INTENSIVE 2013

The Fifth International Conference on Resource Intensive Applications and Services

ISBN: 978-1-61208-258-5

March 24 - 29, 2013

Lisbon, Portugal

INTENSIVE 2013 Editors

Petre Dini, Concordia University, Canada / China Space Agency Center, China

INTENSIVE 2013

Foreword

The Fifth International Conference on Resource Intensive Applications and Services [INTENSIVE 2013], held between March 24 - 29, 2013 in Lisbon, Portugal, continued a series of international events covering a spectrum of topics related to technologies, hardware, software and mechanisms supporting intensive applications and services (RIAS).

Intensiveness is a qualitative concept of expressing the degree of resources needed to fulfill a given task under strong requirements of either communication, computation, storage or simply data-volume where solutions are time-critical or have a mass impact. The well-known computation/resource intensive paradigm portrays a paradigm shift with the advent of high-speed applications, on-line multi-user game services, Grid applications and services, or on-demand resources and services. With the heavy distributed and parallel applications, communication intensive aspects, such as bandwidth-intensive and propagation intensive, became key contributors for optimizing workflows of computation of intensive tasks, or storage and access-intensive applications. For example, the massive scalability and storage capacity make it the clear choice for replication-intensive scenarios; the bandwidth-intensive becomes relevant for content streaming systems, while replication and data reuse are important for data-intensive applications on Grids.

There are monitoring and control aspects related to intensive applications and services aligned to different technologies. To deal with performance, scalability, stability, and accuracy (as some aspects may be NP-complete), different mechanisms and solutions were considered in terms of heuristics for relaxing the intensiveness, optimization, approximation or suboptimal solutions.

Associated with scalability, digital signal processors for computation intensive statistics and simulation relate to new hardware and software supporting the concept of ‘intensive’. Other technologies requiring real-time decoding, mobility, and wireless make the systems computationally very intensive. Performance intensive software is increasingly being used on heterogeneous combinations of OS, compiler, and hardware platforms.

We take here the opportunity to warmly thank all the members of the INTENSIVE 2013 Technical Program Committee, as well as the numerous reviewers. We also kindly thank all the authors who dedicated much of their time and efforts to contribute to INTENSIVE 2013. We truly believe that, thanks to all these efforts, the final conference program consisted of top quality contributions.

We are grateful to the members of the INTENSIVE 2013 organizing committee for their help in handling the logistics and for their work to make this professional meeting a success.

We hope that INTENSIVE 2013 was a successful international forum for the exchange of ideas and results between academia and industry and for the promotion of progress for resource intensive applications and services.

We are convinced that the participants found the event useful and communications very open. We also hope the attendees enjoyed the charm of Lisbon, Portugal.

INTENSIVE 2013 Chairs:

Chih-Cheng Hung, Southern Polytechnic State University, USA
Rainer Schmidt, Austrian Institute of Technology, Austria
Simon Tsang, Applied Communication Sciences, USA
Ouri Wolfson, University of Illinois - Chicago, USA
Alvaro Arenas, IE Business School, Spain
Kaustubh Joshi, AT&T Labs Research - Florham Park, USA
Meikel Poess, Oracle, USA
Arun Saha, Fujitsu Network Communications, USA

INTENSIVE 2013

Committee

INTENSIVE Advisory Chairs

Chih-Cheng Hung, Southern Polytechnic State University, USA
Rainer Schmidt, Austrian Institute of Technology, Austria
Simon Tsang, Applied Communication Sciences, USA
Ouri Wolfson, University of Illinois - Chicago, USA

INTENSIVE Industry/Research Chairs

Alvaro Arenas, IE Business School, Spain
Kaustubh Joshi, AT&T Labs Research - Florham Park, USA
Meikel Poess, Oracle, USA
Arun Saha, Fujitsu Network Communications, USA

INTENSIVE 2013 Technical Program Committee

Scott Alexander, Applied Communication Sciences, USA
Giner Alor Hernandez, Instituto Tecnologico de Orizaba - Veracruz, México
Budak Arpinar, University of Georgia, USA
Rudolf Berrendorf, Bonn-Rhein-Sieg University of Applied Sciences - Sankt Augustin, Germany
Jonathan Blackledge, Dublin Institute of Technology, Ireland
Fernando Boronat Seguí, Universidad Politécnica de Valencia, Spain
Gerardo Canfora, University of Sannio - Benevento, Italy
Li-Der Chou, National Central University - Taipei, Taiwan, ROC
Claudio de la Riva, Univeristy of Oviedo, Spain
Nicholas John Dingle, Imperial College London, UK
Juan Carlos Dueñas López, Universidad Politécnica de Madrid, Spain
Hans-Dieter Ehrich, Technische Universität Braunschweig, Germany
Paul Gibson, TELECOM SudParis, France
Vic Grout, Glyndwr University - Wrexham, UK
Jameleddine Hassine, King Fahd University of Petroleum & Minerals (KFUPM), Saudi Arabia
Wolfgang Hommel, Leibniz-Rechenzentrum - Garching, Germany
Paul Humphreys, University of Ulster, UK
Chih-Cheng Hung, Southern Polytechnic State University, USA
Félix J. García Clemente, Universidad de Murcia, Spain
Jon Hall, Open University, UK
Robert J. Hilderman, University of Regina, Canada
Jinlei Jiang, Tsinghua University - Beijing, China
Alexander Jungmann, University of Paderborn, Germany
Scott A. Klasky, ORNL, USA
Evangelos Kranakis, Carleton University - Ottawa, Canada
Erwin Laure, KTH, Sweden
Mario Marcelo Berón, National University of San Luis (Argentina) / University of Minho, Portugal

Copyright Information

For your reference, this is the text governing the copyright release for material published by IARIA.

The copyright release is a transfer of publication rights, which allows IARIA and its partners to drive the dissemination of the published material. This allows IARIA to give articles increased visibility via distribution, inclusion in libraries, and arrangements for submission to indexes.

I, the undersigned, declare that the article is original, and that I represent the authors of this article in the copyright release matters. If this work has been done as work-for-hire, I have obtained all necessary clearances to execute a copyright release. I hereby irrevocably transfer exclusive copyright for this material to IARIA. I give IARIA permission or reproduce the work in any media format such as, but not limited to, print, digital, or electronic. I give IARIA permission to distribute the materials without restriction to any institutions or individuals. I give IARIA permission to submit the work for inclusion in article repositories as IARIA sees fit.

I, the undersigned, declare that to the best of my knowledge, the article is does not contain libelous or otherwise unlawful contents or invading the right of privacy or infringing on a proprietary right.

Following the copyright release, any circulated version of the article must bear the copyright notice and any header and footer information that IARIA applies to the published article.

IARIA grants royalty-free permission to the authors to disseminate the work, under the above provisions, for any academic, commercial, or industrial use. IARIA grants royalty-free permission to any individuals or institutions to make the article available electronically, online, or in print.

IARIA acknowledges that rights to any algorithm, process, procedure, apparatus, or articles of manufacture remain with the authors and their employers.

I, the undersigned, understand that IARIA will not be liable, in contract, tort (including, without limitation, negligence), pre-contract or other representations (other than fraudulent misrepresentations) or otherwise in connection with the publication of my work.

Exception to the above is made for work-for-hire performed while employed by the government. In that case, copyright to the material remains with the said government. The rightful owners (authors and government entity) grant unlimited and unrestricted permission to IARIA, IARIA's contractors, and IARIA's partners to further distribute the work.

Table of Contents

Interference Control by Best-Effort Process Duty-Cycling in Chip Multi-Processor Systems for Real-Time Medical Image Processing

Mark Westmijze, Marco J. G. Bekooij and Gerard J. M. Smit
University of Twente, Department of EEMCS
Enschede, The Netherlands
{m.westmijze, m.j.g.bekooij, g.j.m.smit}@utwente.nl

***Abstract*—Systems with chip multi-processors are currently used for several applications that have real-time requirements. In chip multi-processor architectures, many hardware resources such as parts of the cache hierarchy are shared between cores and by using such resources, applications can significantly interfere with each other. In previous work, we showed that a single X-ray imaging streaming applications can be executed with low jitter on such systems. However, it was assumed that only one application would be running on the system, which prevents system integration where multiple real-time and best-effort applications are executing on a single chip multi-processor. In this paper, we address the limited bandwidth in the cache hierarchy, which can cause threads to interfere with each other significantly. We propose a technique that implements cache bandwidth reservation in software, by dynamically duty-cycling best-effort applications, based on their cache bandwidth usages using processor performance counters in order to control the influence of best-effort applications on real-time applications. With this technique we can control the latency increase of real-time applications that is caused by best-effort application in order to satisfy real-time requirements with a minimal reduction in best-effort performance. The results of the experiments with real-life applications indicate that we can control the increase of the latency to such an extent that we can almost completely eliminate the influence of bandwidth sharing in the cache at the cost of best-effort performance.**

Keywords-cache; bandwidth; real-time; control; jitter

I. INTRODUCTION

Commercial-off-the-shelf (COTS) systems with general purpose processors (GPPs) have become so powerful that they can be used for applications that could previously only be performed on hardware such as digital signal processors, field-programmable gate arrays (FPGAs) or by application specific integrated circuits (ASICs). However, the techniques that are employed by modern GPPs to reach their performance tend to neglect the worst case performance in order to improve the best-case and average-case performance. This seemingly does not make them a favorable target for real-time applications. A lot of work in the real-time community has been focused on the definition of architectures on which real-time applications can run predictably and modeling techniques that determine the worst case behavior of real-time applications. Applying these modeling techniques on high performance GPPs often results in loose throughput and latency bounds as a result of the GPPs and chip multi-processor (CMP) architecture. This leads to over-provisioned systems, which cannot be economically justified for most soft real-time applications.

We studied a streaming X-ray application on a CMP architecture in a previous paper [1]. In this paper, we provided evidence that it is feasible to run streaming applications in such a way that drastically reduces the occurrences of worst case behavior. The jitter | variation of latency | of the executed application is sufficiently small to satisfy the soft real-time requirements of the X-ray application. However, one important assumption was that the system was only used for the X-ray application. A consequence is that a separate system is required for additional applications while the system that runs the X-ray application has spare processing power most of the time. We therefore study whether we can relax that assumption, and run some best-effort application along with our real-time streaming application.

The problem that must be addressed is that threads that execute concurrently on different cores on a CMP architecture can degrade each others performance significantly. There are several components in a CMP that allow cores to degrade each others performance such as the cache hierarchy, memory controller and central processing unit (CPU) interconnect. In this paper, we focus on space contention and bandwidth congestion within the cache hierarchy, study how this enables applications to degrade each others performance and propose and evaluate a technique to control the degradation.

The paper is structured as follows. First, we introduce the case study | an interventional X-ray imaging system | in Section II and in Section III we briefly give an overview of the CMP architecture that we use. Related work is described in Section IV. This allows us to give a detailed explanation of why we study the space contention and bandwidth congestion in CMP architectures in Section VI and Section V. Section VII explains how we implemented a technique that reduces the bandwidth congestion and thereby controls the interference that is caused by the best-effort application. The experiments that we used for the evaluation can be found in Section VIII and the results of those experiments in Section IX. A few improvements of the control technique are described in Section X as future work. We summarize the conclusions in Section XI. Finally, we give our acknowledgments in Section XII.

 ISBN: 978-1-61208-258-5

II. X-RAY SYSTEM

In the X-ray system there are real-time and best-effort applications that currently run on multiple COTS systems. The real-time applications are streaming applications where images are typically processed at frame rates between 15 and 60 frames per second. For precise hand-eye coordination and to prevent fatigue, the processed X-ray images should preferably have a low latency (e.g., 120 ms) and the jitter on the latency should be low (e.g., âL'd' 16 ms). These requirements enable the physician to smoothly control his equipment. Practical experience has shown that the occasional deviations from the requirements are small enough that they do not affect the patient's safety. Due to the nature of these requirements we can use modeling techniques and architectures that work most of time instead of strictly all the time. Examples of best-effort applications are the graphical user interface (GUI), the storage of processed images and the retrieval and analysis of stored images.

