

eXamen.press

eXamen.press ist eine Reihe, die Theorie und Praxis aus allen Bereichen der Informatik für die Hochschulausbildung vermittelt.

Thomas Rauber · Gudula Rünger

Parallele Programmierung

3. Auflage

 Springer

Thomas Rauber
Universität Bayreuth
Fakultät für Mathematik, Physik
und Informatik
Bayreuth
Deutschland

Gudula Rünger
Technische Universität Chemnitz
Fakultät für Informatik
Chemnitz
Deutschland

ISSN 1614-5216

ISBN 978-3-642-13603-0

DOI 10.1007/978-3-642-13604-7

ISBN 978-3-642-13604-7 (eBook)

Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über <http://dnb.d-nb.de> abrufbar.

© Springer-Verlag Berlin Heidelberg 2012

Das Werk einschließlich aller seiner Teile ist urheberrechtlich geschützt. Jede Verwertung, die nicht ausdrücklich vom Urheberrechtsgesetz zugelassen ist, bedarf der vorherigen Zustimmung des Verlags. Das gilt insbesondere für Vervielfältigungen, Bearbeitungen, Übersetzungen, Mikroverfilmungen und die Einspeicherung und Verarbeitung in elektronischen Systemen.

Die Wiedergabe von Gebrauchsnamen, Handelsnamen, Warenbezeichnungen usw. in diesem Werk berechtigt auch ohne besondere Kennzeichnung nicht zu der Annahme, dass solche Namen im Sinne der Warenzeichen- und Markenschutz-Gesetzgebung als frei zu betrachten wären und daher von jedermann benutzt werden dürften.

Gedruckt auf säurefreiem und chlorfrei gebleichtem Papier

Springer DE ist Teil der Fachverlagsgruppe Springer Science+Business Media
www.springer.de

Vorwort

Das Anliegen dieses Buches ist es, dem Leser detaillierte Kenntnisse der Parallelverarbeitung zu vermitteln und ihn insbesondere mit dem heutigen Stand der Techniken der parallelen Programmierung vertraut zu machen. Das vorliegende Buch ist die dritte Auflage des im Jahr 2000 erstmals erschienenen Buches *Parallele und Verteilte Programmierung*; die zweite Auflage stammt aus dem Jahr 2007. Seit dem Erscheinen der ersten Auflage hat die technologische Entwicklung u. a. durch die weite Verbreitung von Clustersystemen und die Einführung von Multicore-Prozessoren dazu geführt, dass die Techniken der parallelen Programmierung enorm an Wichtigkeit zugenommen haben. Dies gilt nicht nur für die bisherigen Hauptanwendungsgebiete im Bereich wissenschaftlich-technischer Berechnungen. Die parallele Programmierung spielt auch für die effiziente Nutzung typischer Desktop-Rechner eine große Rolle, so dass sich parallele Programmier-techniken in alle Bereiche der Softwareentwicklung ausbreiten. Ein neuer Trend ist auch die Auslagerung von Berechnungen auf Graphics Processing Units (GPUs), die mehrere Hundert Prozessorkerne umfassen können und somit ein großes Rechenpotential zur Verfügung stellen. Durch die durchgängig zur Verfügung stehende parallele Hardware werden in Zukunft Standard-Softwareprodukte auf Konzepten der parallelen Programmierung basieren, um die parallelen Hardwareressourcen auch ausnutzen zu können. Dadurch ergibt sich ein enormer Bedarf an Softwareentwicklern mit parallelen Programmierkenntnissen. Entsprechend fordern Prozessorhersteller, die parallele Programmierung als obligatorische Komponente in die Curricula der Informatik aufzunehmen.

