E-Learning and Benchmarking Platform for Parallel and Distributed Computing

http://dx.doi.org/10.3991/ijet.v9i2.3215

Marjan Gusev, Sasko Ristov, Goran Velkoski, and Bisera Ivanovska
Ss. Cyril and Methodius University, Skopje, Macedonia

***Abstract*—Emerging technologies and increased computing resources challenge the universities to establish learning environments, where the students will learn to efficiently utilize multicore or multi-chip multiprocessors. In this paper, we present a new pilot project with the following goals: (1) as an e-Learning platform for the course on Parallel and Distributed Computing, and (2) as a benchmarking platform to measure the algorithm performance indicators such as speed, speedup and efficiency of the parallel implementation.**

***Index Terms*—Education; Multi-processor; OpenMP; Performance; Remote Engineering.**

I. INTRODUCTION

CPU vendors have changed the research and development directions in the last decade, that is, from pursuing the CPU clock speed towards building CPUs with multiple cores per chip, since the CPU clock speed increases the power consumption and heat dissipation exponentially. Many of today's multiprocessors are not just multicore, but even multi-chip multiprocessors. These innovations of establishing multicore and multi-chip multiprocessors raise the need for more relevant and efficient parallel algorithms. Although the parallelism topics are listed as elective, the IEEE Computer Society and ACM believe that the role of parallelism throughout the computer science curriculum needs to be considered [1]. Also, the parallel multiprocessor architectures are proposed to be covered even in the basic course Computer Architecture and Organization, rather than digital circuits. Many universities implement parallelization issues in other traditional (core) courses, such as Computer Architecture and Organization, Operating Systems, Pervasive Computing, and other courses [2]. Organizing regular contests will disseminate the parallelism among computer science students [3].

Learning parallel and distributed computing based courses requires usage of appropriate equipment, especially when stu-dents work on lab exercises and projects. Although students' desktop computers or notebooks are multicore based, there are a lot of obstacles. For example, these personal computers are usually installed with operating systems and inappropriate to support a parallel computing platform, such as OpenMP or MPI. The alternative for students is to work in a virtual ma-chine instance, but, in this case, they usually achieve performance degradation. Also, teachers cannot verify the correct-ness of homework and projects due to different hardware infrastructure that student might use. For example, results about speed, speedup, efficiency, CPU utilization and so on depend on operating systems and platforms. Lab equipment can be often used only within the perimeter of the University network. University security policies usually prevent the student access outside this perimeter, especially the access to data center servers, which have multiprocessors appropriate for the course of Parallel and Distributed Computing (PDC).

Remote engineering is a trend for sharing the lab equipment and allows the students and researchers with special needs to work from home without traveling to the University [4]. E-Learning development directions should follow open access to learning, open source software, open standards, and open educational resources [5]. However, effort and assistance from experts are required to setup a fully operational open source workbench [6].

In this paper, we present our new pilot project "e-Learning and Benchmarking platform for PDC". We designed a scalable architecture for executing and benchmarking different parallel or sequential codes and to analyze their performance. However, the main purpose of the e-Learning platform is to be used as a baseline in learning the parallelization issues. The students can create a parallel implementation of a given algorithm using already prepared examples (such as basic linear algebra algorithms), as well as a benchmark algorithm to examine the correctness, effectiveness and the efficiency of the student's algorithm.

The paper is organized as follows. Section II presents the efforts on how to lighten the teaching and learning process of a PDC course. In Section III, we present the PDC course organization and challenges that motivated us to develop a new pilot e-Learning and benchmarking platform. The architecture, user interface and the sequence diagram of this pilot e-Learning and benchmarking platform are described in Section IV. Finally, we conclude our work and present a plan for future development of the platform in Section V.

II. RELATED WORK

Using appropriate assignments, hands-on exercises, projects and visual simulators, as well as teaching methodology will lighten the teaching and learning process of PDC course. For example, Liang [7] presented a nice survey of hands-on assignments and projects. Another example is demonstrated by Hatfield and Jin [8]. They designed and developed many lab exercises to design and implement an operating model of a pipelined processor.

There are several simulators that might be used for learning the PDC course. Zhi et al. [9] designed an execution-driven parallel simulator for predicting the performance of large-scale parallel computers, simulating the processors of each node, network components and disk I/O components. However, we observed that none of these simulators could be used in the teaching and learning processes, due to their complexity.