Our aim is to run the real-time and best-effort applications on a single system while controlling the progress of the best-effort application such that real-time requirements are satisfied and the performance of the best-effort application is maximized.

III. ARCHITECTURE

As mentioned in Section II, we consider a CMP architecture. More specifically, we focus on the Intel Nehalem architecture. In Fig. 1 a schematical overview is given for such a CMP in a dual socket system. Each chip consists of a number of cores that all have simultaneous multi-threading (SMT) capability. An inclusive cache hierarchy with three levels is used where the first and second level are local for a physical core and the third level is shared between all cores. Furthermore, a memory controller and a high speed point-to-point interface (i.e. Quick Path Interconnect (QPI)) are integrated into the chip, which improves scalability in a multi-socket system.

In this paper, we only examine the case where threads of multiple applications are mapped onto a disjoint subsets of physical cores (e.g., the real-time threads on core 0 and 1 and a best-effort application on core 2 and 3 with SMT disabled). The first hardware component for which the threads have to contend is the bandwidth to and the space in the third level of the cache. Other components in the system may also be shared, but in this paper we focus on how the cache influences the performance of multiple concurrently executing threads.

IV. RELATED WORK

In many papers cache partitioning techniques have been proposed to improve the performance and predictability of the cache. Cache partitioning splits the cache in disjoint sets that do not share cache lines in the cache hierarchy. This prevents cache thrashing between threads. Stone et al. presented and used this technique in order to minimize the miss rate of each application [2]. Chiou *et al.* presented a technique for cache partitioning based on columnization [3]. Iyer studied cache

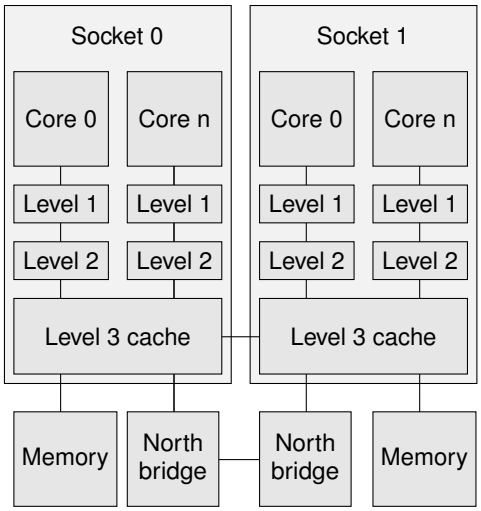

Figure 1. System architecture

partitioning in order to improve the Quality of Service (QoS) [4]. Kannan *et al.* studied how dynamic cache partitioning can be added to an operating system in order to improve the QoS for one or more applications and propose a methodology that can dynamically alter the cache partitioning at run-time [5]. Kim *et al.* also studied cache partitioning in a CMP in [6] and optimize the cache partitioning for fairness (i.e. an application with a small amount of cache accesses is penalized less compared to an application with a lot of cache accesses).

Cache partitioning is a form of performance isolation that at system level is also studied. Verghese *et al.* studied the performance isolation in CMP and focus on isolation in components such as the memory controller and disk access [7]. Nesbit *et al.* introduced Virtual Private Machines (VPMs) in [8], an abstraction supported by hardware modifications in the architecture of a CMP to enable applications to reserve a certain amount of hardware resources. In the cache hierarchy space and bandwidth can be reserved in order to reduce the interference between applications. In [9], Hansson *et al.* introduced an embedded architecture that also implements performance isolation (on clock cycle level), hardware arbitration, and hard real-time scheduling techniques in order to present a virtual platform that provides complete isolation for different applications.

However, to the best of our knowledge there is no prior work that addresses the interference that is caused by bandwidth sharing in COTS CMP architectures.

V. MOTIVATION

In previous work [1], we have shown that under certain conditions COTS can be used for real-time image processing applications. The conditions were:

- **Soft real-time** Because of the complexity of modern high-performance CMP architectures (e.g., Intel Nehalem) it can be hard or even impossible to derive tight worst case

execution times (WCETs). Therefore we do not formally derive the WCET, but we validate that the soft real-time constraints are met by measuring the actual jitter of the system.

- **Static streaming** The application that we examine executes static algorithms. Additionally, the application is streaming, which means that the same algorithms are executed repeatedly on new data. This allows us to map and order the algorithms and allocate the used memory of the application in such a way that less cache thrashing occurs, which is one of the main causes of jitter in CMP architectures.
- **Single application** On a system with only one application there is no contention on hardware resources with other applications.

In this paper, we want to relax the third condition in order to facilitate the execution of additional best-effort applications during the execution of the real-time applications. When two applications share hardware resources | which is the case in COTS | they can influence the execution time of each other. As mentioned before we focus on the allocation and bandwidth sharing in the cache and how that influences the interference between a real-time streaming application and other best-effort applications.

VI. Cache Partitioning

In Section IV we have mentioned that many authors have studied the partitioning of the cache as a means of performance isolation. With cache partitioning, threads do not longer evict cache lines from other threads (cache thrashing), ensuring that data remains in the cache and does not have to be retrieved from main memory. However, this technique only guarantees that other threads do not evict cache lines owned by a thread, not that the latency to retrieve that data is the same, because bandwidth to the third level of the cache is shared. Hence cache partitioning can be used to reduce some of the latency that is introduced by hardware sharing. However, it will not completely remove it. The contention can only be removed when the bandwidth to the shared cache is controlled. Which means there are two orthogonal problems: memory allocation in the cache (solved by partitioning and not studied in this paper) and bandwidth sharing to the shared level in the cache (studied in this paper).

Molka *et al.* [10] performed detailed benchmarking on the Nehalem architecture in order to determine the bandwidths between the core and different levels of the cache hierarchy. We extended those benchmarks so that we can determine how much bandwidth a single application can consume on the connection between the second and the shared third level of the cache. The most important observation from those experiments was that a single core is able to consume a significant portion of the cache bandwidth when it is writing (e.g., 15 GB/s write bandwidth for a single core that only writes where the maximal aggregated write bandwidth is 21.9 GB/s when three cores are writing at 7.3 GB/s). Application that only performs reads will only claim up to 26% of the available bandwidth on a Quad core (19.9 GB/s read bandwidth for a single core where the maximal aggregated read bandwidth is 78.8 GB/s).

With the following experiment we want to demonstrate that this is not the results of cache thrashing, but due to the bandwidth congestion. We recreated the experiments that were performed by Molka *et al* [10] with different memory configuration, namely:

- A The accessed data does not fit in the cache and the cache is not partitioned. The main memory will therefore be accessed and cores will thrash the cache.
- B The accessed data does not fit in the cache and the cache is partitioned. The main memory will be accessed, but cores will only evict their own cache lines.
- C The accessed data of the combined cores does fit in the cache and the cache is not partitioned. The main memory should only be accessed when cache thrashing occurs.
- D The accessed data does fit in the allocated cache partition. Main memory should not be accessed and cache thrashing does not occur between the cores.
- E Only the accessed data of the measured cores does fit into its allocated cache partition and the data of the other cores does not fit into the cache and they access data in one combined cache partition.

Furthermore, we run these five configuration in three different scenarios. In the three scenarios we either perform only data

TABLE I
PARTITIONING EFFECTS IN GB/S | A) DOES NOT FIT IN CACHE, NOT PARTITIONED. B) DOES NOT FIT IN CACHE, PARTITIONED. C) FITS IN CACHE, NOT PARTITIONED. D) FITS IN CACHE, PARTITIONED. E) ONLY MEASURED THREAD FITS IN CACHE, PARTITIONED

	Bandwidth (GB/s)														
	Read					Write					Mixed read (50%) and write (50%)				
Cores	A	B	C	D	E	A	B	C	D	E	A	B	C	D	E
1	11.5	11.5	19.9	19.9	19.8	6.3	6.3	15.0	15.0	15.1	7.1	7.1	18.2	18.2	17.5
2	6.9	7.9	19.8	19.8	16.0	3.1	3.9	10.5	10.6	8.6	4.8	4.9	16.4	16.5	12.8
3	7.0	5.6	19.7	19.8	10.4	2.7	2.5	7.3	7.3	5.1	3.3	3.3	11.5	11.5	7.4
4	4.8	4.4	17.2	19.7	8.0	2.6	1.9	5.4	5.4	4.2	3.7	2.6	8.7	8.8	5.3

 ISBN: 978-1-61208-258-5

read, data writes or a mix of data reads and writes. The results of this experiment can be found in Table I. From these experiments we make the following observations:

- When all the data fits in the cache and the cores are only reading (configuration C and D in the read scenario, i.e., columns 3 and 4) the read bandwidth per core is almost constant, regardless of the number of cores that are running. Only when all the cores are reading and the cache is unpartitioned we see a slightly lower read bandwidth (i.e., 17.2 GB/s for four partitioned cores versus 19.8 GB/s on average for the other cases).
- When the data does not fit in the cache (configuration A and B in all scenarios, i.e., columns 1, 2, 6, 7, 11 and 12) the read and write bandwidths are significantly degraded when more cores are running. This is caused by the limited bandwidth to main memory.
- When the data fits in the cache and the cores are writing (configuration C and D in the write scenario, i.e., columns 8 and 9) the aggregated bandwidth of the system is maximal 21.9 GB/s (7.3 GB/s per core on 3 cores) while a single thread can consume 15 GB/s essentially consuming 68% of the available write bandwidth.

Therefore we conclude that most performance degradation can be caused by the limited bandwidth, and in particular by the limited write bandwidth. Furthermore, cache partitioning does not necessarily reduce the interference, it may even degrade the performance in the situation where the performance of a memory intensive application is degraded more by the reduced cache space than by the cache thrashing of other applications.

VII. Cache Bandwidth

In order to reduce the interference of the best-effort applications on the real-time applications to an acceptable level, we have to be able to determine when this occurs. We first determine current bandwidth usage, and thereafter use this information to reduce the bandwidth of the best-effort application.

A. Bandwidth usage estimation

The system has to detect when a core uses more bandwidth than allocated. To measure this we selected two performance counters in the Nehalem architecture:

- LEVEL_2_TRANSACTIONS.LOAD: Each load transaction is counted. However, read transaction take less bandwidth than write bandwidth.
- LEVEL_2_LINES_OUT.ANY: Counts the number of lines that are written back to the third level of the cache.

The LEVEL_2_TRANSACTIONS.LOAD performance events could accurately estimate the read bandwidths as found in Table I. The LEVEL_2_LINES_OUT.ANY performance can be used to determine the total (read + write) bandwidth. We could not precisely estimating the write bandwidth in all scenarios with only one performance counter, but were able to roughly derive this value based on the total bandwidth and the read bandwidth. These counters can be used during the execution of the best-effort application to determine the currently used bandwidth and during design time to determine the worst case bandwidth usage of real-time and best-effort applications.

B. Bandwidth arbitration

There is no hardware mechanism that can allocate bandwidth within the cache hierarchy of the Nehalem architecture to specific cores. We therefore implemented a mechanism in software that can be used to achieve the same effect, but on a coarser time scale. Our mechanism to reduce the bandwidth on a coarser time scale is to repeatedly suspend the applications for a certain time on a core that uses more bandwidth than allocated, which we will now refer to as duty-cycling. This mechanism results in a much lower (average) bandwidth and reduced performance for the applications on the core that is temporarily suspended. With this technique it is not possible to completely remove the interference because the duty-cycling of cores manages the bandwidth on a coarse level and only reacts after a best-effort application has consumed too much bandwidth.

C. Duty-cycling

When the estimated read or write bandwidth of a core that is running best-effort applications reaches a certain threshold we assume that this core is degrading the performance of the real-time application and that the core that runs that offending best-effort application must be duty-cycled. We implemented this by running a thread, with real-time scheduling priorities, at a higher priority than the best-effort applications in order to temporarily suspend any best-effort application that might consume too much bandwidth. A Linux kernel with real-time patches applied allowed us to precisely sample the performance counters at regular intervals (e.g., 100 μs) and duty-cycle the best-effort applications accordingly.

