ADVANCES IN INTELLIGENT AND SOFT COMPUTING 152

Pierpaolo Vittorini
Rosella Gennari
Ivana Marenzi
Fernando De la Prieta
Juan M. Corchado Rodríguez (Eds.)

International Workshop on Evidence-Based Technology Enhanced Learning

Springer

Advances in Intelligent and Soft Computing 152

Editor-in-Chief

Prof. Janusz Kacprzyk
Systems Research Institute
Polish Academy of Sciences
ul. Newelska 6
01-447 Warsaw
Poland
E-mail: kacprzyk@ibspan.waw.pl

For further volumes:
http://www.springer.com/series/4240

Pierpaolo Vittorini, Rosella Gennari,
Ivana Marenzi, Fernando De la Prieta,
and Juan M. Corchado Rodríguez (Eds.)

International Workshop on Evidence-Based Technology Enhanced Learning

Springer

Editors
Pierpaolo Vittorini
Department of Internal Medicine
and Public Health
University of L'Aquila
Italy

Fernando De la Prieta
Department of Computing Science
University of Salamanca
Salamanca
Spain

Rosella Gennari
Computer Science Faculty
Free University of Bozen-Bolzano
Bolzano
Italy

Juan M. Corchado Rodríguez
Department of Computing Science
University of Salamanca
Salamanca
Spain

Ivana Marenzi
L3S Research Center
Leibniz University of Hanover
Hannover
Germany

ISSN 1867-5662 e-ISSN 1867-5670
ISBN 978-3-642-28800-5 e-ISBN 978-3-642-28801-2
DOI 10.1007/978-3-642-28801-2
Springer Heidelberg New York Dordrecht London

Library of Congress Control Number: 2012933102

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

Research on Technology Enhanced Learning (TEL) investigates how information and communication technologies can be designed in order to support pedagogical activities. The Evidence Based Design (EBD) of a system bases its decisions on empirical evidence and effectiveness. The evidence-based TEL workshop (ebTEL) brings together TEL and EBD. The workshop proceedings collects contributions concerning evidence based TEL systems, like their design following EBD principles as well as studies or best practices that educators, education stakeholders or psychologists used to diagnose or improve their students' learning skills, including students with specific difficulties (e.g. poor/slow readers, students living in impoverished communities or families).

The ebTEL'12 workshop was launched under the collaborative frame provided by the European TERENCE project (http://www.terenceproject.eu). The TERENCE project, n. 257410, is funded by the European Commission through the Seventh Framework Programme for Research and Technological Development, Strategic Objective ICT-2009.4.2: ICT: Technology-enhanced learning. TERENCE is building an AI-based Adaptive Learning System (ALS) for reasoning about stories, in Italian and in English, through reading comprehension interventions in the form of smart games. The project also aims at developing innovative usability and evaluation guidelines for its users. The guidelines and the ALS will result from a cross-disciplinary effort of European experts in diverse and complementary fields (art and design, computers, engineering, linguistics and medicine), and with the constant involvement of end-users (persons with impaired hearing and their educators) from schools in Brighton (Great Britain), and from Veneto region (Italy).

The ebTEL workshop invited authors in the area of TEL, both from computer science as well as evidence-based medicine, educational psychology and pedagogy. In this manner, the international ebTEL'12 workshop wants to be a forum in which TEL researchers and practitioners alike can discuss ideas, projects, and lessons related to ebTEL. The workshop takes place in Salamanca, Spain, on March 28th-30th 2012.

This volume presents the papers that were accepted for the first edition of ebTEL. The full program contains 16 selected papers (3 in the Psychology and Pedagogy Track and 13 in the Information and Communication Technology Track) from 6 countries (Germany, Italy, Romania, Saudi Arabia, Spain, United Kingdom, USA). Each paper was reviewed by, at least, two different reviewers, from an international committee composed of 27 members of 5 countries. The quality of papers was on average good, with an acceptance rate of 76.2%.

We would like to thank all the contributing authors, the reviewers, the sponsors (IEEE Systems Man and Cybernetics Society Spain, AEPIA Asociación Española para la Inteligencia Artificial, APPIA Associação Portuguesa Para a Inteligência Artificial, CNRS Centre national de la recherché scientifique and STELLAR), as well as the members of the Program Committees, TERENCE members and the Organising Committee for their hard and highly valuable work. Special thanks are due to Wolfgang Nejdl of the L3S Research Center of the University of Hannover (Germany), invited speaker of the workshop with the talk "Web Science @ L3S – Interdisciplinary Research Challenges". The work of all these people contributed to the success of the first edition of the ebTEL'12workshop. Thank you all for your help, ebTEL'12 would not exist without your contribution.