Breuer and Bader [10] designed a software package that supports teaching and learning different parallel programming models (OpenMP, MPI and CUDA). However, this package is useful for more advanced courses, rather than basic PDC.

Rague [11] introduced the Parallel Analysis Tool (PAT), a pedagogical tool, which assists the undergraduate students to transform a specific code section from sequential to parallel. The students can examine the impact of parallelizing a block of code at a time on overall program performance. Atanasovski [12] and Ristov [22] developed the EDUCache simulator and exercises with examples for it to simulate cache hits and misses for a particular memory trace in randomly designed multiprocessor. However, none of these simulators allow the students to start the learning process and incrementally develop more complex parallel algorithms.

Many authors proposed methodologies to lighten the learning. Liu et al. [13] developed software and application lab modules, which give the students clear understanding of the principles and practice of parallel programming on multicore systems. Bunde [14] propose introduction of a small unit for multi-threaded programming in core courses. Michailidis and Margaritis [15] proposed a simple and fast analytical model to predict the performance of matrix computations by taking memory access costs into account and data access schemes that appear in many scientific applications. Delistavrou and Margaritis [16] survey the software environments for parallel programming categorizing them according to their main function. Freitas [17] proposed OpenMP (shared-memory model) to be used in the computer science curriculum before MPI (message-passing model), but after consolidating sequential programming, operating systems, computer architecture and networks.

Li et al. [18] improved the PDC course including the information about multicore needs in the learning materials, put emphasis on OpenMP and MPI, and involved programming practice. However, although these issues were already implemented in our PDC course, the students faced many other challenges and obstacles described in Section III-B. Iparraguirre et al. [19] evaluated that practical work, and small practical assignments rather than one big project, will result in a visualization effect. Introducing PDC concepts in the related courses of the University curricula is a requirement set by the students enrolled in the elective PDC course.

III. THE PARALLEL AND DISTRIBUTED COMPUTING COURSE

This section briefly describes the PDC course, its organization and the challenges to establish more efficient learning environment.

A. Course Organization

The PDC course is taught in the 6th semester. The course is organized in three parts: theoretical lectures with 2 classes per week, theoretical exercises with 1 class per week and practical exercises with 2 classes per week in a lab. A (small) group of maximum 20 students usually enrolls the course and therefore both the theoretical and lab assignments are organized for all students in computer labs, with each student working on its own workstation. Prerequisites for enrolling the course are previously completed courses “Computer Architecture and Organization” and “Operating systems”. The course is obligatory for the “Computer Architectures and Networks” curricula and elective for the computer science students.

Theoretical lectures cover the HPC (High Performance Computing) concepts and architectures, performances and limits, topologies and interconnections, concurrent communications and programming, and program and algorithm transformations. Theoretical assignments are divided in two parts. The first part until midterm covers the topics of shared memory programming using OpenMP, while the second part is devoted on message-passing programming using MPI. Hands-on lab exercises follow the topics of the theoretical assignments.

The focus is set on practical work. Homework is assigned both from the theoretical lectures and exercises on a weekly basis. Students are supposed to work on two projects, the first one using OpenMP and the second, MPI. After finalization they have to present the programs and results. The students must attend the theoretical lecture part of the course, defend both projects and finish all homework to pass the course. The hands-on lab exercises are not graded since they are intended as individual preparatory work. The students should not only develop a parallel implementation of a certain algorithm, but they should perform benchmark tests, that is, to measure the algorithm speed, speedup, efficiency, utilization, and so on.

B. Challenges and Motivation

The students faced several obstacles and challenges. Their main reasonable obstacle is the impossibility to continue to work on lab assignments at home if they had not finished all required tasks during the lab. However, the access to the lab is forbidden outside the University network, and all those that are not present in the University Campus face real problems. The students also faced problems while they were working on the homework assignments. Their notebooks or home workstations usually run on Windows operating system, or Mac OS, rather than Linux. The students can solve homework assignments using a virtual machine instance, but they might expect performance degradation in this case. From teacher’s perspective we have faced other problems, for example, we could not verify the homework correctness due to existence of different hardware infrastructure per student that is each student achieved different speed, speedup, efficiency, CPU utilization and so on. Another problem was raised when some of students tried to fabricate the results, tailoring the output according to the results of their colleagues.