D. Threshold and suspension time

The threshold for the read and write bandwidths depends on three things: available bandwidth of the specific processor, consumed bandwidth of the real-time applications and timing constraints of the real-time application. The available bandwidth of the processor in combination with the consumed bandwidths of the real-time applications determine the available bandwidth for the best-effort applications. The timing constraints (e.g., latency) of the real-time application in combination with the unobstructed performance of the real-time application determine how much performance degradation (e.g., latency increase) the real-time application can handle before a timing constraint is violated. The timing constraints and the performance of the real-time application therefore determine how long a best-effort application should be suspended in relation to the sample interval between the measurements of the performance counters.

 ISBN: 978-1-61208-258-5

VIII. EXPERIMENTS

In this section we describe the experiments that we performed to evaluate the proposed technique. First we describe the system that we used for the experiments and then we introduce the different experiments.

A. Experimental setup

We used a Linux kernel with real-time patches applied. For each core that would be monitored and duty-cycled, a thread with real-time priorities was instantiated and mapped to that core in order to periodically sample the performance counters. When a derived bandwidth crossed a certain threshold the thread would suspend the core for a specified amount of time. Furthermore, the threads of the real-time and best-effort applications were allocated to disjoint sets of cores.

In the experiments we could influence the following parameters: the type of real-time application, type and number of applications of which the best-effort performance should be optimized, sampling interval, bandwidth threshold, suspension time. The experiments were categorized in three classes.

1) Synthetic Bandwidth Experiments: In our first set of experiments we ran the same program that we used to determine the bandwidth between the level 2 and level 3 of the cache. However, now we applied our duty-cycling technique on cores 2 through 4 and measured the bandwidth on core 1. The sampling interval was 100 μs and the suspension time 900 μs. The thresholds for duty-cycling were set at 25% of the maximal bandwidth. The results of this experiments can be found in Table II and relative to Table I in Table III.

2) Optimized Real-Time Streaming Experiments: In the second set of experiments we ran a static streaming applications on the first and second core of the processor and ran a set of best-effort applications on the third and fourth core. The X-ray application was generated and compiled with the tooling as presented in [1] with the techniques that reduces jitter on CMP when executed as a single application. In this case the interference memory allocator and the ordering thread scheduling heuristic were used. With the performance counters we estimated the read and write bandwidths of the real-time application to be between 4 and 9 GB/s, which implies that there is enough read bandwidth left, but that the write bandwidth is almost saturated. The threshold for duty-cycling were set at 10%. For the sampling interval and suspension time two sets of parameters were used: 500 μs and 500 μs; and 100 μs and 900 μs for the sampling interval and suspension time respectively. During the experiment the latency of real-time stream processing were measured in order to determine the interference between the applications.

The applications that were used as best-effort applications:

- X-ray^i: The same application that is running as real-time application, but now as background task. Generated using the interference (optimized) memory allocator. Without cache thrashing most memory accesses would hit the cache and memory bandwidth between the cache and memory is low in comparison to the bandwidth between level 2 and level 3 of the cache.
- X-ray^s: The same application that is running as real-time application, but now as background task. Generated using the simple memory allocator, which does not optimize memory accesses and the cache hit ratio is much lower in comparison with the X-ray^i application. The bandwidth between the cache and main memory is therefore also much higher.
- Syn 'C': An application that reads and writes to the third level of the cache as fast as possible.
- gcc: The gcc compiler while compiling an application.
- ffmpeg: A video transcoder while transcoding a video.

Table IV summarizes the results from this experiment. In the table a row is shown for each set of best-effort applications that were used in the experiment. The first two columns show which best-effort application are run on cores 3 and 4. Each set of best-effort application is run in four different configurations, namely: *baseline* where only the real-time application is run on cores 1 and 2; *congested* where the best-effort applications are executed without our proposed duty-cycle technique; and the two *duty-cycled* configuration where the duty-cycle technique is applied with two sets of parameters as explained before.

3) Unoptimized Real-Time Streaming Experiments: In this third set of experiments we reran the same set of applications as in the second set of experiments but we now ran our streaming application with another memory allocator that does not optimize level 3 cache accesses (i.e. the simple memory heuristic from [1]) and therefore has more accesses to the third level of the cache and to main memory. The results for this set of experiments can be found in Table V.

IX. RESULTS

In Table II the results of the first set of experiments are shown. The results in this table clearly show that the maximal reachable bandwidths for the real-time thread is much higher with the duty-cycling technique enabled then without the technique (see Table I. For example, the bandwidth of configuration E under the write scenario without duty-cycling (i.e., column 10 in Table I) drops from 15.1 GB/s when only one core is writing to 4.2 GB/s when all the cores are writing. When the cores 2 through 4 are duty-cycled and all the cores are writing (i.e., the last cell in column 10 in Table II) the bandwidth only drops to 14.0 GB/s. In this case the degradation reduces 89.9% from 72.1% to 7.2%. Table III summarizes the performance degradation reduction between Table I and Table II.

The second set of experiments were used to estimate the benefits of duty-cycling in a realistic setup. The results for this set of experiments can be found in Table IV.

Although the X-ray^s application does not consume the most bandwidth in the cache hierarchy it degrades (increases) the latency of the real-time the most (i.e., the latency increases from 5867 μs to 13873 μs). The main reason that this application

 ISBN: 978-1-61208-258-5

TABLE II
PARTITIONING EFFECTS IN GB/S WHILE DUTY-CYCLED | A) DOES NOT FIT IN CACHE, NOT PARTITIONED. B) DOES NOT FIT IN CACHE, PARTITIONED. C) FITS IN CACHE, NOT PARTITIONED. D) FITS IN CACHE, PARTITIONED. E) ONLY MEASURED THREAD FITS IN CACHE, PARTITIONED

	Bandwidth (GB/s)														
	Read					Write					Read (50%) / Write (50%)				
Cores	A	B	C	D	E	A	B	C	D	E	A	B	C	D	E
1	11.4	11.5	19.9	19.9	19.8	6.3	6.4	15.0	15.0	15.1	7.3	7.3	18.2	18.2	17.6
2	11.1	11.2	19.9	19.9	19.5	6.1	6.1	14.6	15.0	14.5	7.0	7.1	18.0	18.0	17.0
3	11.0	10.8	19.9	19.9	19.1	6.0	6.0	14.2	14.2	14.1	7.0	6.8	17.7	17.6	16.6
4	10.8	10.9	19.9	19.7	18.7	5.7	5.8	14.1	14.0	14.0	6.9	6.7	17.6	17.2	16.1

TABLE III
PERFORMANCE DEGRADATION REDUCTION (%) | A) DOES NOT FIT IN CACHE, NOT PARTITIONED. B) DOES NOT FIT IN CACHE, PARTITIONED. C) FITS IN CACHE, NOT PARTITIONED. D) FITS IN CACHE, PARTITIONED. E) ONLY MEASURED THREAD FITS IN CACHE, PARTITIONED

	Bandwidth (GB/s)														
	Read					Write					Read (50%) / Write (50%)				
Cores	A	B	C	D	E	A	B	C	D	E	A	B	C	D	E
2	91.3	91.7	100.0	100.0	92.1	93.8	91.7	91.1	100.0	90.8	95.7	100.0	88.9	88.2	89.4
3	88.9	88.1	100.0	100.0	92.6	91.7	92.1	89.6	89.6	90.0	97.4	92.1	92.5	91.0	91.1
4	89.6	91.5	100.0	0.0	90.7	83.8	88.6	90.6	89.6	89.9	94.1	91.1	93.7	89.4	88.5

degrades the latency the most because it is the only tested application in our benchmark set that also uses a lot of cache space and therefore introduces more cache thrashing then the other application. Nevertheless, the duty-cycling technique also decreases the latency degradation for this application.

The X-ray^i and Syn 'C' application use a significantly smaller amount of the cache space and therefore inflict less cache line evictions to the real-time application, but use much more bandwidth between the level 2 and level 3 of the cache. Also for these cases (i.e., row 1 and 3) we see a significant degradation of the latency (i.e., to 9283 μs and 9219 μs for the two applications respectively). In both cases the latency degradation is reduced by the duty-cycling technique.

Gcc and ffmpeg (i.e. row 4 and 5) both exhibit much lower bandwidth usage to the third level of the cache and therefore do not degrade the latency to the same extend as the other cases.

Both sets of parameters (500/500 and 100/900 for the sampling interval and suspension time) reduced the degradation of the latency of the real-time application. The 100/900 parameters reduced the latency degradation more than the 500/500 parameters, but also results in less best-effort performance.

In the last set of experiments we see that the latency of the interfered real-time application is higher than in the second set, this is due to the fact that the unoptimized X-ray application consumes more bandwidth to the third level of the cache and to main memory is therefore more susceptible to interference. However, relative to the baseline latency of the real-time application the increase is slightly smaller. Furthermore, we observe that also in this case the latency increase could be controlled by the duty-cycling technique.

X. FUTURE WORK

The evaluated duty-cycling technique uses periodic sampling in order to estimate bandwidth usage. Furthermore, it currently uses a simple control mechanism in order to reduce bandwidth congestion. Two simple improvements that can decrease overhead and improve best-effort performance can use so-called event based bandwidth estimation and proportional suspend times.

The event based bandwidth estimation can be implemented by allowing the performance counters to make an interrupt request when a certain threshold is exceeded. This reduces the amount of times that the bandwidth is estimated in applications with low bandwidth usage.

The second improvement could be proportional suspend times instead of fixed suspend times. The amount of times the best-effort application has exceeded bandwidth usage and by how much could be used to determine the suspend time. This allows the duty-cycling technique finer control over when the best-effort should be duty-cycled and thereby increasing best-effort performance without violating real-time constraints.

XI. CONCLUSION

Best-effort applications can cause | due to bandwidth congestion in the cache hierarchy | an unacceptable amount of interference, which can results in an unacceptable latency increase of real-time applications on CMP architectures. This can make it infeasible to execute best-effort applications concurrently with real-time applications. We therefore proposed a duty-cycling technique that can throttle (duty-cycle) best-effort applications when their bandwidth consumption causes too much interference with the real-time applications. The duty-cycling technique is implemented in the Linux operating system

 ISBN: 978-1-61208-258-5

TABLE IV
DUTY-CYCLING RESULTS | LATENCY OF THE REAL-TIME APPLICATION

		Baseline			*Congested*			Duty-cycled 500/500			Duty-cycled 100/900		
Core 3	Core 4	Avg (μs)	Stdev	Max (μs)	Avg (μs)	Stdev	Max (μs)	Avg (μs)	Stdev	Max (μs)	Avg (μs)	Stdev	Max (μs)
X-ray^i	X-ray^i	5867	15.22	6069	9283	19.68	9470	7339	156.93	8849	6136	88.55	7037
X-ray^s	X-ray^s	5867	15.22	6069	13873	87.85	14185	8305	277.14	9474	6304	66.3	6512
Syn 'C'	Syn 'C'	5867	15.22	6069	9219	48.43	10644	7370	90.93	7912	6037	46.87	7225
Syn 'C'	gcc	5867	15.22	6069	7691	161.95	9327	6755	95.48	7256	6050	41.68	6978
ffmpeg	ffmpeg	5667	15.22	6069	6131	149.37	7855	5977	94.55	7211	5881	22.62	6010

TABLE V
DUTY-CYCLING RESULTS - NOT OPTIMIZED FOR LEVEL 3 ACCESS - LATENCY OF THE REAL-TIME APPLICATION

		Baseline			*Congested*			Duty-cycled 500/500			Duty-cycled 100/900		
Core 3	Core 4	Avg (μs)	Stdev	Max (μs)	Avg (μs)	Stdev	Max (μs)	Avg (μs)	Stdev	Max (μs)	Avg (μs)	Stdev	Max (μs)
X-ray^i	X-ray^i	10705	28.56	10918	14458	211.81	16146	13092	248.73	14704	11295	70.89	12551
X-ray^s	X-ray^s	10705	28.56	10918	21236	62.56	21513	14237	69.32	14506	11315	69.58	11670
Syn 'C'	Syn 'C'	10705	28.56	10918	12742	18.85	12805	12193	39.06	12336	11540	46.48	11754
Syn 'C'	gcc	10705	28.56	10918	11990	187.30	12773	11800	98.88	12197	11767	144.83	12368
ffmpeg	ffmpeg	10705	28.56	10918	11554	259.21	12651	11300	159.26	12260	11059	50.94	11467

and uses performance counters available in the cores of the CMP to estimate the bandwidth usage of the best-effort applications. From our experimental results we conclude that the proposed duty-cycling technique can significantly reduce the interference. For an industrial X-ray imaging application we show that the latency increase that is caused by the interference can be controlled while maintaining some best-effort performance. For synthetic benchmarks we show that there are scenarios where we the duty-cycling technique can reduce the latency degradation by 89%. We therefore conclude that best-effort core duty-cycling based on bandwidth consumption is an essential method to control the interference in CMP architectures used for real-time streaming applications in combination with best-effort applications, such as the interventional X-ray application.