The Editors
Pierpaolo Vittorini
Rosella Gennari
Ivana Marenzi
Fernando De la Prieta
Juan M. Corchado Rodríguez

Organization

Steering Committee

Juan M. Corchado	University of Salamanca, Spain
Fernando De la Prieta	University of Salamanca, Spain
Rosella Gennari	Free University of Bozen-Bolzano, Italy
Ivana Marenzi	L3S Research Center, Germany
Pierpaolo Vittorini	University of L'Aquila, Italy

Program Committee

Fabian Abel	Delft University of Technology, Nederland
Tomaz Amon	AMNIN, Slovenia
Anthony Baldry	University of Messina, Italy
Juan M. Corchado	University of Salamanca, Spain
Fernando De la Prieta	University of Salamanca, Spain
Juan F. De Paz Santana	University of Salamanca, Spain
Dina Di Giacomo	University of l'Aquila, Italy
Tania Di Mascio	University of l'Aquila, Italy
Rosella Gennari	Free University of Bozen-Bolzano, Italy
Ana Belén Gil	University of Salamanca, Spain
Óscar Gil	University of Salamanca, Spain
Nicola Henze	L3S, Leibniz University of Hanover, Germany
Eelco Herder	L3S, Leibniz University of Hanover, Germany
Ralf Kerstel	L3S, Leibniz University of Hanover, Germany
Ralf Klamma	RWTH Aachen University, Germany
Rita Kupetz	Leibniz University Hanover, Germany
Fabrizio Maggi	University of Pavia, Italy
Ivana Marenzi	L3S, Leibniz University of Hanover, Germany
Maria Moreno Jaén	University of Granada, Spain
Stefano Necozione	University of l'Aquila, Italy
Emanuele Pianta	FBK-irst, Italy
Sara Rodríguez	University of Salamanca
Maria Grazia Sindoni	University of Messina, Italy
Marcus Specht	Open University of the Netherlands
Sara Tonelli	FBK-irst, Italy
Pierpaolo Vittorini	University of l'Aquila, Italy
Carolina Zato	University of Salamanca, Spain

Local Organising Committee

Juan M. Corchado (Chairman)	University of Salamanca, Spain
Javier Bajo	Pontifical University of Salamanca, Spain
Sara Rodríguez	University of Salamanca, Spain
Dante I. Tapia	University of Salamanca, Spain
Emilio Corchado	University of Salamanca, Spain
Juan F. De Paz	University of Salamanca, Spain
Fernando De la Prieta	University of Salamanca, Spain
Davinia Carolina Zato Domínguez	University of Salamanca, Spain
Óscar Gil Gonzalo	University of Salamanca, Spain

Contents

Psychology and Pedagogy

Information and Communication Technology

Understanding Computer Architecture with Visual Simulations: What Educational Value?

Besim Mustafa and Peter Alston

Abstract. Many software simulators have been created for educational purposes. Such educational tools need to be both engaging and pedagogically sound if they are to enhance students' learning experiences. A system simulator for computer architecture teaching and learning has been developed and refined according to both established and emerging new principles of pedagogy. A methodology for evaluating the simulator's educational value has been identified and successfully applied during scheduled practical tutorial classes in years two and three of a three year undergraduate computing degree. The results are presented both qualitatively and quantitatively and are strongly indicative of the positive pedagogical value offered by the system simulations.

1 Introduction

We have been designing and delivering undergraduate teaching modules on computer architectures and operating systems for the past seven years. In order to support the practical tutorial sessions we developed an integrated set of simulators collectively identified as the system simulator [5]. The simulators conform to modern principles of pedagogy with respect to facilitating student engagement [6] and enhancing student learning experiences [1]. This paper presents examples of practical assignments using the simulations in order to demonstrate the ways in which the above principles are adhered to in the design of the simulations. We also present a brief account of a quantitative method of evaluating the educational value of the simulations and the ways in which the measurements were defined, gathered and analysed.

The integrated simulators demonstrate the essence of modern computer architecture, by capturing the following key aspects of computer technology in one educational software package [9]: the generation of CPU instructions by

Besim Mustafa · Peter Alston
Business School, Faculty of Arts and Sciences, Edge Hill University, Ormskirk, UK
e-mail: {mustafab,alstonp}@edgehill.ac.uk

P. Vittorini et al. (Eds.): International Workshop on Evidence-Based TEL, AISC 152, pp. 1–9.
springerlink.com

assemblers and compilers; the CPU as the processor of the instructions; the operating system (OS) as the facilitator of multiprocessing and multi-threading of the CPU instructions.

The CPU simulator simulates a fictitious, but highly realistic, RISC type CPU architecture. It has a small set of CPU instructions and a register file with selectable number of registers. This simulator incorporates a five-stage pipeline simulator as well as data and instruction cache simulators. The pipeline simulator supports operand forwarding and jump prediction using a jump table. The cache simulators support cache organizations, placement and replacement policies as well as cache coherency policies. Cache simulators display graphical performance statistics.

