

1. Record Nr.	UNINA9910462474603321
Autore	McCool Michael
Titolo	Structured parallel programming [[electronic resource] ] : patterns for efficient computation // Michael McCool, Arch D. Robison, James Reinders
Pubbl/distr/stampa	Amsterdam ; ; Boston, Mass., : Elsevier/Morgan Kaufmann, 2012
ISBN	1-280-77921-7 9786613689603 0-12-391443-4
Edizione	[1st edition]
Descrizione fisica	1 online resource (433 p.)
Altri autori (Persone)	RobisonArch D ReindersJames
Disciplina	005.1 005.275
Soggetti	Parallel programming (Computer science) Structured programming Electronic books.
Lingua di pubblicazione	Inglese
Formato	Materiale a stampa
Livello bibliografico	Monografia
Note generali	Description based upon print version of record.
Nota di bibliografia	Includes bibliographical references and index.
Nota di contenuto	Front Cover; Structured Parallel Programming: Patterns for Efficient Computation; Copyright; Table of Contents; Listings; Preface; Preliminaries; 1 Introduction; 1.1 Think Parallel; 1.2 Performance; 1.3 Motivation: Pervasive Parallelism; 1.3.1 Hardware Trends Encouraging Parallelism; 1.3.2 Observed Historical Trends in Parallelism; 1.3.3 Need for Explicit Parallel Programming; 1.4 Structured Pattern-Based Programming; 1.5 Parallel Programming Models; 1.5.1 Desired Properties; 1.5.2 Abstractions Instead of Mechanisms; 1.5.3 Expression of Regular Data Parallelism; 1.5.4 Composability 1.5.5 Portability of Functionality1.5.6 Performance Portability; 1.5.7 Safety, Determinism, and Maintainability; 1.5.8 Overview of Programming Models Used; Cilk Plus; Threading Building Blocks (TBB); OpenMP; Array Building Blocks (ArBB); OpenCL; 1.5.9 When to Use Which Model?; 1.6 Organization of this Book; 1.7 Summary; 2 Background; 2.1 Vocabulary and Notation; 2.2 Strategies; 2.3 Mechanisms; 2.4 Machine Models; 2.4.1 Machine Model; Instruction

Parallelism; Memory Hierarchy; Virtual Memory; Multiprocessor Systems; Attached Devices; 2.4.2 Key Features for Performance; Data Locality

Parallel Slack2.4.3 Flynn's Characterization; 2.4.4 Evolution; 2.5 Performance Theory; 2.5.1 Latency and Throughput; 2.5.2 Speedup, Efficiency, and Scalability; 2.5.3 Power; 2.5.4 Amdahl's Law; 2.5.5 Gustafson-Barsis' Law; 2.5.6 Work-Span Model; 2.5.7 Asymptotic Complexity; 2.5.8 Asymptotic Speedup and Efficiency; 2.5.9 Little's Formula; 2.6 Pitfalls; 2.6.1 Race Conditions; 2.6.2 Mutual Exclusion and Locks; 2.6.3 Deadlock; 2.6.4 Strangled Scaling; 2.6.5 Lack of Locality; 2.6.6 Load Imbalance; 2.6.7 Overhead; 2.7 Summary; I Patterns; 3 Patterns; 3.1 Nesting Pattern

3.2 Structured Serial Control Flow Patterns3.2.1 Sequence; 3.2.2 Selection; 3.2.3 Iteration; 3.2.4 Recursion; 3.3 Parallel Control Patterns; 3.3.1 Fork-Join; 3.3.2 Map; 3.3.3 Stencil; 3.3.4 Reduction; 3.3.5 Scan; 3.3.6 Recurrence; 3.4 Serial Data Management Patterns; 3.4.1 Random Read and Write; 3.4.2 Stack Allocation; 3.4.3 Heap Allocation; 3.4.4 Closures; 3.4.5 Objects; 3.5 Parallel Data Management Patterns; 3.5.1 Pack; 3.5.2 Pipeline; 3.5.3 Geometric Decomposition; 3.5.4 Gather; 3.5.5 Scatter; 3.6 Other Parallel Patterns; 3.6.1 Superscalar Sequences; 3.6.2 Futures  
3.6.3 Speculative Selection3.6.4 Workpile; 3.6.5 Search; 3.6.6 Segmentation; 3.6.7 Expand; 3.6.8 Category Reduction; 3.6.9 Term Graph Rewriting; 3.7 Non-Deterministic Patterns; 3.7.1 Branch and Bound; 3.7.2 Transactions; 3.8 Programming Model Support for Patterns; 3.8.1 Cilk Plus; Nesting, Recursion, Fork-Join; Reduction; Map, Workpile; Scatter, Gather; 3.8.2 Threading Building Blocks; Nesting, Recursion, Fork-Join; Map; Workpile; Reduction; Scan; Pipeline; Speculative Selection, Branch and Bound; 3.8.3 OpenMP; Map, Workpile; Reduction; Fork-Join  
Stencil, Geometric Decomposition, Gather, Scatter

---

## Sommario/riassunto

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

---