XII. ACKNOWLEDGMENTS

The authors would like to thank Philips Healthcare, and in particular Marc Schrijver for their help during this research. This research is supported by the Dutch Technology Foundation STW, which is part of the Netherlands Organisation for Scientific Research (NWO), and which is partly funded by the Ministry of Economic Affairs.

REFERENCES

[1] M. Westmijze, M. Bekooij, G. Smit, and M. Schrijver, "Evaluation of scheduling heuristics for jitter reduction of real-time streaming applications on multi-core general purpose hardware," pp. 140 –146, Oct. 2011.

[2] H. Stone, J. Turek, and J. Wolf, "Optimal partitioning of cache memory," *Computers, IEEE Transactions on*, vol. 41, no. 9, pp. 1054 –1068, Sept. 1992.

[3] D. Chiou, D. Chiouy, L. Rudolph, L. Rudolphy, S. Devadas, S. Devadasy, B. S. Ang, and B. S. Angz, "Dynamic cache partitioning via columnization," 2000.

[4] R. Iyer, "CQoS: a framework for enabling qos in shared caches of cmp platforms," in *Proceedings of the 18th annual international conference on Supercomputing*, ser. ICS '04. New York, NY, USA: ACM, 2004, pp. 257–266.

[5] H. Kannan, F. Guo, L. Zhao, R. Illikkal, R. Iyer, D. Newell, Y. Solihin, and C. Kozyrakis, "From chaos to qos: case studies in cmp resource management," *SIGARCH Comput. Archit. News*, vol. 35, no. 1, pp. 21–30, Mar. 2007.

[6] S. Kim, D. Chandra, and Y. Solihin, "Fair cache sharing and partitioning in a chip multiprocessor architecture," in *Proceedings of the 13th International Conference on Parallel Architectures and Compilation Techniques*, ser. PACT '04. Washington, DC, USA: IEEE Computer Society, 2004, pp. 111–122.

[7] B. Verghese, A. Gupta, and M. Rosenblum, "Performance isolation: sharing and isolation in shared-memory multiprocessors," in *Proceedings of the eighth international conference on Architectural support for programming languages and operating systems*, ser. ASPLOS-VIII. New York, NY, USA: ACM, 1998, pp. 181–192.

[8] K. Nesbit, M. Moreto, F. Cazorla, A. Ramirez, M. Valero, and J. Smith, "Multicore resource management," *Micro, IEEE*, vol. 28, no. 3, pp. 6 –16, May-Jun. 2008.

[9] A. Hansson, K. Goossens, M. Bekooij, and J. Huisken, "Compsoc: A template for composable and predictable multi-processor system on chips," *ACM Trans. Des. Autom. Electron. Syst.*, vol. 14, no. 1, pp. 2:1–2:24, Jan. 2009.

[10] D. Molka, D. Hackenberg, R. Schone, and M. Muller, "Memory performance and cache coherency effects on an intel nehalem multiprocessor system," in *Parallel Architectures and Compilation Techniques, 2009. PACT '09. 18th International Conference on*, Sept. 2009, pp. 261 –270.

 ISBN: 978-1-61208-258-5

E-Learning Portal Architecture – From Web Services towards Cloud Services

Alina Andreica*, Florina Covaci*, Daniel Stuparu*, Gabriel Pop*, Călin Miu*, Romulus Gadi**, Flavius Chira** , Grigorie Bogdan Mărcuş** , Cosmin Tarţa** , Cristina Câmpeanu**
IT Department, Babes-Bolyai University, Cluj-Napoca, Romania *
{alina.andreica, florina.covaci, daniel.stuparu, gabriel.pop, calin.miu}@ubbcluj.ro *
Net Brinel SA, Cluj-Napoca, Romania **
{romulus.gadi, flavius.chira, grigorie.marcus, cosmin.tarta, cristina.campeanu}@brinel.ro **

***Abstract*— The present paper focuses on architecture principles for providing integrated e-learning services both as web services and as cloud services, technology that presently emerges as 'on-demand' services. The information system facilities are made available by integrating into a global web portal the dedicated services and synchronizing databases based on various technologies. The web portal architecture that currently comprises e-learning and dedicated information systems facilities is extended with cloud on-demand facilities. The classical portal architecture, based on MS technology, provides, as learning services, management content and e-learning facilities for various user categories, as well as dedicated information system facilities. We design the extension of the this architecture with on-demand services in the cloud using Office 365, which provides cloud services for e-mail, office applications and collaboration tools and we envision the impact of the new architecture. The described services, which are available on-demand, partly hosted in the cloud, create an "intensive" focused architecture that may be applied for various organization cases.**

Keywords- system integration; database synchronization; e-learning; on-demand & cloud services; web portal.

I. INTRODUCTION AND WORKING FRAMEWORK

Information system integration is a very important issue for present organizations and has been tackled in the literature especially for business and organizational processes [12]. System interoperability has also been dealt with from a semantic point of view [13].

System integration is best solved within a single sign-on framework. The problem of managing uniform user identities in organizations is addressed by Shaw [16] within a management access system for a framework with different user identities in specific information systems (Quest OneIdentity Solution). The solution relies in building an unified identity and access management system – IAM within the organization, consolidating multiple identities in one (integrated) identity. The IAM approach [16] simplifies identity access and management within the organization, improving security and productivity, and providing single sign on, role management, multiple authentication and password management facilities. IAM complexity may be managed with [16]: (1) point solutions, which implement a password reset within a system and then synchronize it with the others, or (2) IAM frameworks, which implement a specific IAM solution based on already developed frameworks: IBM (Tivoli Identity Manager), Oracle, Novell, MS Forefront Identity Manager FIM (Identity Lifecyle Manager - ILM). Our paper uses the latter approach. The FIM / ILM server provides automated synchronization mechanisms that enable the integration of Active directory with other services, like SQL, e-mail (OWA), chat, etc.

Universities also adopt information systems and integration solutions based on Oracle technologies [32]. Our option also takes into consideration the existing MS Academic Agreement from an IT strategy point of view [3].

In [6], [7], [8] we present the architecture of an e-learning portal for providing e-learning and dedicated information system services. The architecture, based on Microsoft technology, uses a FIM – Forefront Identity Manager [26] server, and additional interface modules, for integrating the dedicated information systems into the web portal that also provides e-learning facilities, based on SharePoint Portal functionalities. The proposed framework enables data and service integration that may be further exchanged within federated web services – see also [11], [20].

The paper describes the extension of this architecture with on-demand services hosted in the cloud. In this respect, we are going to use Office 365 cloud services for e-mail, office applications and collaboration tools.

Section 2 describes the present architecture integration principles. In Section 3, we present the web services that are provided within the portal, while Section 4 focuses on the new cloud services and on-demand facilities. Conclusions reveal the most important contributions of the paper.

II. SYSTEM INTEGRATION ARCHITECTURE

The integrated e-learning portal provides academic web-services: e-learning and dedicated information system ones. The portal integration with the dedicated information systems (AcademicInfo, ManageAsist, Research Management System in our case; see Section 3), is described further. The architecture we propose (see Fig. 1) is based on MS technology; we use an ILM – Identity Lifecycle Management – type server ensuring single sign-on capabilities and uniform interface to the dedicated information systems. Therefore, these dedicated facilities

 ISBN: 978-1-61208-258-5

will be mapped into the portal, together with supplemental web services – see Section 4.

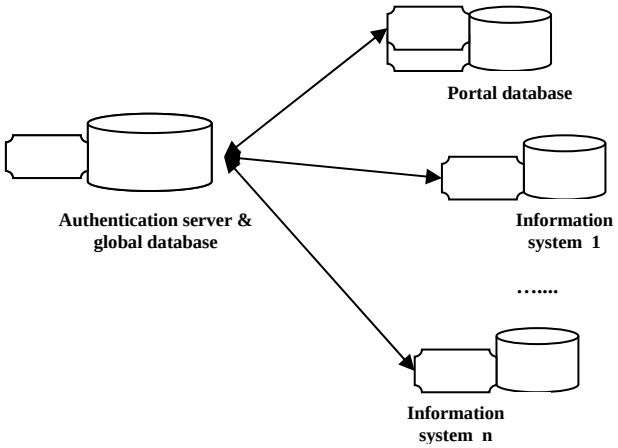

Figure 1. System integration architecture.

In order to access the dedicated information system facilities, we selected the user categories that have specific functionalities and designed, for each, an "access point" into the portal. Authentication is performed by the same account and password into the portal and into the information systems, i.e. credentials are the same for the portal and the accessed information systems. After portal login and credentials check, the logon information is retained as current session variables and further passed on towards the information systems; access points are designed in respect with the account permissions (in our case, user categories are students, academic staff and /or research staff, managers, secretariats). Each user category may have one or more roles, based on which we define access points into the dedicated information systems. Credentials are passed to the accessed information system as session variables and used in order to also perform authentication into the dedicated systems.

Permissions of each user category are retained in a global database, created by synchronizing the information systems' databases with necessary information [7], [8]. In our case:

- *students* access their educational path (AcademicInfo system, see Section 3) and learning resources in view permissions (portal);
- *academic staff* access disciplines they teach and student grades (AcademicInfo [21]) and learning resources in design permissions, available facilities in the Research management system (activity upload, syntheses) and administrative facilities in ManageAsist system;
- *research staff* access their facilities in the Research management system (activity upload, syntheses) and administrative facilities in ManageAsist system;
- *managers* access educational syntheses (AcademicInfo), research and financial syntheses for the unit they manage (department, faculty, institute, university) - Research management and ManageAsist systems [9].

A. Database synchronization principles

Single sign-on correctness is dependent on the permission information, consequently on the database consistency; therefore, database synchronization is required. The synchronization process will include the information systems' databases and the portal database (Fig. 2). Database synchronization ensures global database consistency and updates access permissions both for the portal and the information systems; credentials logged in the portal will therefore be verifiable, within the global database, in respect with their permissions in the dedicated information systems.

The global synchronized database is based on the human resource & organization chart information, retained in dedicated tables, as described in [6]:

```
User[userid, account, password, unitid]
Unit[unitid, unitname, …]
Organization_chart [unitid, superior_id, horiz_id]
```

This common database, used by the ILM server, also contains user and group authentication information, together with dedicated permissions in each of the information systems, in order to ensure access to corresponding permissions into the dedicated information systems. Permissions within the portal are implemented by means of dedicated groups. Active directory AD is provisioned with necessary information from the global database, using ILM as a synchronization interface (see Section 3).

Security issues are managed according to AD and ILM principles – see [25], [26]. Data privacy aspects are treated according to the legal regulation principles (for example student grades, teacher wages), and accessible only to their owners and may be processed within the dedicated systems according to organizational workflows.