Die vorliegende dritte Auflage trägt den neuen Entwicklungen dadurch Rechnung, dass die für die Programmierung von Multicore-Prozessoren erforderlichen Programmiertechniken einen breiten Raum einnehmen. Insbesondere wurde ein Kapitel zur Programmierung von GPUs neu hinzugefügt. Die anderen Kapitel wurden entsprechend der technologischen Entwicklung überarbeitet. Die Überarbeitung betrifft insbesondere das Kapitel über die Architektur paralleler Plattformen, das verstärkt die Architektur von Multicore-Prozessoren behandelt.

Das Buch ist thematisch in drei Hauptteile gegliedert, die alle Bereiche der Parallelverarbeitung beginnend mit der Architektur paralleler Plattformen bis hin zur Realisierung paralleler Anwendungsalgorithmen behandeln. Breiten Raum nimmt die eigentliche parallele Programmierung ein. Im ersten Teil geben wir einen kurzen Überblick über die Architektur paralleler Systeme, wobei wir uns vor allem auf wichtige prinzipielle Eigenschaften wie Cache- und Speicherorganisation oder Verbindungsnetzwerke einschließlich der Routing- und Switching-Techniken konzentrieren, aber auch Hardwaretechnologien wie Hyperthreading oder Speicherkonsistenzmechanismen behandeln.

Im zweiten Teil stellen wir Programmier- und Kostenmodelle sowie Methoden zur Formulierung paralleler Programme vor und beschreiben derzeit aktuelle portable Programmierungsumgebungen wie MPI, Pthreads, Java-Threads und OpenMP. Neu aufgenommen wurde der Bereich der GPU-Programmierung mit CUDA und OpenCL einschließlich der Beschreibung aktueller GPU-Architekturen. Ausführliche Programmbeispiele begleiten die Darstellung der Programmierkonzepte und dienen zur Demonstration der Unterschiede zwischen den dargestellten Programmierungsumgebungen.

Im dritten Teil wenden wir die dargestellten Programmiertechniken auf Algorithmen aus dem wissenschaftlich-technischen Bereich an. Wir konzentrieren uns dabei auf grundlegende Verfahren zur Behandlung linearer Gleichungssysteme, die für eine praktische Realisierung vieler Simulationsalgorithmen eine große Rolle spielen. Der Schwerpunkt der Darstellung liegt dabei nicht auf den mathematischen Eigenschaften der Lösungsverfahren, sondern auf der Untersuchung ihrer algorithmischen Struktur und den daraus resultierenden Parallelisierungsmöglichkeiten. Zu jedem Algorithmus geben wir z. T. mehrere, repräsentativ ausgewählte Parallelisierungsvarianten an, die sich im zugrunde liegenden Programmiermodell und der verwendeten Parallelisierungsstrategie unterscheiden. Eine Webseite mit begleitendem Material ist unter ai2.inf.uni-bayreuth.de/pp_buch_3A eingerichtet. Dort werden u. a. weitere Materialien zum Inhalt des Buches sowie Informationen zu neueren Entwicklungen zur Verfügung gestellt.

Bei der Erstellung des Manuskripts haben wir vielfältige Hilfestellung erfahren, und wir möchten an dieser Stelle all denen danken, die am Zustandekommen dieses Buches beteiligt waren. Für zahlreiche Anregungen und Verbesserungsvorschläge danken wir Jörg Dümmler, Sergei Gorlatch, Reiner Haupt, Hilmar Hennings, Klaus Hering, Michael Hofmann, Christoph Keßler, Raphael Kunis, Jens Lang, Paul Molitor, John O'Donnell, Robert Reilein, Carsten Scholtes, Michael Schwind und Reinhard Wilhelm. Kerstin Beier, Erika Brandt, Daniela Funke, Monika Glaser, Ekkehard Petzold, Michael Stach, Luise Steinbach und Michael Walter danken wir für die Mitarbeit an der L^AT_EX-Erstellung des Manuskriptes. Viele weitere Personen haben zum Gelingen dieses Buches durch zahlreiche Hinweise beigetragen; auch ihnen sei hiermit gedankt. Nicht zuletzt gilt unser Dank dem Springer-Verlag für die effiziente und angenehme Zusammenarbeit.