All these challenges motivated us to develop a new web based e-Learning and benchmarking platform for learning PDC course, trying to solve or reduce the previously mentioned obstacles as much as possible. Using this platform, all the students are now enabled to work on their homework, lab exercises and projects using the same runtime environment and the same hardware infrastructure, appropriate for the course. Also, all their results can be compared with the teacher’s nominal expectations.

IV. THE E-LEARNING AND BENCHMARKING PLATFORM

This section describes our new e-Learning platform for PDC course, its main features and user interface on the front-end application.

A. *The Architecture*

Figure 1 depicts the architecture of the e-Learning and benchmarking platform. It is a client-server based platform independent system developed in JAVA. The platform consists of two servers: the *Application* and *Testing* servers.

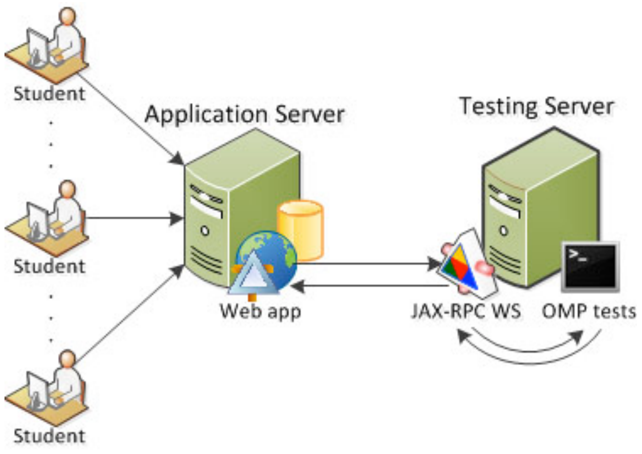


Figure 1. E-Learning platform architecture

The application server is Ubuntu server 12.04 and hosts the web application which is the front-end application accessed by the user. The application offers the students several possibilities. They can execute several predefined linear algebra algorithms, such as matrix-matrix multiplication, matrix-vector multiplication, dot or inner product and so on. The students can select the sequential or parallel implementation with certain number of threads, the matrix and the vector sizes. The application sends the parameters to the web service hosted on the testing server, which executes the chosen algorithm with the parameters and returns the results to the students. The next Section IV-C describes the front-end application user interface in detail. This server works also as a database server to store the students' credentials. The same database is also used for storing the results of previous executions.

The most important subsystem is the testing server. It is also an Ubuntu 12.04 sever installed with necessary OpenMP and MPI libraries. The testing server hosts the web service and benchmark program. The web service is a gateway between the front-end application and the benchmark program. It accepts the parameters (entered by the students) from the front-end application and calls the benchmark program using the selected algorithm. Finally, it continues with execution of multi-threads within the OpenMP environment, or multi-processes within the MPI environment, or with sequential execution of a single process (thread). The benchmark program returns the results to the application through the gateway web service.

B. *The User Interface*

The e-Learning and benchmarking platform user interface is a user-friendly visual environment and uses the Multiple Document Interface (MDI) paradigm. The previous section described the architecture of the e-Learning and benchmarking platform. In this section, we describe the user interface of the front-end application. Figure 2 depicts the main menu after student's successful login.

The main window contains 3 main panels described in details in the following three subsections.

1) *Left Panel – The Learning Panel*

The panel on the left side offers the students a possibility to select which algorithm they would like to analyze. This pilot platform offers three algorithms: matrix-matrix multiplication, matrix-vector multiplications and inner (dot) product. The application depicts the algorithm execution. Figure 2 depicts how the matrix-matrix multiplication algorithm works.

2) *Central Panel – The Benchmarking Panel:*

The central panel offers the student a possibility to benchmark the selected algorithm in the learning panel. They can select either a sequential or a parallel implementation. The parallel implementation further offers a selection between OpenMP and MPI. According to the selected algorithm and its implementation, this panel automatically shows the necessary parameters: the matrices' or vector's sizes and the number of threads or processes for parallel implementation. We implemented an additional field NLOOPS in order to repeat the execution NLOOPS times to achieve more reliable results. After clicking "START", the application checks the database if the selected algorithm was executed with the same parameters previously. If there are results stored in the database, the application presents them immediately. This is usually the case on the lab exercises since all the students work on the same exercise. However, this is not useful when they work on their homework. There is a "REFRESH" button to execute the algorithm again even though the results are stored in the database.