The synchronization process is based on the dataflow sequences that occur in information processing; these sequences have to be defined in each organization case. In our case, the data workflow involves (Fig. 2, left hand side):

- 1. organization chart and user tables are replicated from ManageAsist into the global database;
- 2-3. these tables are afterwards sent into Research Management and AcademicInfo databases;
- 4. grant tables are replicated from Research Management into the global database;
- 5. these tables are then sent into ManageAsist database;
- 6. students, disciplines, educational data, fee tables - transferred from AcademicInfo into the global database;
- 7. these tables are sent into ManageAsist database [9]

The synchronization engine follows the above described steps 1-7; it searches Update, Insert and Delete operations and sends the corresponding information into the global database. Potential error messages are retained in a dedicated table from the global database. The synchronization process runs daily. First run time was around 7 hours, while current updates daily vary from 7 seconds to about 30 minutes, according to the updates that are performed on the databases. For an efficiency evaluation, current database dimensions are: AcademicInfo: 9.75GB (with 1.48GB data, rest – logs), ManageAsist: 4.52G, Research Management: 132M, UBBonline global database: 50.75GB (49.1GB, rest logs). A full permission restoration procedure, which sets single sign on permissions for ILM server, based on data provided by the dedicated systems (human resources, students, contract studies, etc.) runs daily, taking around 5 hours. Any potential errors are automatically logged and later analyzed; even if an

 ISBN: 978-1-61208-258-5

error occurs, the permission setting procedure is fully resumed the next day, so previous errors do not influence it, unless they occur consequent to some structural errors, which can be identified from the logs and solved.

The latter process is performed via the ILM server (MS SQL web) between the global database GDB and Active directory AD information (BDG – UBBonline portal link in Fig. 2). We use ILM's connector space (MS SQL web) and two dedicated agents which transfer information from the global database, respectively from AD into the connector space [9], compare it and synchronize the updated information from GDB into AD, updating therefore the necessary information in order to apply correct permissions within the portal access (ILM – OUs – UBBonline portal links in Fig. 2). The ILM server also performs the integration with AD and the SQL server managing UBBonline database, the e-mail server – MS Exchange and the communication server (see Fig. 2, lower right hand side).

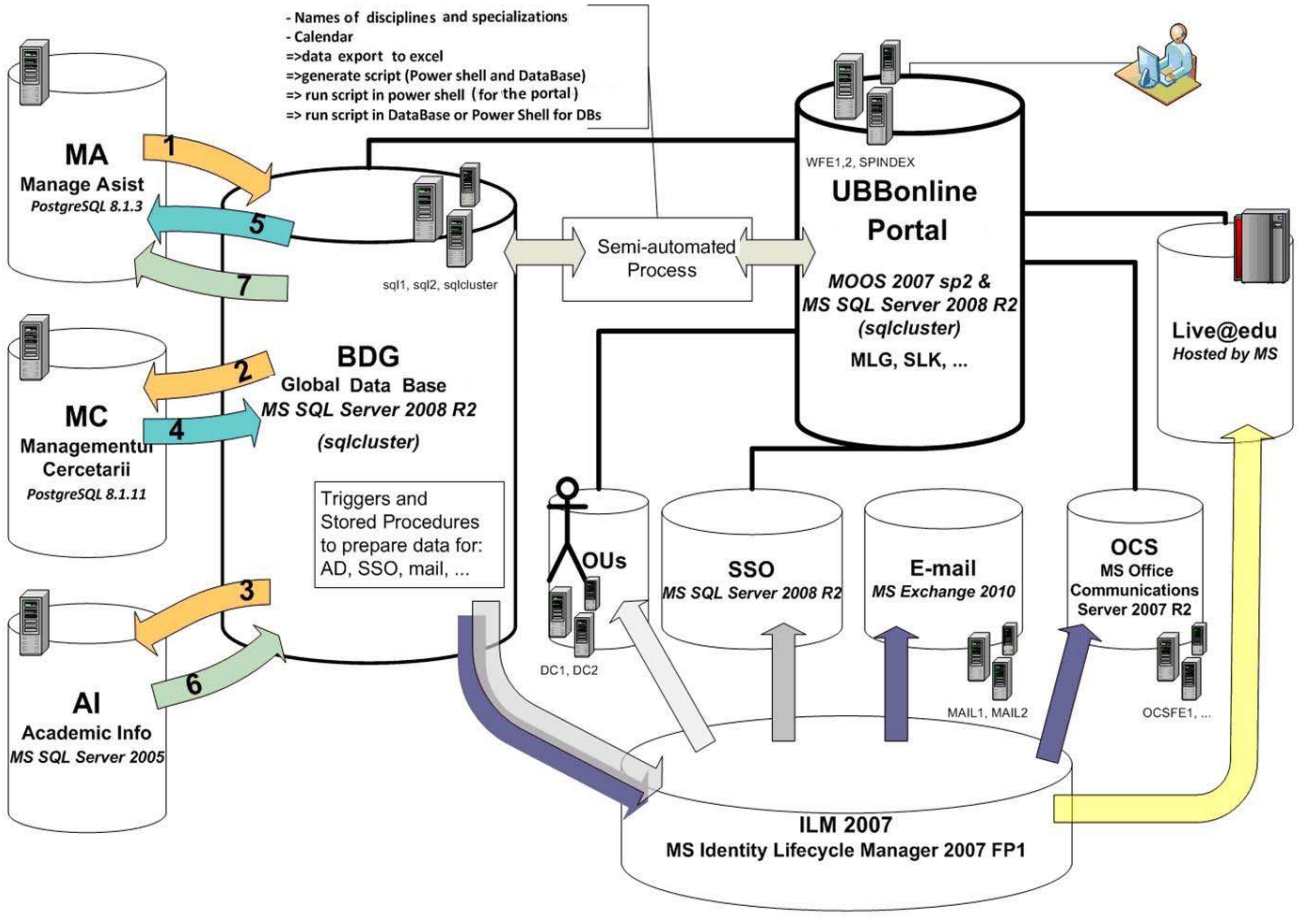

Figure 2. Portal global architecture.

III. THE WEB SERVICES

We further describe the web services provided by UBBonline e-learning portal [24].

A. *E-learning functionalities and virtual labs*

The implementation of e-learning facilities strongly contributed to the development of the student and goal centered learning model [1]. E-learning facilities are usually provided by means of web services.

The web-based e-learning facilities provided by the described portal are the SharePoint [18] built in ones, adapted to our specific needs, and include:

- ◊ content management and sharing (see Fig. 3);
- ◊ schedule management and sharing,
- ◊ communication facilities (e-mail – OWA type, discussion lists, etc.) ,
- ◊ evaluation tools and feed-back facilities;
- ◊ task management, blog and RSS tools,
- ◊ survey tools, as well as other functionalities.

The system is also open to adding new web-parts, services or components (for example, evaluation ones are being developed). **Virtual lab services** enable modelling of processes that may be tedious to be accessed in real conditions, or are required to be accessed remotely. These facilities support learning in experimental sciences and sharing experimental knowledge by electronic means in: process engineering, environmental engineering, physics,

chemistry, biology, etc. Our virtual labs facilities refer to: on-line virtual experiments and on-line labs; case studies based on mathematical modelling and simulation; recorded video sequences and on-line video streaming; posting material to be further processed with dedicated clients. Besides facilities for publishing virtual lab educational content within the discipline's media library, the portal includes tools for managing virtual lab resources, modelled as a workflow for requesting, approving and allocating specific resources, like videoconferencing facilities, virtual machines, specific experimental equipments, etc.

Figure 3. Managing an educational resource (in design permissions).

B. *Dedicated services provided by the information systems*

AcademicInfo [21] is an integrated information system dedicated to managing educational information, with facilities for secretariats, specific access for students and teachers and relevant syntheses on the educational process. The system models educational processes at BBU level, ensuring course selection in all faculties' curricula, models in a flexible manner various types of educational activities (BA, MA, PhD, continuous), ensures multilingual support in processing &reporting. Specific web services are dedicated to: *students, teachers, academic management* – see also [7].

ManageAsist system is the integrated software system for administrative management that has been developed for our university. The system can be viewed as an ERP (Enterprise Resource Planning) system; in its design and implementation, we integrated systematic efficiency principles in software design [4]. ManageAsist's principles and facilities are adapted to high education institutions, containing: Document management, Assets, Warehouse, Cashier, Finance, Accountancy, Grants, Human Resources and Acquisitions modules, decision assistance facilities [4]. Each module contains management reports for the corresponding compartment. Relevant synthesis from each compartment will be integrated, together with global management tools into a decision support module. The web services [22] include access to grant financial information and management of acquisition request, including specific reporting facilities for management levels.

Our **Research Management System** [23] is a web based system that we have developed and implemented within Babes-Bolyai University (BBU) in order to manage research activities. The system offers, via web interfaces, accessible and user-friendly means of collecting specific information, and automatically performing quantitative analyses, syntheses and evaluations based on the collected information. The system is a tool for quantitative research evaluation, ensuring proficient management of BBU research activity and supporting competitive strategies in the field [2].

C. *Authentication Characteristics*

The portal provides single sign-on [29] facilities using a MS ILM - Identity Lifecycle Management Server [26] and ISA Server; the authentication is based on Active Directory facilities [6]. While OpenAuth protocol [27] grants access without sharing passwords, the architecture we describe uses the ILM facilities for synchronizing authentication information (User, password), according to [26]. This credential exchange is similar to the one used by OpenID protocol [28], but is performed by means of the ILM built in

 ISBN: 978-1-61208-258-5

facilities, according to [26]. The authentication mechanism also implements the MS Domain Trust policy [25].

We consider that the architecture based on ILM server [26] has good implementation advantages, since it already provides built-in web authentication facilities [6].

D. Extensibility principles

The solution we propose, based on MS technology and ILM authentication server, may be applied in various cases that require information system integration. Moreover, such an architecture may be enhanced with SharePoint facilities in order to provide sharing, communication and e-learning functionalities, which are often necessary within organizations. Considering we have available n information systems, the software services that can be provided into the web portal are represented in Fig. 4.

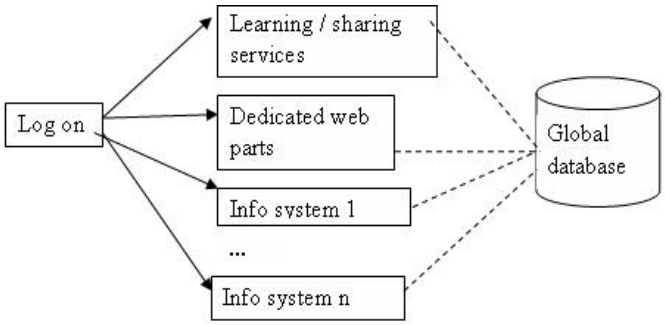

Figure 4. Web portal services.

IV. SERVICES ON-DEMAND

Within the trend of providing proficient on-demand services, we have designed an extended portal architecture in order to include on-demand cloud services. This architecture is based on the integration of MS Office 365 [31], which offers cloud services solution for mail, office applications and collaboration tools. Presently, Live@edu's facilities (initially used for student accounts) have been migrated into Office 365. Office 365 integrates various product facilities within one, on-line tool [31]: office applications streamed from the cloud in real time, access to e-mail and calendar from any location (personal computer, web access, mobile or smart device), professional web sites facilities, file sharing and project management facilities, instant messaging, presence and conferencing, mobility, security and trust.

Authentication integration and single sign on are performed based on the principles used in the initial architecture – see Section 2, by using Forefront Identity Manager and Exchange web functionalities are used for hosted e-mail accounts. E-mail migration is performed using the dedicated tools both for e-mail accounts [33] and office features [34] , and pursuing the specific stages.

Cloud services have important on-demand availability characteristics, on an externally hosted infrastructure, and convenient costs (for example, in the case of student e-mail and file accounts, an owner hardware infrastructure would be more expensive). The use of such facilities may be extended with Office on-line functionalities.

Universities' usage of cloud services, with important benefits cost/proficiency benefits for large numbers of students are to be taken into consideration. E-mail, file space and Office facilities are relevant in this respect. The experience, including the information system integration principles proposed in Section 3D, may be transferred to other organizations with learning goals or with large numbers of employees, especially in the case of initially scarce infrastructure investments.

