

Preface

Innovations in hardware architecture, like hyper-threading or multicore processors, make parallel computing resources available for inexpensive desktop computers. However, the use of these innovations requires parallel programming techniques. In a few years, many standard software products will be based on concepts of parallel programming to use the hardware resources of future multicore processors efficiently. Thus, the need for parallel programming will extend to all areas of software development. The application area will be much larger than the area of scientific computing, which used to be the main area for parallel computing for a many years. The expansion of the application area for parallel computing will lead to an enormous need for software developers with parallel programming skills. Some chip manufacturers already demand to include parallel programming as a standard course in computer science curricula. A more recent trend is the use of Graphics Processing Units (GPUs) comprising several hundreds of cores to execute compute-intensive non-graphics applications.

This book takes up the new development in processor architecture by giving a detailed description of important parallel programming techniques that are necessary for developing efficient programs for multicore processors as well as for parallel cluster systems or supercomputers. Both shared and distributed address space architectures are covered. The main goal of the book is to present parallel programming techniques that can be used in many situations for many application areas and to enable the reader to develop correct and efficient parallel programs. Many example programs and exercises are provided to support this goal and to show how the techniques can be applied to further applications. The book can be used both a textbook for students and a reference book for professionals. The material of the book has been used for courses in parallel programming at different universities for many years.

This second edition of the English book on parallel programming is an updated and revised version based on the third edition of the German version of this book from 2012. The two earlier German editions appeared in 2000 and 2007, respectively. The update of this new English edition includes a new chapter on

general purpose GPUs and the corresponding programming techniques. The remaining chapters have been revised carefully. Especially the chapter on the architecture of parallel systems has been updated considerably putting a larger emphasis on the architecture of multicore systems and adding new material on the recent development in computer architecture.

The content of the book comprises of three main parts, covering all areas of parallel computing: the architecture of parallel systems, parallel programming models and environments, and the implementation of efficient application algorithms. The emphasis lies on parallel programming techniques needed for different architectures.

The first part contains an overview of the architecture of parallel systems, including cache and memory organization, interconnection networks, routing and switching techniques, as well as technologies that are relevant for modern and future multicore processors.

The second part presents parallel programming models, performance models, and parallel programming environments for message passing and shared memory models, including MPI, Pthreads, Java threads, and OpenMP. For each of these parallel programming environments, the book gives basic concepts as well as more advanced programming methods and enables the reader to write and run semantically correct and efficient parallel programs. Parallel design patterns like pipelining, client-server, or task pools are presented for different environments to illustrate parallel programming techniques and to facilitate the implementation of efficient parallel programs for a wide variety of application areas. Performance models and techniques for runtime analysis are described in detail, as they are a prerequisite for achieving efficiency and high performance. A new chapter gives a detailed description of the architecture of GPUs and also contains an introduction into programming approaches for general purpose GPUs concentrating on CUDA and OpenCL. Programming examples are provided to demonstrate the use of the specific programming techniques introduced.

The third part applies the programming techniques from the second part to representative algorithms from scientific computing. The emphasis lies on basic methods for solving linear equation systems, which play an important role for many scientific simulations. The focus of the presentation lies on the analysis of the algorithmic structure of the different algorithms, which is the basis for a parallelization, and not on mathematical properties of the solution methods. For each algorithm, the book discusses different parallelization variants, using different methods and strategies.

Many colleagues and students have helped to improve the quality of this book. We would like to thank all of them for their help and constructive criticisms. For numerous corrections we would like to thank Jörg Dümmler, Marvin Ferber, Michael Hofmann, Ralf Hoffmann, Sascha Hunold, Matthias Korch, Raphael Kunis, Jens Lang, John O'Donnell, Andreas Prell, Carsten Scholtes, Michael

Schwind, and Jesper Träff. Many thanks to Matthias Korch, Carsten Scholtes and Michael Schwind for their help with the exercises. We thank Monika Glaser and Luise Steinbach for their help and support with the \LaTeX typesetting of the book. We also thank all the people who have been involved in the writing of the three German versions of this book. It has been a pleasure working with Springer-Verlag in the development of this book. We especially thank Ralf Gerstner for his continuous support.

Bayreuth, March 2013
Chemnitz, March 2013

Thomas Rauber
Gudula Rünger



<http://www.springer.com/978-3-642-37800-3>

Parallel Programming
for Multicore and Cluster Systems

Rauber, Th.; Rünger, G.

2013, XIII, 516 p., Hardcover

ISBN: 978-3-642-37800-3