After achieving the results of a web service (or loading from the database), they are presented in the lower part of the benchmarking panel. The application shows the average execution time of the sequential implementation, and additionally the execution time, speedup and efficiency of the parallel implementation. For example, we observe almost linear speedup of 2.36 using 3 threads with OpenMP multiplying matrices 100x200 with 200x100.

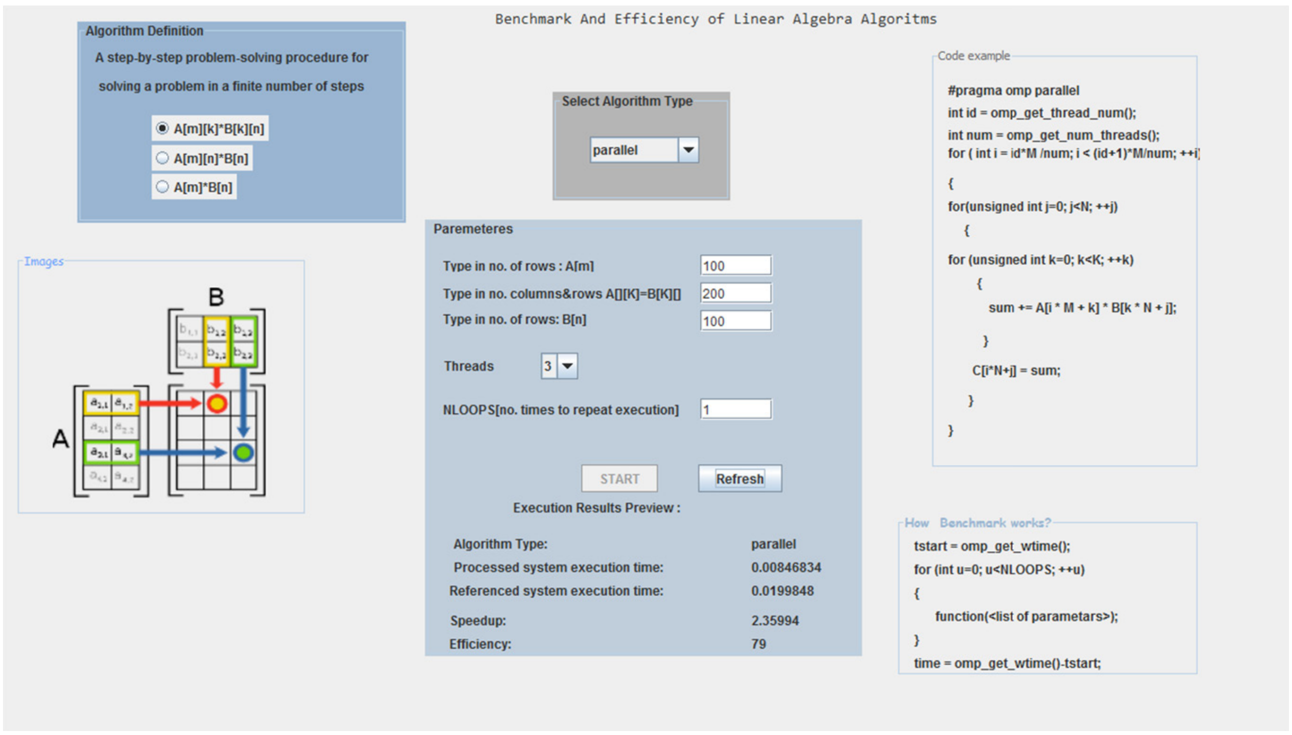
3) *Right Panel – Learning and Benchmarking Panel*

The right panel offers students a source code of the selected algorithm and certain implementation in C++. All algorithms are organized in functions and the parameters are transferred as arguments for the already compiled C++ testing program. The built in example in the main program generates a random matrix of vector elements, measures the average execution time and checks the correctness of the result.

Inappropriate usage of OpenMP directives can lead to a possible execution slowdown or even wrong execution results of the parallel implementation, as presented by Li et al. [20]. Therefore we introduce this panel where students can get the code example in order to check the correctness of the results. Presenting the expected speedup also will guide the students if their directives are written well, that is, if they achieve the expected speedup.