After fully implementing these facilities, we plan to compare access time and workload for the on-site and in-cloud scenarios and to perform a complete cost-benefit analysis.

V. CONCLUSION AND FUTURE WORK

The paper describes the evolution of portal architectures from web services to cloud services. The current portal architecture provides system integration facilities, systems being integrated, together with sharing and communication facilities into a global portal.

Our case study is performed on an academic institution; the universities' case is quite complex, since their activity covers: education, research, administration. The system framework integrates various web services within the portal: e-learning facilities, virtual labs and dedicated information facilities. We describe the integration solution of web services into the portal, including web-based facilities from the dedicated information systems. The solution is based on MS technology and provides means of integrating various information systems by implementing a single authentication server and mapping specific facilities from the dedicated information systems, for each user category. This architecture is based on a global integrated database and a permission mapping scheme for ensuring appropriate access into the dedicated information systems.

Synchronization processes from the information system databases into the global database run daily, taking up to 30 minutes or less, depending on the database updates, for a ~50GB global database. Single sign-on permissions are also updated daily based on data provisioned by the dedicated systems. This web service integration solution has a good extensibility degree and may be applied in various organizational cases, aiming at providing integrated services, by building a single log-on framework for learning, collaboration facilities, and integrating dedicated information system services.

On a second development stage, we have extended the portal architecture with on-demand cloud services taking into account their high availability and convenient costs. Therefore, students use Office 365 facilities for e-mail and file space facilities. This solution has important advantages regarding necessary storage space, which is externalized, as well as the necessary software services, which are provided in the cloud. After fully implementing these facilities, we plan to compare access time and workload for the on-site and in-cloud scenarios.

The advantages of the proposed solution rely in providing a uniform web framework for: database synchronization of various information systems databases

 ISBN: 978-1-61208-258-5

and web access to e-learning and information collaboration & sharing tools and dedicated system facilities. The proposed framework enables data and service integration – both web and cloud – that are available at with single sign-on. This integration solution has a good extensibility degree and may be applied in various cases.

ACKNOWLEDGMENT

The work described in this paper was supported during 2009-2011 by the EU funded grant, within the European Fund for Regional Development, "CCE 124/323/31.08.2009 SMIS 4424 - Sistem electronic aplicativ integrat de educaţie al Universităţii Babes-Bolyai" – Integrated applied electronic system for education of Babes-Bolyai University - BBU, contracted by BBU with the Romanian Ministry of Communication and Information Society, Organismul Intermediar pentru Promovarea Societăţii Informaţionale (the Intermediary Structure for Promoting the Information Society), during 31-08-2009 – 31-08-2011

We thank to the whole development team in our IT department for their contribution to developing ManageAsist, AcademicInfo, Research management information systems and to administering the e-learning portal: F. Tufiş, C. Miu, S. Nemeş, D. Pop, M. Bojan, C. Pavel, A. Iuhos, A. Bara, Kerekes H., Zölde A., Kerekes T.

REFERENCES

[1] M. Allen, Guide to e-Learning. Wiley, 2002

[2] A. B. Andreica and P. S. Agachi, "Design and Implementation of An Integrated Software System for Managing Research Activities in Universities", 7th RoEduNet International Conference - Networking for Research and Education, UT Press, Ed: E. Cebuc, Cluj-Napoca, Romania, pp. 90-95, 2008

[3] A. B. Andreica, IT Management, EFES, Cluj-Napoca, 2009

[4] A. B. Andreica, D. Stuparu, and F. Ghetie, "Design and Implementation of an Erp System for Universities", Proceedings of Information Systems 2009, IADIS, Barcelona, Spain, Eds: M. Nunes, P. Isaias, P. Powell, pp. 315-322, 2009

[5] A. B. Andreica, F. Covaci, D Stuparu, and G. Pop, "An E-Learning Web Portal with System Integration Facilities", Web Information Systems and Technologies 2010, Valencia, Spain, Proceedings of 6th International Conference WEBIST, I, INSTICC, Ed: J. Filipe, J. Cordeiro, 2010, pp. 131-136

[6] A. B.Andreica, F. Ghetie, D. Stuparu, I. Arpad, and G. Pop, "Integrated E-Learning Web Services", 4th International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies, Florence, Italy, UBICOMM 2010 Proceedings CD, IARIA, Ed: J. Lloret Mauri, S. Baladin, C. Dini, 2010, pp. 0-5

[7] A. B. Andreica, D. Stuparu, F. Ghetie, and G. Pop, "An Integrated Software Solution for E-learning and Academic Services", International Technology, Education and Development Conference 2011, Valencia, Spain, 7-9 March 2011, IATED Proceedings CD , IATED, 2011, pp. 1-8

[8] A. B. Andreica, D. Stuparu, R. Gadi, F. Covaci, C. Tarta, G. Marcus, G. Pop, and O. Teodorescu, "Design and Implementaiton of An Integrated Web Service Architecture", Web Information Systems and Technologies - Web Services Principles and Applications, Noordwijkerhout, Holland, 6-9 May 2011, Proceedings of WEBIST 2011, Ed: J. Filipe, Jose Cordeiro, pp. 631-637

[9] A. B. Andreica, D. Stuparu, F. Covaci, C. Miu, G. Pop, Gadi R., F. Chira, G. Marcus, C. Tarta, and C. Campeanu, "An Integrated Web Architecture for Providing Academic Services", IADIS International Conference E-learning 2012, IADIS MCSIS 2012, Lisbon, Portugal, Ed: M. Batista Nunes, M. McPherson, 2012, pp. 449-454

[10] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, Design Patterns, Teora, 2002

[11] M. T. Goodrich, R. Tamassia, D. Yao, "Notarized Federated Identity Management for Web Services" http://www.cs.brown.edu/cgc/stms/papers/notarizedFIM.pdf , accessed July 2011

[12] W.Hasselbring, "Information System Integration". Communications of the ACM, 43 (6) , 2000. pp. 32-36

[13] W. Hasselbring, S. Pedersen, "Metamodelling of Domain-Specific Standards for Semantic Interoperability". Lecture Notes in Computer Science, 3782, 2005, pp. 557 – 559

[14] W. Horton and K. Horton, E-learning Tools and Technologies: A consumer's guide for trainers, teachers, educators, and instructional designers, Wiley, 2003

[15] D. Stuparu, A Andreica, and I. Mantu, "Comparing Access Techniques on Databases in Distributed Application Frameworks", Proc. of Collaborative Support Systems in Business and Education, BBU, Cluj-Napoca, 2005, pp. 1-10

[16] J. Shaw, "Unified and Intelligent Identity and Access Management", Quest Software, White Paper, 2011

[17] R. Webster and F. Sudweeks, "Teaching for e-Learning in the Knowledge Society: Promoting Conceptual Change, in Academics' Approaches to Teaching", Current Developments in Technology-Assisted Education, 2006, http://www.formatex.org/micte2006/pdf/631-635.pdf [retrieved: 7, 2011]

[18] MS Learning Gateway and SharePoint Portal, http://www.microsoft.com/education/solutions/higheredportals.aspx [retrieved: 6, 2011]

[19] PostgreSQL Team, "High Availability, Load Balancing, and Replication", http://www.postgresql.org/docs/8.3/static/high-availability.html [retrieved: 7, 2011]

[20] Business Explorer for Web Services, http://www.alphaworks.ibm.com/tech/be4ws , accessed July 2010

[21] BBU AcademicInfo System, http://academicinfo.ubbcluj.ro/Info [retrieved: 7, 2011]

[22] BBU ManageAsist System, http://manageasist.ubbcluj.ro [retrieved: 7, 2011]

[23] BBU Research Management System, http://infocercetare.ubbcluj.ro , [retrieved: 7, 2011]

[24] BBU E-learning portal, https://portal.portalid.ubbcluj.ro [retrieved: 7, 2011]

[25] Federated Identity Patterns in a Service-Oriented World, http://msdn.microsoft.com/en-us/architecture/cc836393.aspx [retrieved: 7, 2011]

[26] Identity Lifecycle Management Server, http://www.microsoft.com/windowsserver2003/technologies/idm/ilm.mspx [retrieved: 7, 2011]

[27] OAuth protocol, http://oauth.net/ [retrieved: 7, 2011]

[28] OpenID protocol, http://openid.net/ [retrieved: 7, 2011]

[29] SingleSignOn, http://www.authenticationworld.com/ [retrieved: 7, 2011]

[30] Web Services Federation Language, Dec 2006, BEA Systems, IBM Corporation, Layer 7 Technologies, http://www.ibm.com/developerworks/library/specification/ws-fed/ [retrieved: 7, 2011]

[31] MS Office 365, http://www.microsoft.com/en-us/office365/what-is-office365.aspx [retrieved: 11, 2012]

[32] Oracle Users Group, http://education.oracle.com/web_prod-plq-dad/plsql/ou_usergroups.ou_ioug_home [retrieved: 12, 2012]

[33] MS Exchange Deployment to Office 365, http://help.outlook.com/en-us/exchangelabshelp/ff633682 [retrieved: 12, 2012]

[34] MS Office 365 Migration, http://www.microsoft.com/government/en-us/products/office/365/how_to/Pages/default.aspx [retrieved: 12, 2012]

 ISBN: 978-1-61208-258-5

Design of Parallel Architectures of Classifiers Suitable for Intensive Data Processing

Bogusław Cyganek
AGH University of Science and Technology
Department of Electronics
Krakow, Poland
cyganek@agh.edu.pl

Kazimierz Wiatr
AGH University of Science and Technology
Academic Computer Center Cyfronet
Krakow, Poland
wiatr@agh.edu.pl

***Abstract*—Processing of visual data, such as object recognition and image segmentation, is based on data classification. In this paper architectures of ensembles of classifiers are discussed which show superior accuracy in respect to a single classifier. To achieve comparable response time the parallel computer architectures need to be considered, however. In the paper we present a parallel implementation on a graphic card of an ensemble of one-class support vector machines for image segmentation. We show that the parallel architecture of the ensemble of classifiers allows both, the high accuracy and speed up factor of two orders of magnitude compared with the serial software implementation.**

Keywords-ensemble of classifiers; OC-SVM; data processing; GPU implementation; image segmentation

I. INTRODUCTION

Data processing highly relies on classification methods. The two key parameters of the classifiers are their accuracy and response time. However, a choice of a right classifier to a given task depends on a number of factors, such as a number of training data, their type as well as data dimensionality, and frequently is a matter of the skills and experience of a system designer [7][15][18]. The choice of the right classifiers is especially important in the time critical and massive data processing systems. To this group belong vision processing systems addressed in this paper. They are frequently used in such tasks as face and gesture recognition, public area surveillance, driving assistance, and many more.

In this paper implementation issues of different architectures of ensembles of classifiers employed to the tasks of image processing are discussed. Such ensembles show superior accuracy compared to a single classifier, as reported in the literature [3]-[5][7]-[11][19]. However, operation of many classifiers leads to excessive response time of such systems if implemented in serial software. A solution to this problem is parallel operation of the member classifiers which is possible and effective if their operations are independent as much as possible.

In this paper a number of parallel architectures of classifiers is discussed which allow functional and data decomposition of the classification problem and which aim at full utilization of the multi-core processors, graphic cards (GPU), as well as field programmable gate arrays (FPGA) [20]. Our discussion is exemplified with the ensemble of one-class support vector machines (OC-SVM) applied to image segmentation, which is based on our previous work [3][4]. In this paper, we present its parallel implementation on the graphic card which allows real-time operation. We also show that simple application of the data splitter and then a group of cooperating classifiers usually leads to a better performance compared to a single classifier trained with all available training data.

The paper is organized as follows: In Section II, details of the architecture of ensembles of classifiers are discussed. In Section III, details of the data splitter and base classifiers are provided. In Section IV, details of the parallel implementation are discussed. Finally, the paper ends with experimental results discussed in Section V, as well as with conclusions presented in Section VI.

