMC with gather functions can randomly and effectively access many narrow memory banks using short low powered wires. It transmits just effective data on long off-chip wires. Therefore, the proposed memory system architecture must be low power and have high performance. Since the gather function can be implemented on a logic base of the HMC, the additional cost for the proposed architecture must be considerably smaller than that of conventional vector supercomputers. This improved memory can be a candidate for a new accelerator of future power constrained HPC platforms.

The cost of the proposed architecture with HMC interface is very low, since HMC essentially has a logic base chip which can easily have proposed gather functions. HMC is seen to be a low cost commonly used memory in the future. If the proposed logic is listed in the standard of future HMC, the hardware cost of the proposed architecture can be negligible.

IV. TARGET WORKLOAD FOR THE PROPOSED SYSTEM

In this paper, we use a CG solver for the evaluation of the proposed system as one of the target workloads. SpMV is the main part of the CG solver. In [3], an inter GPU communication reduction method for SpMV with keeping high scalability against the matrix shape and the number of GPUs is presented. The approach in [3] presents one of the best SpMVs from comprehensive standpoint considering scalability, load balancing and memory access efficiency.

As shown in Figure 2, an SpMV can be divided into multiplications of a set of row vectors and a column vector. Since there is no data dependence among the multiplications, it is quite easy to decompose the sparse matrix for each GPU by row vector according to the memory capacity constraint. The SpMV algorithm of [3] guarantees the scalability to remove the inter node communications by the strategy shown in Figure 2. In the case of the proposed architecture for accelerating CG, the resultant column vectors must be written back to the memory accelerator. In the case of multiple memory accelerators, the resultant column vectors must be multicasted to each memory accelerator. This multicast can be implemented so that it does not depend on the number of memory accelerators by using appropriate interconnection networks. Furthermore, the application of streaming would overlap the execution of the SpMV and multicast data transfer of resultant column vectors.

Furthermore, [3] proposes some pre-processing algorithms shown in Figure 3 to accelerate the performance of SpMVs. The pre-processes are padding, folding, and transposition to get better disposition of non-zero elements of the sparse matrix to GPUs. The use of the pre-processes in combination with the proposed hardware gives a maximum performance improvement of four times compared to [7]. In this paper, we follow [3] to use the SpMV algorithms and the hardware except the host interface of the GPUs.

In the CG solver, two parameters are generated from inner product operations to dense matrices, and using the two parameters each column vector of a sparse matrix is updated to apply SpMVs. The operation number of the dense matrices inner product is proportional to the number of unknowns of the simultaneous equations (the row number of the coefficient matrix), and the number of SpMVs is proportional to the number of non-zero elements of the coefficient matrix. It depends exactly on the disposition of non-zero elements. In general, the more non-zero elements per row the sparse matrix has, the more computation for the SpMVs the total operations contain. The less non-zero elements per row, the more computation there is for dense matrix inner product. Especially in the latter case, if the dense matrix inner product is not performed at the GPU but sent to the host to be calculated, it seems to have a risk to move the bottleneck to the host as well as data transfer overhead. Therefore, all the calculation should be performed in the GPU.

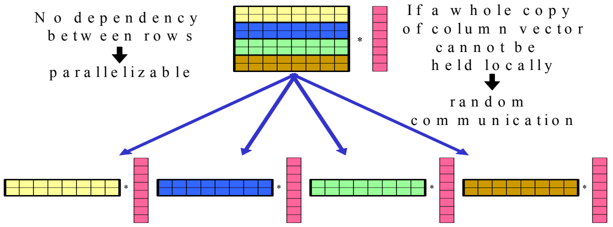

No dependency between rows
parallelizable
*
If a whole copy of column vector cannot be held locally
random communication

Figure 2. Strategy of scalable Sparse Matrix-Vector Multiplication in [3].

