Proceedings of the 9th International Conference on Applied Informatics
Eger, Hungary, January 29–February 1, 2014. Vol. 1. pp. 239–249
doi: 10.14794/ICAI.9.2014.1.239

New textbooks on parallel architectures, algorithms and programming*

Benedek Nagy[a], Péter Battyányi[a], Norbert Bátfai[a]
Zoltán Gál[b], Tamás Herendi[a], György Kovács[a]

[a]University of Debrecen, Faculty of Informatics
`nbenedek|battyanyi.peter|batfai.norbert|herendi.tamas|kovacs.gyorgy@inf.unideb.hu`

[b]University of Debrecen
`zgal@unideb.hu`

Abstract

In the recent years there were several TÁMOP projects in Hungary to develop new and modern high-level textbooks (in mostly electronic form) in various fields. Starting from January of 2012 there was a two year project entitled "Sokprocesszoros rendszerek a mérnöki gyakorlatban" (Multiprocessor systems in engineering practice) involving three universities, namely, University of Pécs, University of Debrecen and Óbuda University. In the project 23 new materials were written, 10 books by the authors from the University of Debrecen. In this paper we present these new books. The topics of these books cover a wide range from theory of algorithms through on parallel programming technologies, languages till cloud computing technologies. The books are provided by the publisher Typotex and can effectively be used to teach students in various levels in various Universities in Software, System and Computer Engineering, Computer Science, Information Technology and in related disciplines.

Keywords: parallel computing, parallel algorithms, parallel programming, cloud computing, concurrent processes

MSC: 68Q85, 65Y05, 68Q10, 68W10, 68N15, 68N19, 97Q10

*The textbooks are written in the frame of the "Sokprocesszoros rendszerek a mérnöki gyakorlatban" TÁMOP-4.1.2.A/1-11/1-2011-0063 project. The work of the authors is also supported by the TÁMOP-4.2.2.C-11/1/KONV-2012-0001 project. The projects have been supported by the European Union, co-financed by the European Social Fund.

1. Introduction – about the project

In April 2011, supercomputers were installed in Pécs, Debrecen, Szeged and Budapest. Then, for the call TÁMOP-4.1.2.A/1-11/1 to write and develop lecture notes, textbooks and other teaching materials, especially in Mathematics, Science, Engineering and Information Technology, a consortium was founded by University of Pécs (leader of the consortium), University of Debrecen and Óbuda University. The aim of the consortium was to organize a project that results high-level textbooks and teaching materials connected to Multiprocessor Systems in Engineering Practice (Sokprocesszoros rendszerek a mérnöki gyakorlatban). The project was 2-year long, and the University of Debrecen had a budget of approximately 50 million HUF. A wide range of materials are written from theoretic, algorithmic issues, through on mathematical methods to practical and technological ones, like parallel programming languages and services, like cloud computing. All 23 books written in the project were presented in details by the authors at the conference of the project (Harkány, Hungary, 16-18 October, 2013). In this paper, because of the strict page limit we concentrate only on those ten books (on seven topics) that were written by authors from University of Debrecen.

2. The new textbooks developed at the University of Debrecen

In this section we present some details about the seven materials that are written by the staff of the University of Debrecen. We start with the books covering theoretical topics.

2.1. Parallel Approach of Algorithms

This book was written by a cooperation of Tamás Herendi and the project leader, Benedek Nagy in English [7] and in Hungarian (Párhuzamos algoritmusmodellek, [8]). Most of the traditional computing models (Markov algorithm, generative grammars, finite automata) are sequential, such as Neumann type architecture and traditional algorithms. However, the world and our devices are not sequential any more. Desktop computers, laptops, tablets, smartphones may have many processors and several cores in a processor. Furthermore, – particular hardware units serve parallel computing, such as GPUs, FPGAs, etc. They work in networks, in the Internet, etc. The aim of this 200-page long book is to have a theoretical foundation of parallel algorithms by presenting various forms of views and thought of parallelism. We present a variety of models, systems and tools that are able to describe/analyse parallel algorithms/parallel computations. The book contains 60 figures and also 5 animations. The units are closed by questions, exercises and literature.