The OS simulator supports two main aspects of a computer system's resource management: process management and memory management. The process scheduler supports priority-based, pre-emptive and round-robin scheduling. Threads are supported via teaching language constructs enabling students to explore mutual exclusion, process synchronization and deadlock concepts. The virtual memory simulations explore address translation, placement and replacement concepts.

2 Related Work

Modern computer architectures are examples of co-operating systems across tightly coupled interfaces. For example, there are design considerations in hardware in order to assist the operating system and vice versa and highly-optimized compilers include technology in order to assist the hardware and vice versa. It follows that a simulator for computer architecture education should also reflect this interplay and interdependency between computer hardware and software. We have not come across such an educational simulator and therefore we were motivated to develop our own.

Educational simulators often originate from different sources, look and feel different and simulate isolated cases of computer technology in ad hoc and piecemeal fashion using artificial data [3,7,8,11,12]. This makes them unsuitable for studying how different parts of a system fit and work together in educational settings. Moreover, apart from the efforts to evaluate isolated algorithm simulations [7,10], we did not find any references to methodical evaluation of more complex system simulations for their pedagogical value. This paper offers one such methodology.

3 The Teaching and Learning Strategy

We have successfully integrated the system simulator into our modules on computer architecture and operating systems. During each practical tutorial session the students work in small groups. The practical exercises are available online and are

designed to encourage critical thinking and deeper understanding of the theory. Year two students study advanced computer architecture and concentrate on performance issues. They study caches, pipelines and compiler optimizations. Year three students study internals of operating systems and explore scheduling mechanisms, memory management techniques, threads, deadlocks and synchronization.

Each week a one-hour lecture is followed by a two-hour tutorial session. The simulation exercises are completed during each tutorial session and over a semester the simulations cover a broad range of computer architecture topics. At the end of the semester the completed portfolio of exercises are assessed.

4 Sample Simulation Tutorial Examples

In this section we look at examples of practical tutorials taken from *Advanced Computer Architecture* and *Operating Systems* modules in which the students use the integrated simulators as part of their coursework portfolios. All examples below rely on the CPU simulator executing instructions, generated by the inbuilt compiler or manually entered, in the background as in real computer systems.

During the tutorial on cache technology the students investigate different cache types, cache sizes, block and set sizes as well as the placement and replacement policies. They study the advantages and disadvantages of directly-mapped and 2-way/4-way/8-way set-associative cache organizations. They use the inbuilt compiler to create programs that write and read arrays of bytes and run the programs on different sizes of cache on the above cache organizations and record the cache performances. They then comment on their observations. Finally they devise experiments to show the impact of programming style on cache performance. Fig 1 shows the data cache simulator.

The tutorial on CPU pipelines allows the students to switch the CPU pipeline on or off and pipeline optimizations such as operand forwarding and jump prediction can be selected or deselected. The simulator displays colour-coded stages of instructions as they progress through the pipeline, calculates and displays the average clocks per instruction (CPI) and the speed-up factor (SF). The students run programs on various configurations of the pipeline (e.g. hazard bubbles on and off, use NOP instructions to delays, optimizations on and off) and record the CPI and the SF. They study the impact of loop-unrolling and out-of-order instruction optimizations on pipeline performance.

The tutorial on threads uses the inbuilt compiler where the threads are implemented as system calls via a language construct that identifies code in selected subroutines as part of a thread. Calls to these subroutines invoke the OS simulator to create the threads. Students explore the concepts of critical regions, synchronization and counting semaphores for protecting shared global resources. They explore parent/children hierarchies and demonstrate client/server software using multiple threads when processing client requests in separate threads.