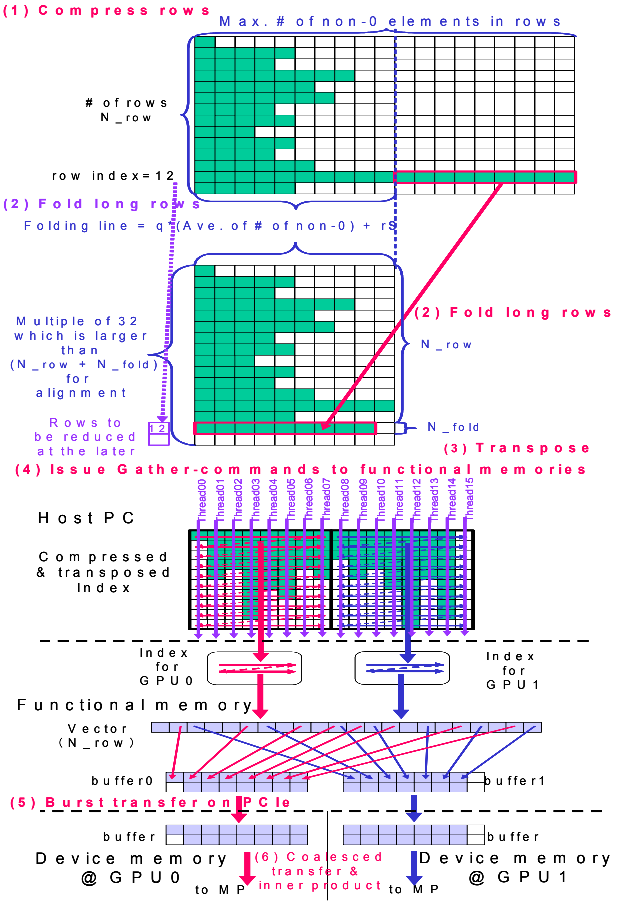


Figure 3. Flow of SpMV in [3].

V. EVALUATION

A. *Evaluation environment and test matrices*

The evaluation environment is illustrated in Table I (Tesla C1060) and Table II (Tesla C2050). Table III shows the test matrices. The test matrices are chosen from University of Florida Sparse Matrix Collection [2]. Although these sparse matrix problems are insufficient for our goal, namely not "so large problems that each vector cannot be stored in GPU device memory", we assume that the test matrices have the same characteristics to the too large vectors distributed among GPUs of the proposed system for parallel execution. In previous work [8], the same kinds of matrices were chosen for the performance evaluation of SpMV. In this paper, we chose the sparse matrices in [8]. Note that the chosen test matrices are not large enough. The size of the largest vector to be multiplied is so small (6.3MB) as to be insignificant compared with the device memory size. Therefore, we can say that the evaluation experiments are performed in a condition that cache effect is stronger (the condition that the cache-based previous work is profitable) than the finally supposed case (the condition that the problem is too large). Actually our method does not need cache effect so much.

TABLE I. EVALUATION ENVIRONMENT (C1060)

CPU	Intel®[a] Core(TM) i7 CPU920 @ 2.67GHz
GPU	Nvidia Tesla C1060 (# of core=240, 4GB, Memory bandwidth103GB/s)
Host I/F	PCI express x16 Gen2 (bandwidth 8GB/s)
OS	Fedora10
CUDA	Cuda3.0

TABLE II. EVALUATION ENVIRONMENT (C2050)

CPU	Intel® Xeon®[a] CPU X5670 @ 2.93GHz
GPU	Nvidia Tesla C2050 (# of core=448, 3GB, Memory bandwidth144GB/s)
Host I/F	PCI express x16 Gen2 (bandwidth 8GB/s)
OS	Red Hat Enterprise Linux Client release 5.5
CUDA	Cuda3.2
ECC	Off

a. Intel, Intel Core, and Xeon are trademarks of Intel Corporation in the U.S. and/or other countries.