The first part provides some mathematical background and basics of complexity theory. In addition it presents a Super Turing Machine model that allows changing the number of used tapes during a run in a dynamical way. This concept could be used to define parallel complexity concepts. In Part II, parallel grammar models are shown, such as Indian, Russian and bounded parallel grammars (e.g., scattered context grammars), various L-systems (that were introduced by A. Lindenmayer to model growth of plants), Cooperative Distributive and Parallel Communicating grammar systems. The analysis of distributed systems gives the concepts of parallelism, concurrency and communication. These concepts can be formally defined and analysed in grammar systems. Part III is about parallel automata models including multihead automata and bio-inspired models, e.g., Watson-Crick automata, P automata and cellular automata. In Part IV commutations, traces and trace languages are analysed. Automata models are also shown to accept trace languages. Part V is about Petri nets. They are popular models of concurrent systems. Binary and place-transition nets are detailed, more specific and extended models are also mentioned. Part VI is about parallel programs including some elementary parallel algorithms and some thoughts about parallelization of sequential programs. Finally in Part VII, the two basic forms of the parallelism, the "or-parallelism" and the "and-parallelism" are described.

2.2. Parallel Numerical Methods

This 82-page long book is written by Tamás Herendi, both in Hungarian (Párhuzamos numerikus módszerek, [6]) and in English [5]. The primary goal of this book was to give an overview on parallel algorithmic solutions of particular problems having numeric nature. The first approach was to find and analyze general algorithms, which are useful in the solution in many cases. However, certain tasks may have quite general solution methods so they deserve an overview. The secondary goal was improving the skills in parallel algorithmic thinking. The recently appeared hardwares provide wide deposit of possibilities to increase efficiency so that one cannot neglect them (supercomputers, multicore processors, FPGA, video processors). The book is divided into 12 chapters:

1. Introduction
2. Algorithm Description Models
3. Complexity Theory
4. Basic Algorithms
5. Linear Algebra
6. Fast Fourier Transformation
7. Long Arithmetic
8. Interpolation
9. Iterations
10. Computation of Polynomial Evaluation
11. Monte Carlo Method
12. Random Number Generators

The fist chapter (Introduction) gives a short and general overview on the possibility and need for parallel computation. The second chapter (Algorithmic Description Models) describes some of the algorithm representation methods: parallel pseudo code, parallel flow chart and data flow graphs. The third chapter (Complexity Theory) defines different complexity measures suitable for expressing parallel properties of algorithms. The next chapter (Basic Algorithms) deals with general problems and their parallel solutions. The methods applied here appear at many places in the latter parts. Among others, different sorting algorithms are detailed and compared. In the further chapters certain fields are discussed (linear algebra, Fast Fourier Transformation, long arithmetic, interpolation, iteration, polynomial evaluation, Monte Carlo method and random number generators) with a special attention on their possibility of parallelizing. Three of the chapters should be highlighted:

One of them is the chapter of Fast Fourier Transformation, since it has rather efficient and smart parallel solution. The method based on a strong analysis of the targeted transformation and uses unexpected intermediate results. The next highlighted chapter deals with Monte Carlo Method. Here again strong parallelization can be achieved, but by a completely different approach. The computation stands of totally independent subcomputations, thus without substantial changes in the algorithm, parallel solution can be derived. The third is the chapter of Long Arithmetic, which contains a negative example. It is proven there that the problem of general exponentiation cannot be solved in an efficient way by parallel algorithm.

2.3. Selected Topics in the Theory of Concurrent Processes (with Applications)

The aim of this course material is to give a little insight into the most widespread and fundamental techniques of the mathematical descriptions and modelling of reactive systems. The usual approach to sequential programs is to model the behaviour of a program as a sequence of actions which have the effect of changing the state of the program, the values assigned to the program variables, and at the point of termination the result, as the final state, emerges. Unlike this approach a reactive or concurrent system can be viewed as a system sensitive to changes from stimuli or information from itself or from the environment. In most of the cases, like operating systems, communication protocols, control programs, even termination is not desirable. For a formal treatment of these systems several approaches were proposed, the most prominent ones among them were probably the theory of communicating sequential processes (CSP) of Hoare ([9]) and the calculus of communicating systems (CCS) of Milner ([15]). In this course material we have chosen Milner's approach.