Inhaltsverzeichnis

1	Einleitung	1
1.1	Motivation	1
1.2	Begriffe der Parallelverarbeitung	4
1.3	Überblick über den Inhalt des Buches	6
2	Architektur paralleler Plattformen	9
2.1	Überblick über die Prozessorentwicklung	10
2.2	Parallelität innerhalb eines Prozessorkerns	14
2.3	Klassifizierung von Parallelrechnern	17
2.4	Speicherorganisation von Parallelrechnern	20
2.4.1	Rechner mit physikalisch verteiltem Speicher	21
2.4.2	Rechner mit physikalisch gemeinsamem Speicher	25
2.4.3	Reduktion der Speicherzugriffszeiten	27
2.5	Verbindungsnetzwerke	32
2.5.1	Bewertungskriterien für Netzwerke	34
2.5.2	Direkte Verbindungsnetzwerke	37
2.5.3	Einbettungen	43
2.5.4	Dynamische Verbindungsnetzwerke	46
2.6	Routing- und Switching-Strategien	54
2.6.1	Routingalgorithmen	54
2.6.2	Switching	66
2.6.3	Flusskontrollmechanismen	73
2.7	Caches und Speicherhierarchien	75
2.7.1	Charakteristika von Cache-Speichern	75
2.7.2	Cache-Kohärenz	86
2.7.3	Speicherkonsistenz	95
2.8	Parallelität auf Threadebene	101
2.8.1	Hyperthreading-Technik	102
2.8.2	Multicore-Prozessoren	103

2.8.3	Designvarianten für Multicore-Prozessoren	105
2.8.4	Beispiel: Architektur des Intel Core i7	109
2.9	Beispiel: IBM Blue Gene Supercomputer	112
3	Parallele Programmiermodelle	117
3.1	Modelle paralleler Rechnersysteme	118
3.2	Parallelisierung von Programmen	121
3.3	Ebenen der Parallelität	124
3.3.1	Parallelität auf Instruktionsebene	124
3.3.2	Datenparallelität	126
3.3.3	Parallelität in Schleifen	128
3.3.4	Funktionsparallelität	131
3.4	Explizite und implizite Darstellung der Parallelität	133
3.5	Strukturierung paralleler Programme	135
3.6	SIMD-Verarbeitung	139
3.6.1	Verarbeitung von Vektoroperationen	139
3.6.2	SIMD-Instruktionen	141
3.7	Datenverteilungen für Felder	143
3.8	Informationsaustausch	148
3.8.1	Gemeinsame Variablen	148
3.8.2	Kommunikationsoperationen	151
3.8.3	Parallele Matrix-Vektor-Multiplikation	158
4	Laufzeitanalyse paralleler Programme	165
4.1	Leistungsbewertung von Rechnersystemen	166
4.1.1	Bewertung der CPU-Leistung	166
4.1.2	MIPS und MFLOPS	168
4.1.3	Leistung von Prozessoren mit Cachespeichern	170
4.1.4	Benchmarkprogramme	172
4.2	Parallele Leistungsmaße	176
4.3	Modellierung von Laufzeiten	181
4.3.1	Realisierung von Kommunikationsoperationen	182
4.3.2	Kommunikationsoperationen auf dem Hyperwürfel	189
4.3.3	Kommunikationsoperationen auf einem Baum	199
4.4	Analyse von Laufzeitformeln	202
4.4.1	Paralleles Skalarprodukt	203
4.4.2	Parallele Matrix-Vektor-Multiplikation	205
4.5	Parallele Berechnungsmodelle	208
4.5.1	PRAM-Modelle	208
4.5.2	BSP-Modell	210
4.5.3	LogP-Modell	213