C. *Sequence Diagram for Test Execution*

Figure 3 presents the sequence diagram of the e-Learning platform in terms of how the student executes a particular test. In order to login at the web page, the student must first insert login credentials. These credentials



Algorithm Definition

A step-by-step problem-solving procedure for solving a problem in a finite number of steps

[✓ A[m][k]*B[k][n]] A[m][n]*B[n] A[m]*B[n]

Select Algorithm Type: parallel

Paremeteres

Type in no. of rows : A[m]	100
Type in no. columns&rows A[][K]=B[K][]	200
Type in no. of rows: B[n]	100
Threads	3
NLOOPS[no. times to repeat execution]	1

Execution Results Preview :

Algorithm Type:	parallel
Processed system execution time:	0.00846834
Referenced system execution time:	0.0199848
Speedup:	2.35994
Efficiency:	79

Code example

```
#pragma omp parallel
int id = omp_get_thread_num();
int num = omp_get_num_threads();
for ( int i = id*M /num; i < (id+1)*M/num; ++i)

{
for(unsigned int j=0; j<N; ++j)
   {
for (unsigned int k=0; k<K; ++k)
      {
         sum += A[i * M + k] * B[k * N + j];
       }
     C[i*N+j] = sum;
    }
}
```

How Benchmark works?

```
tstart = omp_get_wtime();
for (int u=0; u<NLOOPS; ++u)
{
    function(<list of parametars>);
}
time = omp_get_wtime()-tstart;
```

Figure 2. Main menu of e-Learning and benchmarking platform

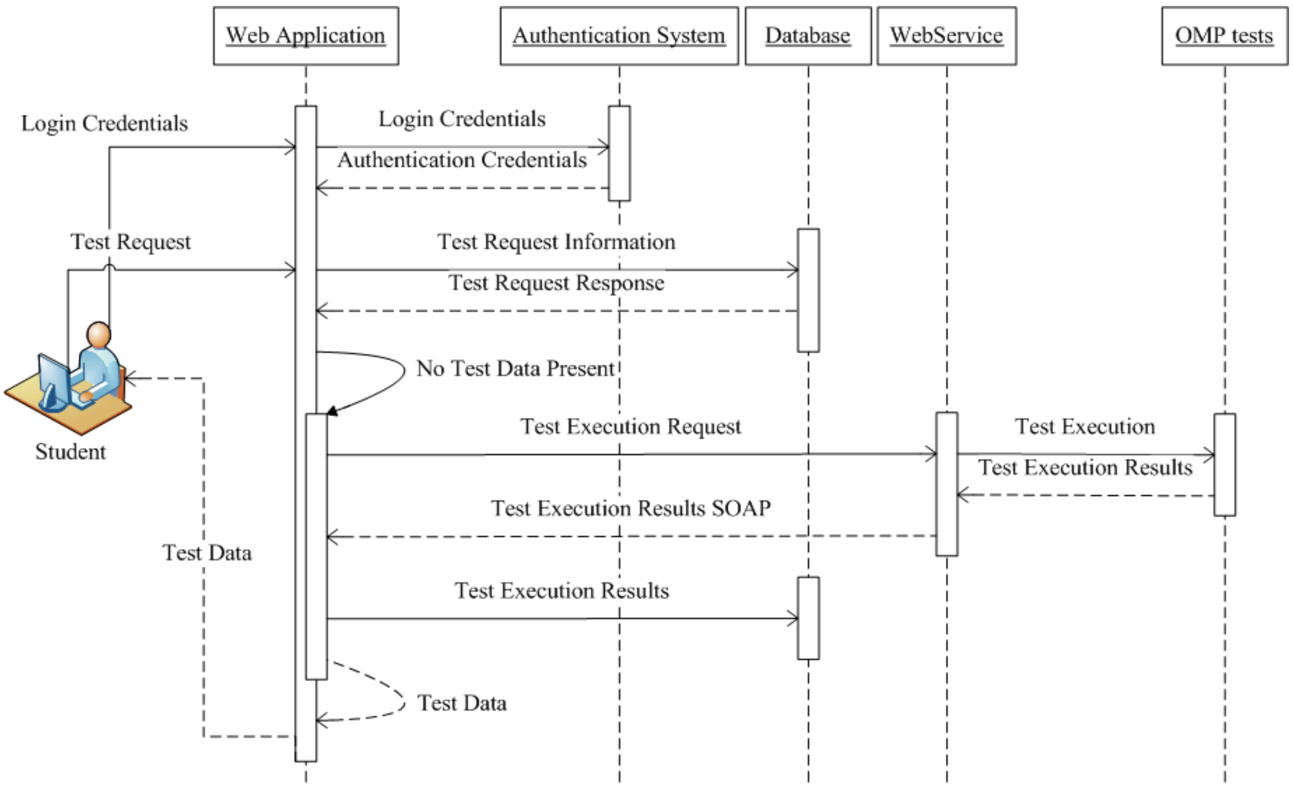


Figure 3. Sequence diagram for executing test on e-Learning and benchmarking platform

are then passed to an external authentication system, which authenticates the user and authorizes user access for the web application.