TABLE III. TEST MATRICES

Name	**# of rows**	**Non-0 elements**			
		Total	***Ave.***	***Max***	*σ*
Na5	5,832	155,731	26	185	35.7
msc10848	10,848	620,313	57	300	49.4
thermal2	147,900	3,489,300	23	27	6.9
hood	220,542	5,494,489	24	51	13.3
F1	343,791	13,590,452	39	306	20.0
ldoor	952,203	23,737,339	24	49	12.9
G3_circuit	1,585,478	4,623,152	2	4	2.2

When we solve a large problem for the same kind of applications, it is expected that the number of non-zero elements per row of the sparse matrices is considered as a constant for most cases. When a large matrix is decomposed for multiple GPUs by row based on the strategy described in Figure 2, the memory accesses to the matrix issued by each GPU does not change so much when the matrix size is small. Since the indirect referenced column vector size increases when the problem size is large, the cache hit rate of existing cache based systems deteriorates. In the meantime, in the case of the proposed method, such performance degradation does not occur when the column vector size increases because the proposed method is based on a vector processor architecture. Furthermore, the strategy described in Figure 2 does not include any point-to-point communication, namely it is scalable. In short, the effect of the proposed method to the performance gain for small matrices to be evaluated for a GPU gives the lower limit of the performance gain for large matrices to be evaluated for GPU clusters.

B. *Implementation of CG solver*

For the evaluation of the proposed architecture, we implement three types of CG solvers with different accesses to the column vectors in the kernel, as explained below. In either case, pre-processes described in the previous section are applied to perform SpMV so that we analyze the relation between the access types of vectors and acceleration efficiency.

1) Texture memory version: In this version, column vectors are stored in GPU texture memory so that Tex2D function is called to make use of texture cache. This version gives the criteria of performance to be compared with other versions, and the number of iterations to be converged in this version is given to (3) as described later.

2) Shared memory version: In this version, column vectors on device memory are accessed via shared memory. In the case of Fermi C2050, those accesses are accelerated by L1 and L2 caches.

3) Proposed architecture version: In this version, column vectors on device memory, which get disposition operations in advance (fairing and transposition), are accessed. Because of the disposition operations, the original indirect references are converted to direct references in the source code so that they are in the form of burst and coalesced access. We assume that we have enough memory bandwidth of memory accelerator.

Note that the proposed method assumes the use of the mixed precision iterative refinement algorithm [19]. Although the algorithm [19] mostly consists of a single precision CG solver, it provides rich convergence ability to be comparable with double precision operations. For this reason, we measure the execution speed of a single precision CG solver in this evaluation. In the case that there is a matrix not to be converged with single precision operations, we just measure the execution speed of the loop which includes the matrix for a fixed number of interactions.

C. *Texture cache hit rate*

We measure the texture cache hit rate of the texture memory version to be executed on a C1060 with profiling. CUDA 3.0 provides performance counters tex_cache_hit and tex_cache_miss. Figure 4 shows the relation between the matrix size (the number of rows) and the texture hit rate. As the number of rows increases, the texture cache hit rates decreases because column vector accesses protrude the small texture cache. In Figure 4, we show the linear approximation by the black line, which shows a precipitously falling diagonal stroke from top left to bottom right. The linear approximation indicates that when the matrix size increases further, any cache effect is not expected.

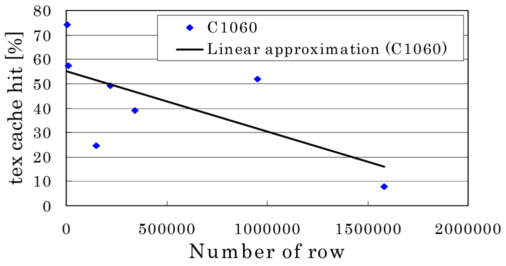


Figure 4. Matrix size and the texture hit rate.

