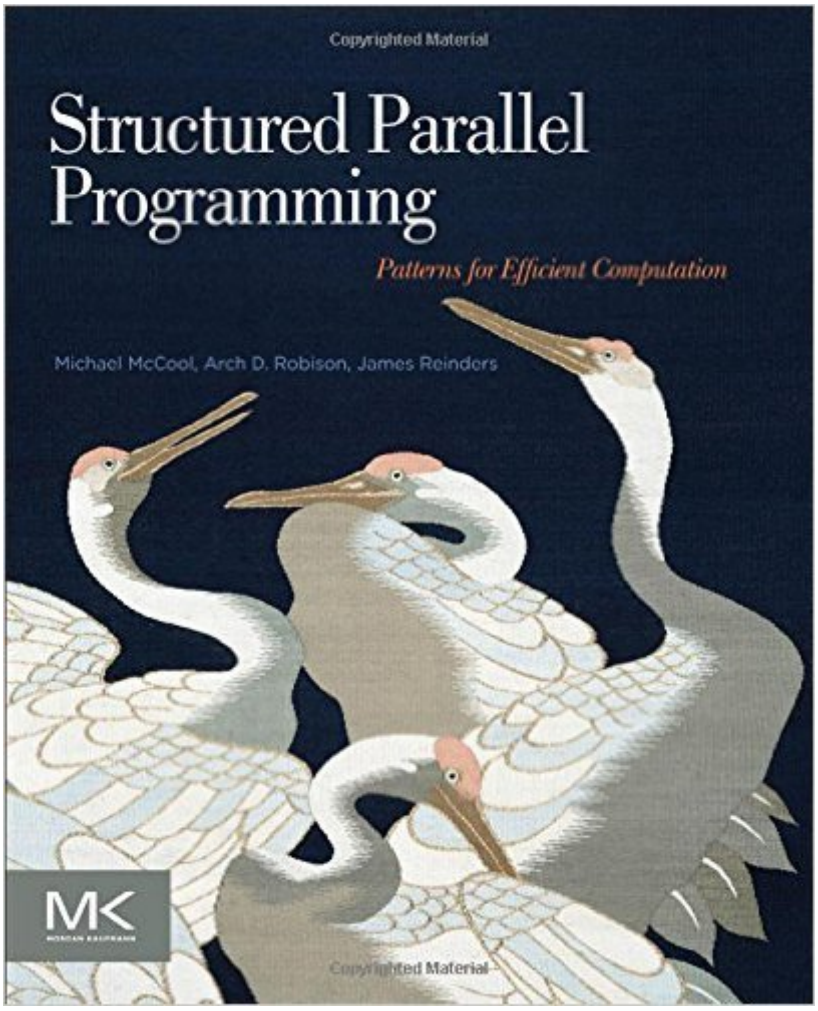
Structured Parallel Programming: Patterns For Efficient Computation



Synopsis

Programming is now parallel programming. Much as structured programming revolutionized traditional serial programming decades ago, a new kind of structured programming, based on patterns, is relevant to parallel programming today. Parallel computing experts and industry insiders Michael McCool, Arch Robison, and James Reinders describe how to design and implement maintainable and efficient parallel algorithms using a pattern-based approach. They present both theory and practice, and give detailed concrete examples using multiple programming models. Examples are primarily given using two of the most popular and cutting edge programming models for parallel programming: Threading Building Blocks, and Cilk Plus. These architecture-independent models enable easy integration into existing applications, preserve investments in existing code, and speed the development of parallel applications. Examples from realistic contexts illustrate patterns and themes in parallel algorithm design that are widely applicable regardless of implementation technology.The patterns-based approach offers structure and insight that developers can apply to a variety of parallel programming modelsDevelops a composable, structured, scalable, and machine-independent approach to parallel computingIncludes detailed examples in both Cilk Plus and the latest Threading Building Blocks, which support a wide variety of computers

Book Information

Paperback: 432 pages
Publisher: Morgan Kaufmann; 1 edition (July 9, 2012)
Language: English
ISBN-10: 0124159931
ISBN-13: 978-0124159938
Product Dimensions: 7.5 x 1 x 9.2 inches
Shipping Weight: 2 pounds (View shipping rates and policies)
Average Customer Review: 4.7 out of 5 starsÂ Â See all reviewsÂ (6 customer reviews)
Best Sellers Rank: #232,464 in Books (See Top 100 in Books) #22 inÂ Books > Computers & Technology > Programming > Parallel Programming #70 inÂ Books > Textbooks > Computer Science > Algorithms #129 inÂ Books > Computers & Technology > Hardware & DIY > Design & Architecture

Customer Reviews

I’ve bought the eBook edition of this book so I can only comment on that. The language used in the