After successful authentication the user can request tests by passing test information such as algorithm type, benchmarking algorithm type (sequential or parallel, OpenMP or MPI) and benchmarking parameters such as matrix or vector sizes, the number of threads or processes, and number of test repetition. Based on the test parameters the application queries the database for previous executions. If the test result data is found, it is passed back to the user; otherwise a special web service is invoked. This web service requests OpenMP test execution on one of the configured servers. After the OpenMP test finishes, the test results are sent back to the user through the web service and these results are written in the database.

V. CONCLUSION AND FUTURE WORK

This paper presents the new pilot project for e-Learning and benchmarking platform for learning the PDC or other related courses. The contribution of this platform is two-fold: 1) it allows the students a possibility to learn parallelization concepts and parallelize more complex algorithms; and 2) to benchmark the algorithm execution with a particular problem size and check if it achieves the expected performance indicators speed, speedup and efficiency for parallel implementation.

The students can work remotely on the same hardware infrastructure and runtime environment and compare their benchmarking results for a particular algorithm with a certain problem size. This is very important since the memory access time, cache misses and hits, and also cache associativity affect the algorithm overall performance.

Although this platform reduces the students' problems, we faced a problem even at the beginning. This is a situation when two or more students execute the benchmark programs in the same time using higher number of threads or processes, greater than the server can stand for. Most of the results in this case become unreliable. Therefore, we will use the cloud based architecture of the e-Assessment system [21] in order to upgrade this e-Learning and benchmarking platform prototype into a scalable and elastic platform, where the system will send the execution on a virtual machine instance hosted on the cloud node with available resources.

REFERENCES

[1] ACM/IEEE-CS Joint Interim Review Task Force, "Computer science curriculum 2008: An interim of CS 2001, report from the interim review task force," 2008. [Online]. Available: http://www.acm.org/education/curricula/ComputerScience2008.pdf.

[2] T. Chen, Q. Shi, J. Wang, and N. Bao, "Multicore challenge in pervasive computing education," in *Proceedings of the 2008 the 3rd Int. Conf. on Grid and Pervasive Computing*. 2008, pp. 310-315.

[3] F. Almeida, J. Cuenca, R. Fernandez-Pauscal, D. Gimenez, and J. A. P. Benito, "The Spanish parallel programming contest and its use as an educational resource," in *Proc. of the IEEE 26th Int. Parallel and Distributed Processing Symposium Workshop,* 2012, pp.1303-1306.

[4] D. Pop, D. G. Zutin, M. E. Auer, K. Henke, and H.-D. Wuttke, "An online lab to support a master program in remote engineering," in *Proceedings of the 2011 Frontiers in Education Conference*, ser. FIE'11. 2011, pp. GOLC2-1-1-GOLC2-6

[5] D. Dinovski, "Open educational resources and lifelong learning," in *Information Technology Interfaces, 2008. ITI 2008. 30th International Conference on,* 2008, pp. 117-122.

[6] C. T. Delistavrou and K. G. Margaritis, "Towards an integrated teaching environment for parallel programming," in *Proceedings of the 2011 15th Panhellenic Conference on Informatics,* ser. PCI '11. 2011, pp. 3-7.

[7] X. Liang, "A survey of hands-on assignments and projects in undergraduate computer architecture courses," in *Advances in Computer and Information Sciences and Engineering,* T. Sobh, Ed. Springer Netherland, 2008, pp. 566-570.

[8] B. Hatfield and L. Jin, "Improving learning effectiveness with hands-on design labs and course projects for the operating model of a pipelined processor," in *Frontiers in Education Conference (FIE), 2010 IEEE,* 2010, pp. F1E-1-F1E-6.