5	Message-Passing-Programmierung	217
5.1	Einführung in MPI	218
5.1.1	Einzeltransferoperationen	220
5.1.2	Globale Kommunikationsoperationen	234
5.1.3	Auftreten von Deadlocks	249
5.1.4	Prozessgruppen und Kommunikatoren	252
5.1.5	Prozessstopologien	258
5.1.6	Zeitmessung und Abbruch der Ausführung	263
5.2	Einführung in MPI-2	264
5.2.1	Prozesserzeugung und -verwaltung	264
5.2.2	Einseitige Kommunikation	267
6	Thread-Programmierung	279
6.1	Einführung in die Programmierung mit Threads	280
6.2	Programmiermodell und Grundlagen für Pthreads	286
6.2.1	Erzeugung und Verwaltung von Pthreads	289
6.2.2	Koordination von Threads	292
6.2.3	Implementierung eines Taskpools	306
6.2.4	Parallelität durch Pipelining	310
6.2.5	Realisierung eines Client-Server-Modells	315
6.2.6	Steuerung und Abbruch von Threads	320
6.2.7	Thread-Scheduling	328
6.2.8	Prioritätsinversion	333
6.2.9	Thread-spezifische Daten	336
6.3	Java-Threads	337
6.3.1	Erzeugung von Threads in Java	337
6.3.2	Synchronisation von Java-Threads	342
6.3.3	Signalmechanismus in Java	347
6.3.4	Erweiterte Java-Synchronisationsmuster	351
6.3.5	Thread-Scheduling in Java	354
6.4	OpenMP	357
6.4.1	Steuerung der parallelen Abarbeitung	358
6.4.2	Parallele Schleife	361
6.4.3	Nichtiterative parallele Bereiche	365
6.4.4	Koordination von Threads	368
6.5	Unified Parallel C	374
6.5.1	UPC Programmiermodell und Benutzung	375
6.5.2	Gemeinsame Felder	377
6.5.3	Speicherkonsistenzmodelle von UPC	378
6.5.4	Zeiger und Felder in UPC	380
6.5.5	Parallele Schleifen in UPC	382
6.5.6	UPC Synchronisation	384

7	GPU-Programmierung	387
7.1	Überblick über die Architektur von GPUs	387
7.2	Einführung in die CUDA-Programmierung	395
7.3	CUDA-Synchronisation und gemeinsamer Speicher	401
7.4	CUDA Thread Scheduling	407
7.5	Effizienter Speicherzugriff und Tiling-Techniken	408
7.6	Einführung in OpenCL	414
8	Lösung linearer Gleichungssysteme	417
8.1	Gauß-Elimination	418
8.1.1	Beschreibung der Methode	418
8.1.2	Parallele zeilenzyklische Implementierung	422
8.1.3	Parallele gesamtzyklische Implementierung	426
8.1.4	Laufzeitanalyse der gesamtzyklischen Implementierung	432
8.2	Direkte Verfahren für Gleichungssysteme mit Bandstruktur	437
8.2.1	Diskretisierung der Poisson-Gleichung	438
8.2.2	Lösung von Tridiagonalsystemen	444
8.2.3	Verallgemeinerung auf beliebige Bandmatrizen	456
8.2.4	Anwendung auf die Poisson-Gleichung	459
8.3	Klassische Iterationsverfahren	461
8.3.1	Beschreibung iterativer Verfahren	462
8.3.2	Parallele Realisierung des Jacobi-Verfahrens	466
8.3.3	Parallele Realisierung des Gauß-Seidel-Verfahrens	468
8.3.4	Rot-Schwarz-Anordnung	474
8.4	Cholesky-Faktorisierung für dünnbesetzte Matrizen	480
8.4.1	Sequentieller Algorithmus	481
8.4.2	Abspeicherungsschemata für dünnbesetzte Matrizen	487
8.4.3	Implementierung für gemeinsamen Adressraum	488
8.5	Methode der konjugierten Gradienten	497
8.5.1	Beschreibung der Methode	498
8.5.2	Parallelisierung des CG-Verfahrens	501
	Literatur	505
	Sachverzeichnis	513