The course material focuses both on the definitions and the key notions in the field of CCS and on the various results concerning the semantical aspects of labelled transition systems. The first chapter defines processes as constituents of labelled transition systems. The next chapter is about process equivalences: it is a nontrivial question whether two processes have the same meaning. If one process is a specification and the other one is an implementation: how do we know that the implementation meets the requirements of the specification? Then Hennessy–Milner logic as a basic tool for process behaviour is introduced. In the fourth chapter another approach of process equivalence is discussed: we present the results of Stirling ([17]) on characterizing process equivalence by two-player games. The next two chapters treat the model-theoretic aspects of the behaviour of transition systems in detail. Timed logics are introduced, which enable us to pose questions relating temporal behaviour of processes. These questions are sought to be answered as questions of formula satisfiability, in our present terminology, as questions of model checking: given a labelled transition system and a formula, is it the case that the transition system, or some of its states satisfy the given formula? The last but one chapter is about processes defined by recursive equations. Finally, the last chapter introduces the modal μ-calculus together with a game semantic tool for model checking (see [17]).

The course material written in English by Péter Battyányi and it consists of more than 150 pages, it contains 31 figures [2]. The material is intended to serve both as readings for a course about the foundations of parallel programming and as a material for self-study. It should be taught at a graduate or more advanced level.

2.4. Parallel Programming in GNU/Linux Environment: SysV IPC, P-threads, OpenMP

This book (only in Hungarian [1]: Párhuzamos programozás GNU/Linux környezetben: SysV IPC, P-szálak, OpenMP) is written in DocBook [19] XML 4.4 directly by Norbert Bátfai. Following Eric Raymond's "Release Early, Release Often" [16] principle, it has already been downloadable since the writing of this book was begun from the author's web page at `http://www.inf.unideb.hu/~nbatfai/konyvek/`. From this URL the book can be found in printable PDF and in some other formats such as browsable HTML and EPUB for ebook readers as well. The number of pages of the book's PDF version is 232 including 85 figures, 8 tables and 19 exercises. The XML source of the book contains more than 150 different DocBook XML 4.4 elements.

The book contains mainly C and C++ examples from following programming areas: P-threads, OpenMP, Qt, CUDA and Hadoop. The quintessence of the book is that a simulation computation was implemented and run in a real supercomputer. To be more precise a simulation example related to the double slit experiment was performed on the HPC facility of the University of Debrecen.

The book can be partially used as lecture notes for the course "High-Level Programming Languages 1" as well as for other courses by the following breakdowns:

- Case studies in C and C++ (PTI, GI M.Sc. lab), the Copenhagen Pascal's triangle experiment.
- Programming in GNU/Linux environment (PTI, GI M.Sc. lecture and lab), computing Pi using the kernel's message queue.
- Java case studies (PTI, GI B.Sc. lab), the Map-Reduce examples.
- High-Level Programming Languages 2 (PTI, MI, GI B.Sc. lecture and lab), the CUDA examples.

2.5. OpenCL

Although circuit technology approaches the to the physical limits, Moore's law and Kurtzweil's singularity theory seem to be still valid: nowadays Central Processing Units (CPU), Graphical Processing Units (GPU) and Field Programmable Gate Arrays (FPGA) have the same computing power as clusters of CPUs ten years ago. In fact, the electrical power of a computing device is proportional to the square of its clock rate frequency. Thus, the computing power can not be increased by increasing the clock rate, because the maintenance becomes too expensive. The high computing power of nowadays devices is a result of increasing inner complexity. Particularly, CPU and GPU devices have, while FPGA cards can be configured to have many processor cores. In the middle of the 2000s, the programming of these devices highly differed. There were conventional techniques for the parallel programming of CPUs, vendor dependent toolkits for the programming of GPUs and also vendor dependent, but low level languages for FPGA cards. The similar computing power of these devices gave birth to the need of a unified framework enabling device independent high performance computing by hiding the properties of the physical device. This unified framework standardized and published in the end of 2008 is called the Open Computing Language (OpenCL).