[9] Y. Zhi, Y. Liu, L. Jiao, and P. Zhang, "A parallel simulator for large-scale parallel computers," in *Grid and Cooperative Computing (GCC), 2010 9th International Conference on,* 2010, pp. 196-200.

[10] A. Breurer and M. Bader, "Teaching parallel programming models on a shallow-water code," in *Proc. of the 2012 11th Int. Symposium on Parallel and Distributed Computing,* ser. ISPDC '12. 2012, pp. 301-308.

[11] B.W. Rague, "Exploring concurrency using the parallel analysis tool," in *Proceedings of the 43rd ACM technical symposium on Computer Science Education,* ser. SIGCSE '12. ACM, 2012, pp. 511-516.

[12] B. Atanasovski, S. Ristov, M. Gusev, and N. Anchev, "EDUCache simulator for teaching computer architecture and organization," in *Global Engineering Education Conference (EDUCON), 2013 IEEE,* March 2013, pp. 1015-1022.

[13] P. Liu, G. C. Lee, J.-K. Lee, and C.-Y. Lin, "Innovative system and application curriculum on multicore systems," in *Proceedings of the 6th Workshop on Embedded Systems Education,* ACM, 2011, pp. 25-31.

[14] D. P. Bunde, "A short unit to introduce multi-threaded programming," *J. Comput. Sci. Coll.,* vol. 25, no. 1, pp. 9-20, Oct. 2009.

[15] P. D. Michailidis and K. G. Margaritis, "Performance models for matrix computations on multicore processors using OpenMP," in *Proceedings of the 2010 Int. Conference on Parallel and Distributed Computing, Applications and Technologies,* ser. PDCAT '10. 2010, pp. 375-380.

[16] C. T. Delistavrou and K. G. Margaritis, "Survey of software environments for parallel distributed processing: Parallel Programming education on real life target systems using production oriented software tools," in *Proceedings of the 2010 14th Panhellenic Conference on Informatics,* ser. PCI'10. 2010, pp. 231-236.

[17] H. de Freitas, "Introducing parallel programming to traditional undergraduate courses," in *Frontiers in Education Conference (FIE), 2012,* 2012, pp. 1-6.

[18] J. Li, W. Guo, and H. Zheng, "An undergraduate parallel and distributed computing course in multi-core era," in *Proceedings of the 2008 the 9th International Conference for Young Computer Scientists*, ser. ICYCS '08. 2008, pp. 2412-2416.

[19] J. Iparraguirre, G. Friedrich, and R. Coppo, "Lessons learned after the introduction of parallel and distributed computing concepts into ECE undergraduate curricula at UTN-Bahia blanca Argentina," in *Parallel and Distributed Processing Symposium Workshops Phd Forum (IPDPSW), 2012 IEEE 26th International,* 2012 pp. 1317-1320.

[20] J. Li, D. Hei, and L. Yan, "Correctness analysis based on testing and checking for OpenMP programs" in *Proceedings of the 2009 Fourth China Grid Annual Conference,* ser. CHINAGRIP, 2009, pp.210-215.

[21] S. Ristov, M. Gusev, G. Armenski, K. Bozinovski, and G. Velkoski, "Architecture and organization of e-assessment cloud solution," in *Global Engineering Education Conference (EDUCON), 2013 IEEE,* March 2013, pp. 736-743, best paper award.

[22] S. Ristov, B. Atanosovski, M. Gusev and N. Anchev, "Hands-On Exercises to Support Computer Architecture Students Using EDUCACHE Simulator", in *2013 Federated Conference on Computer Science and Information Systems FEDCIS'13.* Krakow, Poland.2013, pp. 769-776

AUTHORS

Marjan Gusev, Sasko Ristov, Goran Velkoski, and **Bisera Ivanovska** are with "Ss. Cyril and Methodius" University, Skopje (Faculty of Information Sciences and Computer Engineering,Rugjer Boshkovik 16, PO Box 393, 1000 Skopje, Macedonia) Email: {marjan.gushev, sashko.ristov}@finki.ukim.mk, velkoski.goran@gmail.com, ivanovskabisera@yahoo.com

This article is an extended and modified version of a paper presented at the 2013 IEEE International Conference on Teaching, Assessment and Learning for Engineering (TALE2013), held 26-29 August 2013, Bali Dynasty Resort, Kuta, Indonesia. Submitted 29 September 2013. Published as re-submitted by the authors 20 March 2014.