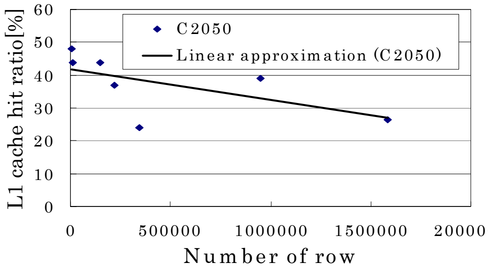


Figure 5. Matrix size and the L1 cache hit rate.

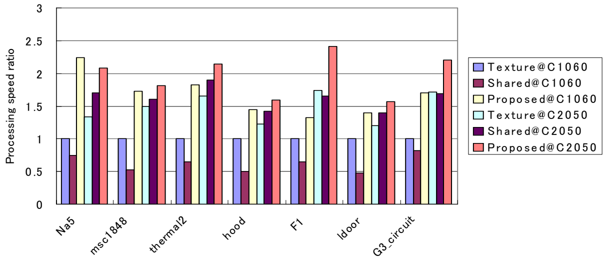


Figure 6. Processing speed ratio of SpMV.

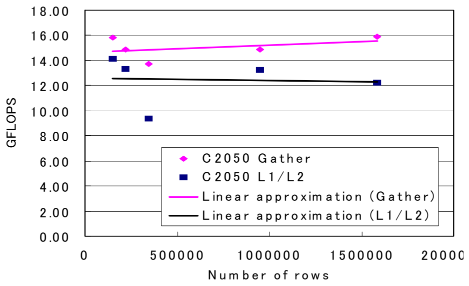


Figure 7. Matrix size and GFLOPS of SpMV.

Actually, among the matrices in this experiment, the largest one G3_circuit (1,585,478 rows) gets the cache hit rate of 7.74%, which reveals that the texture cache has spilt out. In the meantime, in G3_circuit the average number of non-zero elements per row is 2, and this means there are very few reusable data in each cache line, which makes the cache hit rate low, too.

D. General purpose cache hit rate

We measure the general purpose cache (L1) hit rate of the shared memory version to be executed on a C2050 with profiling. CUDA 3.0 provides performance counters L1_global_load_hit and L1_global_load_miss.

Figure 5 shows the relation between the matrix size (the number of rows) and the L1 cache hit rate, where the preference of L1 cache is LARGER (L1 cache is 48KB while shared memory is 16KB). As in the case of texture cache, the L1 cache hit rate decreases when the matrix size is large. G3_circuit gets the cache hit rate of 26.5% which is better than 7.74% on C1060. In the case of F1, the L1 cache hit rate is 23.9% which is lower than the texture cache hit rate. This phenomenon can be explained as follows. The L1 cache tries to keep any row vectors even if they do not have reusable data. Since F1 contains a large number of non-zero elements, F1 gets the lower cache hit rate as the result. To avoid this situation, when the row vector without reusable data is loaded, a special instruction, which skips L1 cache to directly load vector data, should be used.

E. SpMV execution times for three implementations

Figure 6 shows the processing speed ratio of each execution time of SpMV kernel to the execution time of the texture memory version on C1060. At a glance, it turns out that the proposed architecture version on both C1060 and C2050 is better than other implementations. Note that the effect of the additional hardware for the proposed architecture is very limited because the target matrices are so small. In other words, the proposed architecture provides more throughputs without any cache effect than the throughputs of GPUs with a certain amount of texture or L1 cache effect for relatively small matrices. Since the previous experiment shows that the use of larger matrices decreases the cache effect, the proposed architecture provides much more throughput for larger matrices compared with other implementations.

Considerable performance improvement is observed also in the shared memory version on C2050 compared with C1060 because of the improvement of device memory bandwidth and L1/L2 cache effect on C2050. This performance improvement seems to be given mainly by L2 cache effect. Since L2 cache is not so large compared with L1 cache, the performance improvement is limited when the target matrices are larger.