book is easy and fluent enough to get you going on even over passages that you don’t immediately grasp. The authors take great pain and a lot of paper to introduce the history of parallel processing, the actual architectures of today’s computers, today’s compilers, how processors manage memory, use vectorization, switch contexts, lose time waiting for hard-disks or system memory and so on and so forth. All this is very thorough and interesting if you didn’t know any of it before - but then you wouldn’t just throw yourself into parallel programming right now, wouldn’t you? ;) Since at least James Reinders is an employee of intel they do only talk about intel products and CPU’s. They find very elegant ways to justify that decision but it really all just comes down to marketing.The content and samples are well explained and give you enough practical ammunition at hand to have a start using the latest tools and libraries on the market. The emphasis on ThreadingBuildingBlocks (TBB) and CilkPlus, both from intel, is not per se bad since they are both FOSS and very useful. The only setback is if you don’t actually use an intel compiler suite. Because as of now only TBB can be used on every modern compiler. For CilkPlus only intels compilers and a specially customized version of gcc 4.8 (don’t know where you can get it though) offer support.Since I bought the eBook version I was very disappointed at the quality of the conversion. Apart from the occasional spelling mistakes I found that all the graphics/diagrams and code samples in the book are included as pixel pictures with a low resolution and low contrast. That makes it impossible to read it on a 6" ePaper device! And it is still unpleasant on my PC - add to this that no code from the book can be searched nor copied to the clipboard!So why am I still giving it 4 stars? Because the book has a lot of good info’s, indispensable knowledge and how to’s and it was made by intel folks who know their s*** when it comes to use processors to their fullest. If you are patient enough to digest the long intro and can live with the flaws of the eBook conversion then you have a book that is by those named attributes inevitable as of today! :D But I sure hope they give us an update of it soon that is actually optimized to read on an eReader! Pliiiiiis!!

The specific code examples are there if you need them (C and C++ libraries). But what is most important is that it lays down a solid framework for thinking about how concurrent/vectorized code needs to be written. I haven’t seen Big-O notation anywhere else applied to measuring how parallelizable an algorithm is, or how much power it will consume. What is really interesting is the set of primitives dealing with maximizing vectorization and concurrency (ie: scatter, gather, map, etc.).These ideas are applicable to everything from writing for Hadoop clusters to signal processing and statistics code. It makes it really clear what is required of a modern programming language that might make optimal parallel code.

Are you a parallel programmer? If you are, then this book is for you! Authors Michael McCool, James Reinders and Arch Robison, have done an outstanding job of writing a book that illustrates the essential strategies that are needed for writing efficient, scalable programs that use a set of patterns.McCool, Reinders and Robinson, begin by discussing why it is necessary to Think Parallel; and, to present recent hardware trends that have lead to the need for explicit parallel programming. In addition, the authors cover background material applicable to most forms of parallel programming, including a review of relevant computer architecture and performance analysis topics. Next, they focus on patterns that lead to well-structured, maintainable, and efficient programs. The authors then, introduce the map pattern, which describes embarrassing parallelism: Parallel computation on a set of completely independent operations. They continue by discussing the collective reduce, scan patterns and various options for their implementation; and, then give some simple examples of their use. In addition, the authors present data reorganization patterns and discuss important issues around data layout. Next, they discuss a special case of the map pattern, the stencil pattern, which has a regular data access pattern. The authors then, describe the fork-join pattern and give several examples, including its use to implement other patterns, such as map, reduce, recurrence, and scan. They continue by covering a simple pipeline model embodied in the TBB parallel_pipeline template. In addition, the authors discuss the reflection seismology survey, which consists of collecting data in the field and processing it to generate a subsurface image. Next, they describe the k-means algorithm, which is a way of finding clusters in a dataset. The authors then, show you how the bzip2 program partitions a file into blocks and compresses each one separately. They continue by showing you how in a merge sort, a serial merge takes time 0(N). In addition, the authors show you how the sample sort example demonstrates a partitioning-based sort that overcomes the scaling limitation of Quicksort; which arose, because Quicksort's partitioning operation is serial. Finally, they give a brief introduction to the art of using Fortran-oriented BLAS routines in C/C++.This most excellent book supports teaching parallel programming in any programming class using C or C++, or as a focused topic in a semester-long class. Perhaps more importantly, this great book purposely approaches parallel programming from a programmer's point of view without relying on an overly detailed examination or prior knowledge of parallel computer architecture.

Structured Parallel Programming: Patterns for Efficient Computation Using OpenMP: Portable Shared Memory Parallel Programming (Scientific and Engineering Computation) Using MPI - 2nd

Edition: Portable Parallel Programming with the Message Passing Interface (Scientific and Engineering Computation) Parallel Programming with MicrosoftÂ® .NET: Design Patterns for Decomposition and Coordination on Multicore Architectures (Patterns & Practices) Parallel Programming: Techniques and Applications Using Networked Workstations and Parallel Computers (2nd Edition) Parallel Programming with Intel Parallel Studio XE ADA Programming Success In A Day: Beginner's guide to fast, easy and efficient learning of ADA programming Perl Programming Success in a Day: Beginners Guide to Fast, Easy, and Efficient Learning of Perl Programming Prolog Programming Success in a Day: Beginners Guide to Fast, Easy and Efficient Learning of Prolog Programming Prolog Programming Success in a Day: Beginner's Guide to Fast, Easy, and Efficient Learning of Prolog Programming RPG Programming success in a day: Beginners guide to fast, easy and efficient learning of RPG programming XML Programming Success in a Day: Beginner?s Guide to Fast, Easy, and Efficient Learning of XML Programming Parallel Scientific Computing in C++ and MPI: A Seamless Approach to Parallel Algorithms and their Implementation Introduction to Parallel Computing: Design and Analysis of Parallel Algorithms Short Stories in Spanish: New Penguin Parallel Text (New Penguin Parallel Texts) (Spanish and English Edition) Learn German: Parallel Text - Easy, Funny Stories (German - English) - Bilingual (Learning German with Parallel Text Book 1) Learn German III: Parallel Text - Easy Stories (German - English) Bilingual - Dual Language (Learning German with Parallel Text 3) (German Edition) Jackson Structured Programming: A Practical Method of Program Design Fortran 77: Featuring Structured Programming (3rd Edition) RPG II and RPG III Structured Programming