II. ARCHITECTURE OF ENSEMBLES OF CLASSIFIERS

In this section, we discuss two of the most popular architectures of ensembles of classifiers: the serial and the parallel ones. Architecture of the serial cascade of classifiers is shown in Fig. 1. Input data is processed in the *pipeline* like structure. Each classifier filters out data, passing the positively classified ones to the next one in the chain, and so on.

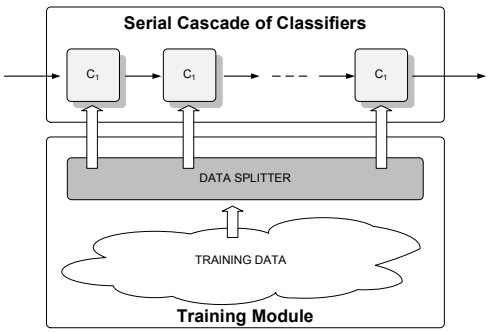

Figure 1. Architecture of the serial cascade of classifiers. Input data is processed in the pipeline structure. Each classifier filters out data, passing the positivly classified ones to the next in the chain, and so on. Training is done with the AdaBoost to amplify response on poorly classified examples.

An example of such a system is the face detection method by Viola and Jones [19]. Training is done with the

 ISBN: 978-1-61208-258-5

AdaBoost which amplifies response on poorly classified examples. Such strategy imposes data decomposition into sets of usually decreasing number of elements [9].

Data processing in a serial chain of classifiers is effective if member classifiers are able to operate in a pipeline mode. One of the requirements in this case is that each classifier in the chain consumes the same time quant for data processing. The penalty of using a cascade is a delay necessary to fill up the processing line which is proportional to the number of used classifiers. However, in practice these requirements are not easy to fulfill.

On the other hand, architecture with a parallel ensemble of classifiers is depicted in Fig. 2. In this case all classifiers are assumed to operate independently which is a big advantage considering implementation and execution time. However, all partial responses need to be synchronized and collected by the answer fusion module which outputs a final response. There are different methods of training of the member classifiers C_i, shown in Fig. 2, as is discussed in many publications [7]. Some of the most popular are data bagging and data clustering, which will be discussed in the next sections.

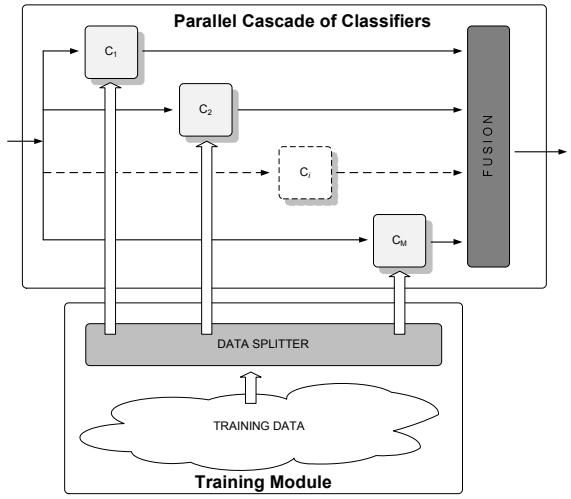

Figure 2. Parallel cascade of classifiers. Input data is split and fed to all classifiers in the ensemble. A final response is provided from the fusion module. To obtain diversity of the ensemble either different classifiers are used and/or different data partitions are used obtained from bagging or clustering. The architecture allows parallel operation of the member classifiers.

There are many examples of the parallel classifier systems organized as shown in Fig. 2. In this paper we exemplify our discussion with parallel implementations of the two of our systems. The first one employs tensor based classifiers (HOSVD) trained with different data partitions obtained with data bagging [5]. The system was tested with the problem of handwritten character recognition showing significantly better results when compared to the single HOSVD classifier. The second of the systems, discussed with more details in this paper, contains one-class SVM classifiers [16][17]. Each of them is trained with a separate partition obtained from data clustering with the k-means and the kernel k-means methods [3][4]. This type of ensemble is discussed in further sections of this paper, whereas details of the tensor based system can be accessed in [5].

III. DESIGN OF THE BUILDING BLOCKS

In this section, we discuss details of blocks of the parallel systems of ensembles of classifiers shown in Fig. 2.

A. Data Splitters

The role of a data splitter is to arrange the training process in order to obtain the best accuracy of the ensemble. The two tested methods are as follows:

- Bagging - consists of creating a number of data sets D_i from the training set D with a uniform data sampling with replacement [18]. As shown by Grandvalet, bagging reduces variance of a classifier and improves its generalization properties [6]. Each set D_i is used to train a separate member of the ensemble, which contains less data than D. Thanks to this data decomposition a better accuracy can be obtained due to a higher diversity. Also, the problem of processing massive data can be greatly reduced. It is also possible to extend the ensemble with a new classifier if new training data are available at a later time.
- Data clustering - consists in usually unsupervised partitioning of the input dataset D into typically disjoint sets D_i [7][10]. In our previous systems the k-means as well as their fuzzy and kernel versions were used. In this case a first step is the choice of data centers. Then data distances to each center are computed and the points are assigned to their nearest centers. After that, positions of the centers are recomputed to account for new members of that partition. The procedure follows until there are no changes in data partitioning [4][18]. Similarly to bagging, splitting by clustering also allows better accuracy and data decomposition useful in parallel realizations.

B. Selection of the Member Classifiers

Choice of the member classifiers depends on many factors, such as type and dimensionality of data. However, the classifiers need to be chosen in a way to assure the best accuracy and speed of operation, especially when processing massive vision data. Good results were obtained in the tested systems using the aforementioned tensor classifiers, as well as using the OC-SVMs which we address in this section. Further references are provided in [4][5].

The binary SVMs were introduced by Cortes and Vapnik [1]. Their one-class version is due to Tax and Duin [16][17], as well as to Schölkopf and Smola [13]. Training of the OC-SVM relies on computation of the parameters of the hyperplane $\mathbf{w}$ that separates data points D_i with the maximal margin from the origin, as shown in Fig. 3. The separating hyperplane is defined as follows [13]

$$\langle \mathbf{w}, \mathbf{x} \rangle - \rho = 0, \tag{1}$$

where $\langle \mathbf{w},\mathbf{x} \rangle$ is a scalar product between vector $\mathbf{w}$ and $\mathbf{x}$. Further, to allow some outliers in the training set D_i with N data points, the slack variables ξ_n are introduced. This leads to the following convex optimization problem

$$\begin{aligned} &\min_{\mathbf{w},\xi_1 \ldots \xi_N,\rho} \left[\frac{1}{2}\|\mathbf{w}\|^2 + \frac{1}{\nu N}\sum_{n=1}^{N}\xi_n - \rho \right] \\ &\forall_{1 \le n \le N} \quad \langle \mathbf{w}, \mathbf{x}_n \rangle \ge \rho - \xi_n, \quad \xi_n \ge 0, \end{aligned} \tag{2}$$

where ν is a training parameter that controls allowable number of outliers. The above optimization problem can be solved with the Lagrange multipliers α_n, as follows

$$\begin{aligned} &\min_{\alpha_1 \ldots \alpha_N} \left[\sum_{n=1}^{N}\sum_{m=1}^{N} \alpha_n \alpha_m \langle \mathbf{x}_n, \mathbf{x}_m \rangle \right], \text{ with} \\ &\forall_{1 \le n \le N} \quad 0 \le \alpha_n \le \frac{1}{\nu N}, \text{ and } \sum_{n=1}^{N} \alpha_n = 1. \end{aligned} \tag{3}$$

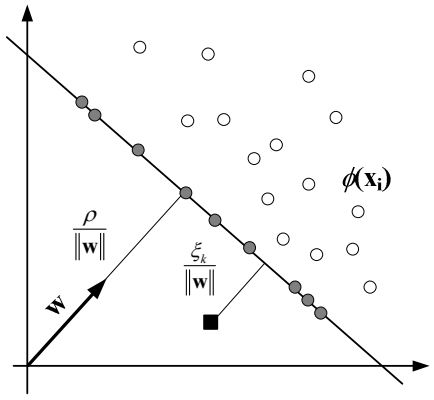

Figure 3. Hyperplane separating a single class of data in the feature space. Support vectors are on the hyperplane. An outlier (black square) is a data on the second side of the hyperplane. This one is controlled with the slack variable.

Data points for which α_n>0 lie on the hyperplane $\mathbf{w}$. These are called support vectors (SV). Solution to (3) results with a set of SV and associated scalar multipliers α_n. The hyperplane $\mathbf{w}$ can be represented as the following weighted sum of the SVs

$$\mathbf{w} = \sum_{n \in SVs} \alpha_n \mathbf{x}_n , \tag{4}$$

Taking any support vector $\mathbf{x}_m$ a distance of the hyperplane $\mathbf{w}$ to the origin can be computed as the following scalar product

$$\rho = \langle \mathbf{w}, \mathbf{x}_m \rangle = \sum_{n \in SVs} \alpha_n \langle \mathbf{x}_n, \mathbf{x}_m \rangle . \tag{5}$$

The real power of OC-SVM is their operation in the so called feature space. This is obtained replacing the inner product in (5) with the kernel function. For the latter the most frequent choice is the Gaussian kernel function, defined as follows [13][14]

$$K_G(\mathbf{x}_i, \mathbf{x}_j) = e^{-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|^2}, \tag{6}$$

where γ is a parameter that control a spread of the kernel. During classification of a test point $\mathbf{x}_x$ its distance to the hyperplane in the feature space is computed. This can be expressed as $K(\mathbf{w},\mathbf{x}_x)$, which if is greater than ρ indicates that a point belongs to the class. Thus, a point $\mathbf{x}_x$ is classified if the following inequality is fulfilled

$$\sum_{n \in SVs} \alpha_n K(\mathbf{x}_n, \mathbf{x}_x) \ge \underbrace{\sum_{n \in SVs} \alpha_n K(\mathbf{x}_n, \mathbf{x}_m)}_{p}. \tag{7}$$

To speed up response of the system the value on the right side of (7) can be precomputed to a scalar p.

Thanks to the relatively simple classification rule (7) its implementation on a graphic card is also not very complicated, as will be discussed in implementation section.

C. Fusion Module

Partial answers of the members of the ensemble need to be collected and used to produce a final response of the system. This is a task of the output fusion module, shown in Fig.2. Again, there are many algorithms for this task from which the following two were tested in the discussed ensembles:

- Majority voting
- Weighted majority voting

Further details are provided in the references [4][5][7][9].

IV. Implementation of the Ensemble of Classifiers

Two implementations were made in order to show properties of the proposed ensemble of classifiers. The first one is the serial software in C++ which uses the HIL library described in [2]. On the other hand, the parallel implementation of the system relies on the graphic cards with the CUDA (Compute Unified Device Architecture) environment [21]. CUDA is architecture of the parallel computing platform and programming model of the majority of the graphic cards by nVidia [21]. Two graphic cards were used in the tests: the FX3800M and the GTX280. Results of comparison of serial and parallel implementations are described in the next section.

Implementation of the CUDA kernel function *SVM_Answer_Kernel*, which carries out the OC-SVM classification, is shown in Fig. 6. This kernel is launched in parallel on the GPU devices. In the lines 8-11 of the listing in Fig. 6 an offset to the data buffer is computed to find out which part of data a kernel is assigned to. Then the *Compute_SV_DistanceTo* kernel function is invoked (shown in the line 16 of Fig. 6), which computes a distance of a test

 ISBN: 978-1-61208-258-5

point (a color pixel) to the hyperplane found in the training process. A training is done off line exclusively on the CPU. Found distance to the hyperplane of each of the test points is saved in the output buffer, as shown in the code lines 17-19 in Fig. 6.

Finally, the CUDA implementation of the Gaussian kernel is presented in Fig. 8. The function *exp_kernel* operates in accordance with (6). It can process data of any length which is provided by the *kDataDim* input parameter. In the case of color pixels *kDataDim*=3.