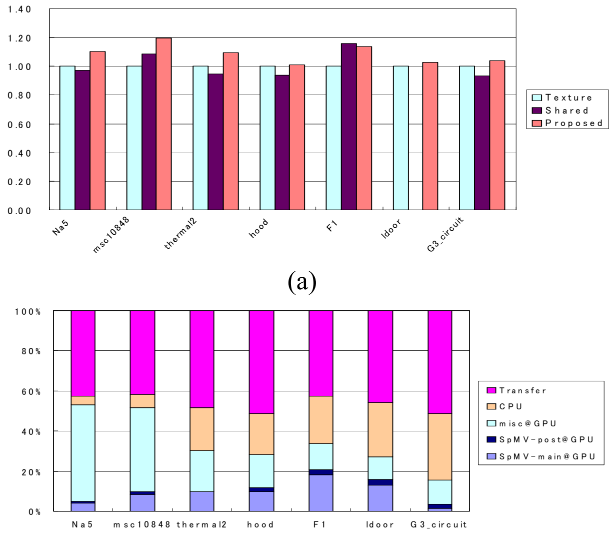
In [10], M. M. Baskaran et al. reports the execution times of F1 and ldoor with double precision SpMV in the JDS format on C2050. Our shared memory version on C2050 is 4.1 times and 2.74 times faster than in [10] for F1 and ldoor, respectively. Although our shared memory version is single precision that naturally makes two times performance improvement because of the difference of the device memory bandwidth (double vs. single), the pre-processing (folding) of our shared memory version achieves more performance improvement than their implementation.

Figure 7 shows the relation between matrix size and GFLOPS of SpMV on C2050. The shape of the graph of GFLOPS values with L1/L2 is similar to that of the L1 cache hit rate (Figure 5). We observe a tendency of performance degradation for L1/L2 as the matrix size increases. On the contrary, we observe an improving tendency of matrix size for Gather. This result predicts that the performance ratio between Gather and L1/L2 cache will increase when the gap of cache capacity and solution vector size increases.

F. CG solver execution times for three implementations

When the SpMVs are fully accelerated on the GPU, the inner product operations performed on the host PC become a bottleneck. The inner product of dense matrices can be calculated in parallel, and seems to be accelerated using CUBLAS [18]. After the above consideration, we apply CUBLAS to the inner product operations and move most of the miscellaneous operations (except residual operations to be executed on the host PC) to the GPU. Figure 8(a) shows the speed ratio of the CG solver using CUBLAS applied to various matrices on C2050 with three implementations.

Each matrix is symmetric but some matrices are not positive definite. Furthermore, they are single float operations. So, the number of iterations is totally different by matrix, and some matrices cause the CG solver not to converge. Taking into account the above problems, we measure partial iterations to calculate the average execution time by iteration. Since the shared version program could not be executed for ldoor on our environment, the result is not available.



(a)

(b)

Figure 8. Performance of the CG solver using CUBLAS based inner products on C2050: (a) Speed ratio, (b) Breakdown.

The result is that the proposed architecture version works 1.01 to 1.20 times faster than texture memory version although the matrices are small enough to take advantage of the texture cache effect. These speed ratios are not attainable by the kernels shown in Figure 6. This is because operations other than SpMVs and inner products are executed on the host PC.

To investigate the performance effect of non-kernel operations, we measure other operations of the CG solver. Figure 8(b) shows the breakdown of the execution times by our proposed architecture using CUBLAS based inner products on C2050. It turns out that the post SpMV operations which are reductions of partial inner products of folded vectors and the main part of SpMVs do not take the major part of the processes, but the other calculations on the CPU and the data transfer between the host PC and the GPU dominate the total processes. This is one of the reasons why the speed-up ratio in Figure 8(a) is not so large. The second reason why the speed-up ratio in Figure 8(a) is not so large is that the workload matrix is too small for the L2 cache hit rate and the device memory bandwidth to dominate the performance. If workload matrices are large enough that these parameters dominate the performance, the speed-up ratio must be increased since the proposed memory system is based on the vector architecture whose performance does not decrease when the vector length is larger.