OpenCL specifies an abstract device (OpenCL device) to hide the details of real computing devices, a programming language (OpenCL C) used to write programs for OpenCL devices and a set of functions enabling the management and execution of devices and OpenCL C programs. The success of OpenCL is due to its support by the largest vendors and software companies, including but not limited to Adobe, Apple, Nvidia, AMD, Intel and Altera. OpenCL support has been already built in many desktop applications, like MatLab, Photoshop, WinZip and VLC, enabling the utilization of the high computing power of the available CPU, GPU and FPGA devices with the very same program. Furthermore, many developments are going on to extend the applications of OpenCL: enabling the use of OpenCL in browsers (WebCL); mobile devices, etc.

However, an efficient OpenCL program is more than a working code calling OpenCL functions. In order to utilize the resources of the real computing devices efficiently, one needs a deep understanding of the pretty complex OpenCL device and the way it is adapted for CPUs, GPUs or FPGA cards. The best way to learn this knowledge is studying case studies and comparing various OpenCL implementations of the same algorithm in terms of execution time.

OpenCL technology tends to become a determinate technology in high performance computing. In order to make computer science students fit the requirements of industry and science, the education of OpenCL seems to be a very important task. The concept of the book called OpenCL is to go through the elements of the OpenCL specification and demonstrate the use of the technology by case studies covering the main fields of OpenCL applications.

In numbers, the book has 360 pages. The OpenCL technology is introduced in 8 chapters covering approximately 210 pages and the case studies (matrix multiplication, convolutional filtering, histogram computation, discrete Fourier-transform and particle simulation) are discussed in 5 further chapters covering 120 pages. The technology and the case studies are illustrated with dozens of sample codes and charts of execution times. The appendices introduce the reader to the basics of cmake, R and the reading and writing of PGM and PNG images. In order to aid the testing of students, more than one hundred exercises are given at the ends of the chapters. The textbook is authored by György Kovács; it is available in Hungarian [12] and English languages [11], and according to the best of our knowledge, this is the first book about OpenCL in Hungarian [12].

2.6. Parallel programming tools and combined applications

Many types of parallelism are present in computer science from the very early years. For example, the connection of individual computers enabled the distributed solution of problems with high computing demands; time-sharing operating systems enable the illusion of parallel execution in the presence of one CPU. When the first general purpose two-core CPU for desktop computers was introduced in 2005, new parallel programming technologies appeared, and the development of GPUs also introduced parallelism in GPU computing. We probably won't have any more products without multicore processors and Internet connection to other, similar devices, thus, distributed computation and parallelism is a keyword in nowadays computer science. However, the term parallel covers a wide range of technologies, approaches and applications in computer science.

Due to the wide variety of technologies, the education of parallel and distributed computing techniques is a hard task. Studying one technology is not enough, because it influences the students to work with that technology only. On the other hand, there is no way to organize a dozen courses on the parallel programming technologies used in nowadays computing, although the heterogeneity of hardware environments usually requires the combined application of technologies.

We have decided to work out a textbook covering the most popular approaches of parallel and distributed computing. The book describes five technologies in details, particularly, auto-vectorization, OpenMP, P-threads, OpenMPI and OpenCL and illustrates their applications on various parallel implementations of the matched filtering operation of digital image processing.