V. EXPERIMENTAL RESULTS

The ensemble of classifiers was tested in a time demanding task of human face detection from color images. For the training, manually gathered samples of pixels of human skin were used. However, the initial training set was clustered into three disjoint partitions with the k-means method, as described in Section IIIA. Then each of these partitions was used to train a separate base OC-SVM classifier. The parameters ν from (2) and γ from (6) of each of the OC-SVM were found by the grid search method. Operation of the system on the few color frames is depicted in Fig. 4.

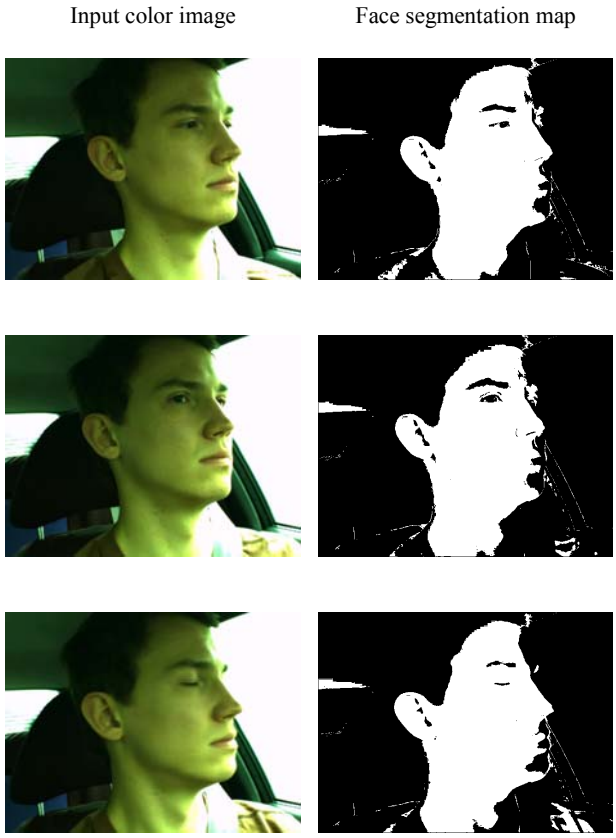

Figure 4. Examples of real-time face segmentation with the ensemble of three OC-SVM classifiers. Color images shown in the left column. Face segmentation maps shown in the right column.

Found parameters of the three base OC-SVM classifiers of the ensemble are shown in TABLE I. The parameter #SVs denotes number of support vectors and ρ is defined in (5).

TABLE I. PARAMETERS AND CONFIGURATION OF THE ENSEMBLES USED IN THE EXPERIMENTS

No.	#SVs	γ	ν	ρ
1	5	0.016279	0.0001262	0.780138
2	5	0.011379	0.0001198	0.864579
3	5	0.013299	0.0001626	0.783055

Accuracy of such ensembles in application of image segmentation reaches up to 0.85-0.90 of the *F* measure [18]. Almost in all examples an application of more than one classifier in an ensemble leads to the higher accuracy, as also discussed in [4]. However, thanks to their parallel architecture the real-time processing of HD images is easy to achieve. A plot of a speed-up ratio of the serial C++ vs. parallel CUDA implementation of the ensemble with the OC-SVM classifiers is shown in Fig. 5.

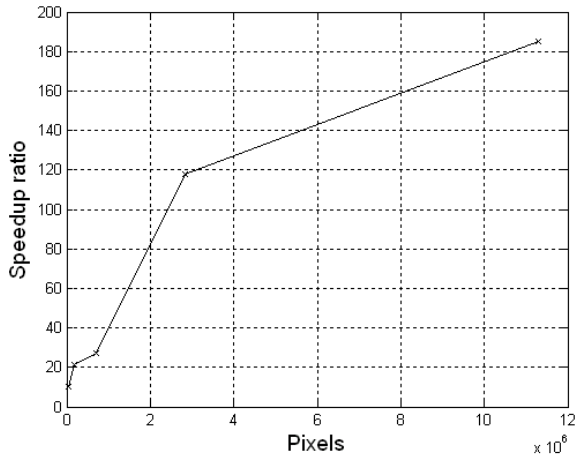

Figure 5. Speed-up ratio of the serial C++ vs. parallel CUDA implementations of the ensemble with OC-SVM classifiers.

In the above plot, we notice that for images which a large number of pixels, such as the ones with resolution 3968x2848, a speed-up ratio exceeds 180. This can be even higher on the FPGA platforms, at a cost of a much larger implementation effort, however. Also, interesting is comparison of serial implementations ported to the multi-core platforms, for example with help of the OpenMP library. In this case, a speed-up ratio of 3-5 times was achieved on the system with the Intel® quad-core processor i7 Q820 with a clock 1.73GHz and 8GB of the system RAM.

VI. CONCLUSIONS AND FUTURE WORK

In the paper different architectures of ensembles of classifiers suitable for parallel processing were discussed. It

 ISBN: 978-1-61208-258-5

was demonstrated that an application of an ensemble of classifiers usually leads to a higher accuracy when compared with a single classifier. In the paper an ensemble of OC-SVM classifiers was used in the problem of color image segmentation in real-time. It was shown that thanks to the highly parallel architecture, the ensemble with OC-SVM classifiers allows two-orders of magnitude speed-up ratio in the CUDA implementation when compared with the serial software version. Details of the parallel implementation with the CUDA code are also provided. It is worth noticing that the method is more general and allows classification of other than visual types of data as well.

Further research will be focused on development of new architectures with different base classifiers which are well suited for parallel implementations on different computer platforms.

ACKNOWLEDGMENT

Financial support in the years 2012-2013 of the National Center for Research and Development of the Republic of Poland, under the project SYNAT is greatly appreciated.

REFERENCES

[1] C. Cortes and V. Vapnik, Support vector machines. Machine Learning, Vol. 20, 1995, pp. 273-297.

[2] B. Cyganek and J.P. Siebert, An Introduction to 3D Computer Vision Techniques and Algorithms, Wiley, 2009.

[3] B. Cyganek and K. Wiatr, Image Contents Annotations with the Ensemble of One-Class Support Vector Machines. International Conference on Neural Computation Theory and Applications, 24-26 October, Paris, France, 2011.

[4] B. Cyganek, One-Class Support Vector Ensembles for Image Segmentation and Classification. Journal of Mathematical Imaging & Vision, Vol. 42, No. 2-3, Springer, 2012, pp. 103–117.

[5] B. Cyganek, Ensemble of Tensor Classifiers Based on the Higher-Order Singular Value Decomposition. HAIS 2012, Salamanca, Springer, Part II, LNCS 7209, 2012, pp. 578–589.

[6] Y. Grandvalet, Bagging equalizes influence. Machine Learning, Vol. 55, 2004, pp. 251-270.

[7] L.I. Kuncheva, Combining Pattern Classifiers. Methods and Algorithms. Wiley Interscience, 2005.

[8] M. Kurzyński and M. Woźniak, Combining classifiers under probabilistic models: experimental comparative analysis of methods. Expert Systems, Vol. 29, No. 4, 2012, pp. 374–393.

[9] R. Polikar, Ensemble Based Systems in Decision Making. IEEE Circuits and Systems Magazine, 2006, pp. 21-45.

[10] A. Rahman and B. Verma, Cluster-based ensemble of classifiers. Expert Systems, 2012.

[11] E. Rafajłowicz, Classifiers sensitive to external context theory and applications to video sequences. Expert Systems, Vol. 29, No. 1, 2012, pp. 84–104.

[12] J. Sanders and E. Kandrot, CUDA by Example. An Introduction To General-Purpose GPU Programming. Addison-Wesley, 2011.

[13] B. Schölkopf and A.J. Smola, Learning with Kernels, MIT Press, 2002.

[14] J. Shawe-Taylor and N. Cristianini, Kernel Methods for Pattern Analysis, Cambridge University Press, 2004.

[15] R. Tadeusiewicz and M.R. Ogiela, New Diagnostics Perspectives Connected with a Concept of Automatic Patterns Understanding. Chapter, in book by Korbicz J., Patan K., Kowal M., Fault Diagnostics and Fault Tolerant Control, EXIT, Warsaw, ISBN 83-60434-32-1, 2007, pp. 43-49.

[16] D.M.J. Tax, One-class classificatio. PhD thesis, TU Delft University, 2001.

[17] D. Tax, R. Duin, Support Vector Data Description, Machine Learning, Vol. 54, 2004, pp.45-66.

[18] S. Theodoridis and K. Koutroumbas, Pattern Recognition, 4th ed., Academic Press, 2009.

[19] P. Viola and M. Jones, Robust real-time face detection, Proceedings of the International Conference on Computer Vision, 2001, pp. 747-755.

[20] K. Wiatr, CYFRONET supercomputers in support of modern research projects, KU KDM 2011, fourth ACC Cyfronet AGH users' conference, Zakopane, March 09–11, 2011.

[21] www.nvidia.com

```
1     __global__ void SVM_Answer_Kernel( const float * inDataBuf, unsigned char * outDataBuf,
2                                        float * outValuesBuf, const int kTotalNumOfData,
3                                        float * SV_vectors, float * alpha_vector,
4                                        const int kNumOfSVs, const float kGamma,
5                                        const float kRho, const int kDataDim  )
6     {
7          // Here we map threadIdx and BlockIdx to pixel position in the buffer
8          int x = threadIdx.x + blockIdx.x * blockDim.x;
9          int y = threadIdx.y + blockIdx.y * blockDim.y;
10
11         int offset = x + y * blockDim.x * gridDim.x;
12
13         if( offset < kTotalNumOfData )        {
14              const float * this_data_offset = inDataBuf + offset * kDataDim;
15              // since inDataBuf and SV_vectors actually are 2D structures
16              float distance = Compute_SV_DistanceTo(  this_data_offset, SV_vectors,
                                                         alpha_vector, kNumOfSVs, kGamma, kDataDim );
17              float val = distance - kRho;
18              * ( outValuesBuf + offset ) = val;
19              * ( outDataBuf + offset ) = val > 0.0 ? 0 : 255;
20         }
21    }
```

Figure 6. Implementation of the CUDA kernel for the OC-SVM classifier. The *Compute_SV_DistanceTo* function is called (shown in Fig. 7) which computes a distance of a test point (a pixel) to the hypersphere.

```
1     // SV_vectors and alpha_vector should be in the constant memory
2     __device__ float Compute_SV_DistanceTo(    const float * data_point, const float * SV_vectors,
3                                                const float * alpha_vector, const int kNumOfSVs,
4                                                const float kGamma, const int kDataDim )
5     {
6          // We assume RBF kernels ONLY!!
7          float alpha = 0.0;
8          float kernel_product = 0.0;
9
10         // Compute the second term
11         register double theSum = 0.0;
12         #pragma unroll
13         for( register int i = 0; i < kNumOfSVs; ++ i )
14         {
15    #if USE_CONST_MEMORY
16
17              kernel_product  = exp_kernel( in_SVs_vector_GPU + i * kDataDim, data_point, kGamma, kDataDim );
18              alpha           = in_alphas_vector_GPU[ i ];
19
20    #else //USE_CONST_MEMORY
21
22              kernel_product  = exp_kernel( SV_vectors + i * kDataDim, data_point, kGamma, kDataDim );
23              alpha           = alpha_vector[ i ];
24
25    #endif //USE_CONST_MEMORY
26
27              theSum     += alpha * kernel_product;
28         }
29
30         return theSum;
31    }
```

Figure 7. CUDA kernel function executed for each pixel. The function invokes the *exp_kernel* function shown in Fig. 8.

```
1     __device__ float exp_kernel( const float * x, const float * y, const float kGamma, const int kDataDim )
2     {
3          register float tmp = 0.0;
4          register float sum = 0.0;

5          #pragma unroll
6          for( register int d = 0; d < kDataDim; ++ d )
7          {
8               tmp = x[ d ] - y[ d ];
9               sum += tmp * tmp;
10         }

11         return expf( - kGamma * sum );
12    }
```

Figure 8. CUDA implementation of the exponential (Gaussian) kernel.

 ISBN: 978-1-61208-258-5