The amount of inner product operations depends on unknown variables of the linear equation. In the case of G3_circuit where the average number of non-zero elements per row is two, the amount of SpMVs is almost as same as inner product operations. This is the reason why miscellaneous operations dominate the execution time of G3_circuit. As the result of reducing CPU execution times, the data transfer between the host and the GPU comes to dominate the total execution time. If the complete optimization, namely all operations on a GPU such as Kepler by Nvidia, was achieved, the data transfer between the host and the GPU would be reduced and higher acceleration would be expected.

VI. CONCLUSIONS AND FUTURE WORK

In this paper, we evaluate texture and general purpose cache hit rates for conventional SpMVs. We confirm that relatively small matrices (several hundred Kilo Byte to several Mega Byte) taken from University of Florida Sparse Matrix Collection [2] generate cache hit rates of 10% to 70%, which are low. Larger matrices tend to degrade cache hit rates and FLOPS performance.

To keep high scalability of the number of GPUs, each GPU should contain a copy of the target column vector in its device memory. When the target is a mesh of 1,000 cubic, the column vector size becomes 8GB and exceeds the capacity of the device memory. In this case, the overhead of data transfer via PCI express would make the performance worse. The memory accelerator, which has gather functions and a huge capacity of memory optimized for short bursts, has been proposed in [3]. The memory accelerator can replace cache memory so that it treats huge problems that matrices and column vectors are too large to be stored in the device memory.

The problem of the memory accelerator in [3] is that SpMV on GPUs reveals PCI express is the bottleneck. In this paper, we proposed that a memory accelerator is connected to the GDDR5 port or the HMC port of GPUs. In these cases, the bandwidth per port increases, and the total bandwidth additionally increases using multiple ports.

The previous evaluation could not explain the performance effects by replacing cache memory with the memory accelerator and by pre-processing algorithms separately. In this paper, we evaluate the performance effects on University of Florida Sparse Matrix Collection [2] with different memory systems including the memory accelerator. As a result, even in the case of small matrices where texture cache is effective, it turned out that our proposed architecture works 1.01 to 1.20 times faster than the texture cache based existing method. If workload matrices are large enough that the L2 cache hit rate and the device memory bandwidth dominate the performance, speed-up ratio will be greater since proposed memory system is based on vector architecture whose performance does not decrease when the vector length is larger. If the complete optimization, namely all operations on a GPU such as Kepler by Nvidia, was achieved, higher acceleration would be expected.

While the previous evaluation was just for SpMV, in this paper we evaluate the CG solver including SpMV. In the CG method execution, the full acceleration of SpMV on a GPU exposes the other processes of dense matrix inner product on a CPU and data transfer latency between the CPU and the GPU. Shifting some part of the miscellaneous processes to the GPU side, we observe considerable performance improvement.

Our future work includes the implementation and evaluation of streaming that hides the write-back latency for the memory accelerator, the exact evaluation of L2 cache effect for larger matrices, the evaluation of the scalability of many GPUs and memory accelerators, and the design and evaluation of a memory accelerator with large capacity optimized with short bursts.

ACKNOWLEDGMENT

A part of this work is supported by the Ministry of Internal Affairs and Communications (Soumu-sho).