In the first, introductory chapter the various interpretations of parallelism are described and the basic concepts and laws of parallel programming are introduced. The second chapter aids the reader to set up the software environment for parallel computing. In the third chapter, an overview is given on the development and analysis of parallel programs. The issues of communication, coarse and fine parallelism, load balancing, dynamic and static scheduling are discussed and the use of the software gprof is also presented to identify the hot points (computationally intensive points) in the source code of an application. The use of gprof is illustrated by the analysis of the source code of matched filtering. Although automated parallelization is an unsolved and hard task in the theory and practice of parallel computing, some features are already available as part of the GNU compiler collection. Accordingly, the next chapter describes the parallel programming technology, called auto-vectorization, enabling the automated utilization of vector-processor features in modern CPUs. Although automated parallelization is hard, there are very popular semi-automated solutions using compiler directives to indicate those parts of the code that can be executed in parallel. The sixth chapter describes one representative compiler directive based parallelization technique, called OpenMP. The next chapter describes shortly the conventional P-threads technology used to write parallel programs for single or many core desktop computers. The most popular technologies of distributed computing are based on the MPI standard and we have chosen the implementation OpenMPI for overview in the eighth chapter. The OpenCL technology used for GPU programming is introduced in the ninth chapter. All of the technologies are illustrated by sample programs and various parallel implementations of matched filtering. In the next chapter the combined applications of these technologies are discussed involving the concept of thread-safety. In the final chapter a short overview is given on further parallel programming technologies. Since the book covers the most popular approaches of parallel and distributed programming and gives an overview of available technologies, the authors hope that it helps the reader to navigate in the world of parallelism in computer science.

The book is written by György Kovács; it has 12 chapters including appendices and covering 323 pages, available in Hungarian (Párhuzamos programozási eszközök és összetett alkalmazásaik, [13]). The parallel programming techniques are illustrated by dozens of sample codes, 11 figures and 14 charts of execution times. The book contains more than one hundred exercises related to the topics and sample codes discussed in the chapters.

2.7. Cloud computing architectures and services

Based on the classical evaluation methods applied in the second half of the twenties century the fifth generation of the computers was blocked by the lack of spectacular evolution in the last decades. The prognoses of the new computer generation suggested not only higher computation capacity, smaller physical size, higher communication rate, cheaper price but software based applications producing significant change of the whole human activity. The powerful progress based on the Internet generated software development and communication techniques caused all-inclusive spreading of the electronic services. The effect of reforming economic and business processes had growing speed of the society processes, as well. This role had persistent feedback influence to the ICT services. The competitive effect of the business draw up critics addressed to the application developers and service providers filtering out ICT systems with low efficiency and relatively high price.

The Cloud Computing book is prepared as lecture note for the students but the overall synthesis of the topic makes it useful not only for specialists in informatics, but for people involved in economics, or engineering, as well. Leaders of such fields can find important information for the right direction of the computer fabrication and software development trade named Cloud Computing (abbreviated C2) informatics system based on the effect of business processes. This document analyses deeply the architecture and service aspects related to the C2 subject together with the necessary mechanisms and technologies. Because of the epoch-making characteristics the C2 can be considered the fifth generation network-computer system model. The reader is introduced step by step from the field of classical IT services and systems to the complex world of infocommunication solutions playing growing role in the near future. Because of the hot topic the book relays the C2 concept of the dominant ICT companies. The lecture note creates a structure of the C2 notions and necessary mechanisms to be easily understood by the readers. The author is engineer in electricity and computers with over twenty years of experience in practice and presents in this book the evolution, the actual structure and the services of the C2 together with the possible evolution direction and the environmental effect, as well.

The book has eight chapters. The preface chapter discusses the basic terms of the computation to give knowledge enough for non-ICT professionals, too. Each chapter at the end includes tens of questions making possible self-testing the C2 knowledge of the reader. The core chapters discuss issues of the current ICT services (ICT power consumption efficiency, Service Oriented Architecture) [14], evolution of the C2 (trends, virtualization, security), C2 services concept and their industry support (environment protection, provision elements of C2: strategy and development, cloud data, cloud storage, cloud security, cloud infrastructure) [3], functional elements of the C2 services, expenditure models of the C2 [18]. The XaaS (X as a service) models are presented, where X can be infrastructure, platform, and software or business process. In this context C2 models of the IBM, Microsoft, Apple, Google, Amazon, Oracle and Cisco companies are presented [10]. In the last two chapters expected technological and societal effects of the C2 are discussed. The book ends with the future and possible development direction of the C2, including the next generation C2 architecture and the C2 in high performance computing environment.

This 172-page long book is written by Zoltán Gál, in Hungarian with title Cloud computing architektúrák és szolgáltatások [4].