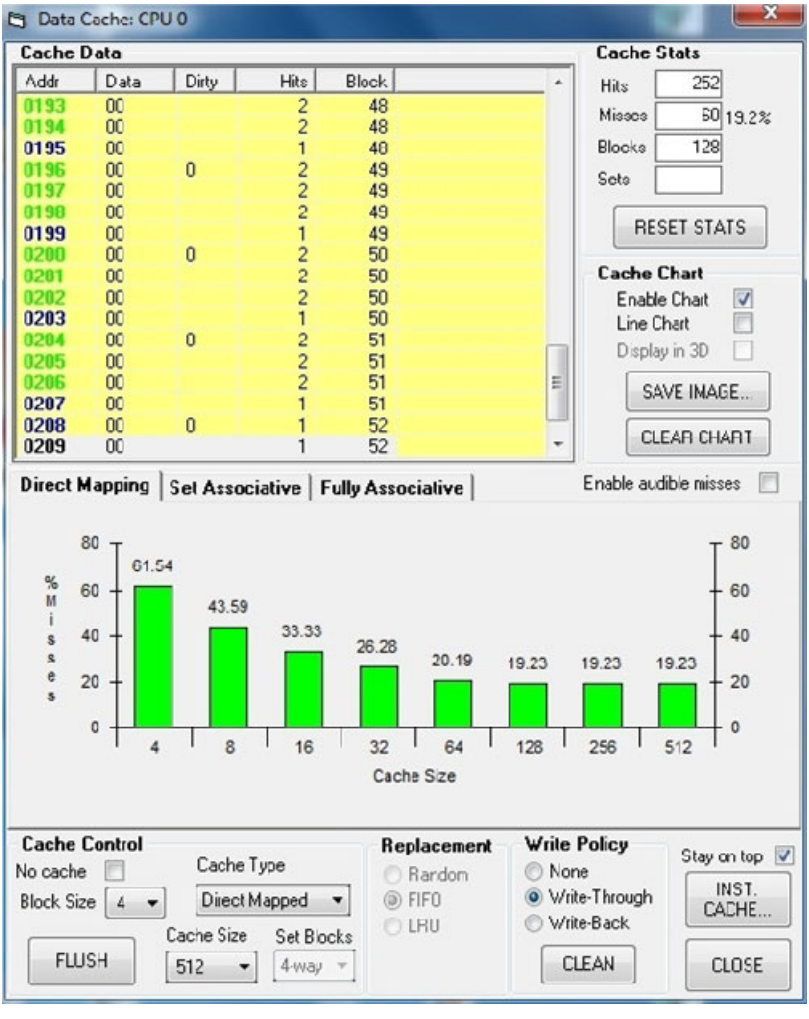

Fig. 1 Data cache simulator window

5 Evaluation Methodology

We used a two-pronged strategy in evaluating the simulations. We first evaluated the simulations by mapping the scope of the simulations to the *Engagement Taxonomy* [6] and to Bloom's *Learning Taxonomy* [1]. The former is developed as a measure of the effectiveness of algorithm visualizations, whilst the latter is based on learning theory and provides a measure of the depth of learning. We next used the quasi-experimental method to gather and analyze the quantitative data.

5.1 Taxonomies of Pedagogy and Qualitative Evaluation

In this section we show the degree of conformance of the simulations to the two taxonomies by mapping various features of the simulations to the individual elements of the taxonomies as a measure of the effectiveness of the simulations. It is interesting to note that there is some common ground between the two mappings shown in Tables 1 and 2. It can be seen that the simulations are able to fully support the requirements of the two taxonomies across all their elements qualitatively indicating that they have the capacity to both gainfully engage and educate.

Table 1 Mapping onto the Engagement Taxonomy

Attributes of Engagement Taxonomy	
View	Can view data held in cache, its address, no of hits against each entry, block numbers, set numbers, etc. Colour coded bar charts display in real-time data misses and hits.
Respond	Responding is mainly guided by the instructions in the exercise sheets; students respond to questions and make notes on their observations; simulations don't require specific responses.
Change	Can change pipeline status: pipeline enabled or disabled; pipeline optimizations selected or deselected; pipeline history recording switched on or off; pipeline history played back or forward and can be reset. Jump table contents can be cleared.
Construct	Students use the inbuilt compiler and the language to construct scenarios where the style of programming can affect the efficient use of the cache by writing assembly code to highlight and demonstrate strengths and weaknesses of cache mappings.
Present	Students can present the implementation of "consumers" and "producers" problem and solution using threads and semaphores. Students construct a simple multi-threaded server program and use it to demonstrate basics of client request handling.

Table 2 Mapping onto Bloom's Taxonomy

Bloom's Levels of Learning Taxonomy	
Knowledge	Knowing of different cache organizations. Awareness of hit rate as a measure of performance. Be able to explain meanings of sets and blocks.
Comprehension	Understanding the use of NOP instructions to compensate for data hazards. Appreciation of the role of compiler optimizations on pipeline performance. Ability to calculate CPI and SF and understand how they are related.
Application	Application of compiler optimization loop-unrolling to increase pipeline performance. Designing test cases to demonstrate different jump instruction's impact on pipeline jump prediction mechanism.
Analysis	Experimenting with set associative mapping using 2-way, 4-way and 8-way sets and comparing relative merits. Students can experiment with write-through and write-back policies and comment on which is best and under what conditions.
Synthesis	Using inbuilt compiler and assembler to create CPU instructions to demonstrate pipeline features and show how software can assist hardware in increasing system performance.
Evaluation	Evaluating the relative merits of set associative cache organization's 2-way to 8-way sets and evaluate the optimum value. Running a program and evaluating graphs of miss rates against cache sizes for all cache organizations.