REFERENCES

[1] R. R. Schaller : Moore's law: past, present and future, Spectrum, IEEE , Vol.34, No.6, pp.52,59 (1997)

[2] Tim Davis : "The University of Florida Sparse Matrix Collection", http://www.cise.ufl.edu/research/sparse/matrices/ [retrieved March 2013]

[3] N. Tanabe, Y. Ogawa, M. Takata, K. Joe : Scaleable Sparse Matrix-Vector Multiplication with Functional Memory and GPUs, 19th Euromicro Conference on Parallel, Distributed and Network-Based Computing (PDP 2011), pp. 101-108 (2011)

[4] X. Yang, S. Parthasarathy, P. Sadayappan : Fast Sparse Matrix-Vector Multiplication on GPUs: Implications for Graph Mining, 37th International Conference on Very Large Data Bases (VLDB2011), pp.231-242 (2011)

[5] A. Cevahir, A. Nukada, S. Matsuoka : High performance conjugate gradient solver on multi-GPU clusters using hypergraph partitioning, Computer Science - Research and Development, Vol.25, No.1-2, pp.83-91 (2010)

[6] M. Ament, G. Knittel, D. Weiskopf, W. Straser : A Parallel Preconditioned Conjugate Gradient Solver for the Poisson Problem on a Multi-GPU Platform, 18th Euromicro Conference on Parallel, Distributed and Network-based Processing (PDP 2010), pp.583-592 (2010)

[7] A. Cevahir, A. Nukada, S. Matsuoka : An Efficient Conjugate Gradient Solver on Double Precision Multi-GPU Systems, Symposium on Advanced Computing Systems and Infrastructures (SACSIS2009), pp.353-360 (2009)

[8] Y. Kubota, D. Takahashi : Optimization of Sparse Matrix-Vector Multiplication by Auto Selecting Storage Schemes on GPU, Computational Science and Its Applications(ICCSA 2011), LNCS Vol. 6783, pp.547-561 (2011)

[9] N. Bell, M. Garland : Eficient Sparse Matrix-Vector Multiplication on CUDA, Nvida Technical Report NVR-2008-004 (2008)

[10] M. M. Baskaran, R. Bordawekar : Optimizing Sparse Matrix-Vector Multiplication on GPUs, IBM Research Report, RC24704 (2009)

[11] A. Monakov, A. Lokhmotov and A. Avetisyan : Automatically Tuning Sparse Matrix-Vector Multiplication for GPU Architectures, 5th International Conference on High Performance Embedded Architectures and Compilers (HiPEAC 2010), LNCS 5952, pp.111-125 (2010)

[12] F. Vazquez, G. Ortega, J. J. Fernandez and E. M. Garzon : Improving the Performance of the Sparse Matrix Vector Product with GPUs, 10th IEEE International Conference on Computer and Information Technology (CIT 2010), pp.1146-1151 (2010)

[13] Nvida : CUSPARSE User Guide, http://docs.nvidia.com/cuda/pdf/CUDA_CUSPARSE_Users_Guide.pdf [retrieved March 2013]

[14] - : cusp-library : Generic Parallel Algorithms for Sparse Matrix and Graph Computations, http://code.google.com/p/cusp-library/ [retrieved March 2013]

[15] N. Tanabe, M. Nakatake, H. Hakozaki, Y. Dohi, H. Nakajo, H. Amano : A New Memory Module for COTS-Based Personal Supercomputing, International Workshop on Innovative Architecture for Future Generation High-Performance Processors and Systems (IWIA2004), pp.40-48, DOI: 10.1109/IWIA.2004.10019 (2004)

[16] IAA: Focus area, http://iaa.sandia.gov/focus-areas/ [retrieved March 2013]

[17] Micron Technology, Inc. : Hybrid Memory Cube : Breakthrough DRAM Performance with a Fundamentally Re-Architected DRAM Subsystem, Hot Chips 23 (2011)

[18] Nvida : CUBLAS User Guide, http://docs.nvidia.com/cuda/pdf/CUDA_CUBLAS_Users_Guide.pdf [retrieved March 2013]

[19] A. Buttari, J. Dongarra, J. Kurzak, P. Luszczek, S. Tomov : Using Mixed Precision for Sparse Matrix Computations to Enhance the Performance while Achieving 64-bit Accuracy. ACM Transactions on Mathematical Software, Vol.34, No.4 (2008)