3. Topics of materials written in the project

In this section, we shortly list the topics of all the textbooks including the ones written by authors from University of Pécs and from Óbuda University in the project. The materials can be grouped into three modules:

Module I: Parallel algorithms (in engineering practice)

- Parallel numerical methods (Tamás Herendi)
- Parallel approach of algorithms (Tamás Herendi, Benedek Nagy)
- Parallel algorithms of numerical linear algebra (Aurél Galántai; Óbuda)
- Discrete optimization in parallel environment (Sándor Szabó; Pécs)

- Preprocessing for parallel numerical simulations (Péter Iványi, János Radó; Pécs)
- Parallel models and algorithms of computer vision and image processing (András Rövid; Óbuda)

Module II: Parallel programming tools and methods (in engineering practice)

- Parallel programming in GNU/Linux environment: SysV IPC, P-threads, OpenMP (Norbert Bátfai)
- MPI programming and exercises (Bogdán Zavalnij, Géza Várady; Pécs)
- OpenCL (György Kovács)
- Selected topics in the theory of concurrent processes with applications (Péter Battyányi)
- Parallel programming tools and combined applications (György Kovács)

Module III: Parallel applications (in engineering practice)

- Cloud computing architectures and services (Zoltán Gál)
- Applications of cloud computing environments in engineering practice (Tamás Schubert; Óbuda)
- GPGPUs and their programming (Dezső Sima, Sándor Szénási; Óbuda)
- Medical image processing for diagnostics in parallel and distributed systems (Miklós Kozlovszky; Óbuda)
- Embedded systems in engineering practice (András Molnár; Óbuda)
- Multicore processors (Dezső Sima; Óbuda)

As one can observe, the materials written by authors from University of Pécs and Óbuda University are complementing the textbooks written by the authors from University of Debrecen, to cover a wide range of parallel computing.

4. Conclusion

We believe that the new textbooks (including the ones written by the consortium partner universities) will be used in several universities and colleges helping both students and teachers in our higher education to have up-to-date knowledge in the field of parallel computing and multiprocessor systems.

References

[1] BÁTFAI, N., Párhuzamos programozás GNU/Linux környezetben: SysV IPC, P-szálak, OpenMP, Typotex, (2013)

[2] BATTYÁNYI, P., Selected Topics in the Theory of Concurrent Processes with Applications, Typotex, (2013)

[3] de CHAVES, S.A., et al., Customer Security Concerns in Cloud Computing. ICN 2011: The Tenth International Conference on Networks, (2011)

[4] GÁL, Z., Cloud computing architektúrák és szolgáltatások, Typotex, (2013)

[5] HERENDI, T., Parallel numerical methods, Typotex, (2013)

[6] HERENDI, T., Párhuzamos numerikus módszerek, Typotex, (2013)

[7] HERENDI, T., NAGY, B., Parallel approach of algorithms, Typotex, (2013)

[8] HERENDI, T., NAGY, B., Párhuzamos algoritmusmodellek, Typotex, (2013)

[9] HOARE, C. A. R., Communicating Sequential Processes, Prentice Hall International, (1985)

[10] JONES, M.T., Anatomy of a Cloud Storage Infrastructure Models, Features, and Internals, Trademarks, IBM Corporation, (2010)

[11] KOVÁCS, Gy., OpenCL, Typotex, (2013)

[12] KOVÁCS, Gy., OpenCL, (in Hungarian), Typotex, (2013)

[13] KOVÁCS, Gy., Párhuzamos programozási eszközök és összetett alkalmazásaik, Typotex, (2013)

[14] LINTHICUM, D.S., Cloud Computing and SOA Convergence in Your Enterprise – A Step-by-Step Guide, Addison-Wesley Information Technology Series, (2009)

[15] MILNER, R., Communication and Concurrency, Prentice Hall International, (1989)

[16] RAYMOND, E.S., The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary,ISBN: 0596001088, O'Reilly, (2001)

[17] STIRLING, C., Modal and Temporal Properties of Processes, Springer Verlag, (2001)

[18] VELTE, A.T., VELTE, T.J., ELSENPETER, R., Cloud Computing: A Practical Approach, ISBN: 978-0-07-162695-8, (2010)

[19] WALSH, N., MUELLNER, L., DocBook: The Definitive Guide, ISBN: 1565925807, O'Reilly, (1999)