1. Record Nr. UNISA996464547403316

Titolo	Languages and compilers for parallel computing : 34th international workshop, LCPC 2021, Newark, DE, USA, October 13-14, 2021 : revised selected papers / / Xiaoming Li and Sunita Chandrasekaran (editors)
Pubbl/distr/stampa	Cham, Switzerland : , : Springer, , [2022] ©2022
ISBN	3-030-99372-8
Descrizione fisica	1 online resource (159 pages)
Collana	Lecture notes in computer science ; ; Volume 13181
Disciplina	004.35
Soggetti	Parallel processing (Electronic computers) Parallel programming (Computer science) Compilers (Computer programs)
Lingua di pubblicazione	Inglese
Formato	Materiale a stampa
Livello bibliografico	Monografia
Note generali	Includes index.
Nota di contenuto	Intro -- Preface -- Organization -- Contents -- Compiler -- Locality-Based Optimizations in the Chapel Compiler -- 1 Introduction -- 2 Chapel Background -- 2.1 Distributed Arrays -- 2.2 Forall Loops -- 3 Compiler Analysis and Optimizations -- 3.1 Automatic Local Access -- 3.2 Automatic Aggregation -- 4 Results -- 5 Future Work -- 6 Related Work -- 7 Conclusion -- References -- iCetus: A Semi-automatic Parallel Programming Assistant -- 1 Introduction -- 2 Rationale for the iCetus Interactive Parallelizer and Tool Features -- 2.1 Automatic Parallelization in Cetus -- 2.2 The Opportunity of Interactive Parallelization -- 2.3 iCetus Features -- 2.4 Limitations of the Current Version of iCetus -- 3 iCetus System Overview -- 4 Evaluation -- 4.1 Importance and Usefulness of Existing iCetus Features -- 4.2 Importance and Usefulness of our Proposed iCetus Features -- 4.3 Requested Features for iCetus -- 5 Related Work -- 6 Conclusion -- References -- Hybrid Register Allocation with Spill Cost and Pattern Guided Optimization -- 1 Introduction -- 2 Background and Challenges -- 3 Preliminary Analysis -- 4 Design and Implementation -- 4.1 Code Pattern Recognizer -- 4.2 Spill Cost Tracking Mechanism -- 4.3 Putting it all Together: Cost-Guided Allocation Optimizer -- 5 Methodology -- 6 Evaluation Result -- 6.1 Benchmark Performance --

6.2 Sensitivity Study -- 6.3 Compilation Overhead -- 7 Related Work -- 8 Conclusion and Future Work -- References -- Performance Evaluation of OSCAR Multi-target Automatic Parallelizing Compiler on Intel, AMD, Arm and RISC-V Multicores -- 1 Introduction -- 2 The OSCAR Automatic Parallelizing Compiler -- 3 Investigated Multicore Architectures -- 4 Benchmark Programs -- 5 Compile Flow -- 6 Performance of OSCAR Compiler-Parallelized Programs -- 6.1 OSCAR Compiled Benchmark Performance on Intel x86.
6.2 OSCAR Compiled Benchmark Performance on AMD X86 -- 6.3 OSCAR Compiled Benchmark Performance on Arm -- 6.4 OSCAR Compiled Benchmark Performance on RISC-V -- 7 Conclusion -- References -- Accelerators -- LC-MEMENTO: A Memory Model for Accelerated Architectures -- 1 Introduction -- 2 Background -- 2.1 Memory Consistency Models -- 2.2 The Abstract Runtime System: ARTS -- 2.3 NVIDIA CUDA Programming and Execution Environment -- 3 LC-MEMENTO Design and Implementation -- 3.1 Asynchronous Runtime Scheduler for Accelerators -- 3.2 Memory Models for Accelerators -- 4 Evaluation -- 4.1 STREAM Benchmark -- 4.2 Random Access Benchmark -- 4.3 Breadth-First Search -- 5 Related Work -- 6 Conclusions and Future Work -- References -- The ORKA-HPC Compiler-Practical OpenMP for FPGAs -- 1 Motivation -- 2 Related Work -- 3 The ORKA-HPC OpenMP-to-FPGA Compiler -- 3.1 OpenMP Lowering -- 3.2 FPGA Path -- 3.3 ORKA-HPC LLP-Backend -- 3.4 Host Path -- 4 Deployment -- 5 Evaluation -- 6 Contributions and Future Work -- References -- Graphs and Kernels -- Optimizing Sparse Matrix Multiplications for Graph Neural Networks -- 1 Introduction -- 2 Background -- 2.1 Graph Neural Networks -- 2.2 Sparse Matrix Storage Formats -- 3 Motivation -- 3.1 Setup -- 3.2 Results -- 4 Our Approach -- 4.1 Predictive Modeling -- 4.2 Problem Modeling -- 4.3 Training Data Generation -- 4.4 Feature Engineering -- 4.5 Training the Model -- 4.6 Using the Model -- 5 Experimental Setup -- 5.1 Software and Hardware -- 5.2 Evaluation Methodology -- 6 Experimental Results -- 6.1 Overall Results -- 6.2 Compare to Prior Methods -- 6.3 Compare to Oracle Performance -- 6.4 Model Analysis -- 6.5 Discussion -- 7 Related Work -- 8 Conclusions -- References -- A Hybrid Synchronization Mechanism for Parallel Sparse Triangular Solve -- 1 Introduction -- 2 Motivation and Related Work -- 3 Preliminaries.
3.1 Sparse Matrix and Serial SpTS -- 3.2 Parallel SpTS -- 4 Our Approach -- 4.1 Overview -- 4.2 no-busy-wait -- 4.3 busy-wait -- 5 Evaluation -- 5.1 Experimental Setup -- 5.2 SpTS Performance Comparison -- 6 Conclusion and Future Work -- References -- Techniques for Managing Polyhedral Dataflow Graphs -- 1 Introduction -- 2 Background -- 2.1 GeoAc -- 2.2 SPF and the Computation API -- 2.3 Polyhedral Dataflow Graphs -- 3 Case Study: Expressing GeoAc and Examining Polyhedral Dataflow Graphs -- 3.1 Approximate Static Single Assignment -- 3.2 Producer Consumer Reductions -- 3.3 Graph Components -- 3.4 Data Dependent Control Flow -- 3.5 Dead Code Elimination -- 3.6 Subgraphs -- 3.7 Constant Size Arrays -- 3.8 Debugging Information -- 4 Related Work -- 5 Conclusion -- References -- Author Index.