

*P*ARALLEL COMPUTING *W*ORKS!

Geoffrey C. Fox

Roy D. Williams

Paul C. Messina

Parallel Computing Works!

Geoffrey C. Fox
Roy D. Williams
Paul C. Messina

Morgan Kaufmann Publishers, Inc.
San Francisco, California

Executive Editor: Bruce M. Spatz
Production Manager: Yonie Overton
Assistant Editor: Douglas Sery
Production Coordinator: Julie Pabst
Cover Design: Studio Silicon
Color Insert Design & Production: Carron Design
Copyeditor: Gary Morris
Printer: R.R. Donnelley & Sons

This book was typeset by the author using $\text{\LaTeX}$.

Editorial Offices:
Morgan Kaufmann Publishers, Inc.
340 Pine Street, Sixth Floor
San Francisco, CA 94104

© 1994 by Morgan Kaufmann Publishers, Inc.

All rights reserved
Printed in the United States of America

98 97 96 95 94 5 4 3 2 1

No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means—electronic, mechanical, photocopying, recording, or otherwise—without the prior written permission of the publisher.

Library of Congress Cataloging-in-Publication Data is available for this book.
ISBN 1-55860-253-4

Parallel Computing Works!

Preface

This book describes a set of application and systems software research projects undertaken by the Caltech Concurrent Computation Program (C³P) from 1983–1990. This parallel computing activity is organized so that applications with similar algorithmic and software challenges are grouped together. Thus, one can not only learn that parallel computing is effective on a broad range of problems but also why it works, what algorithms are needed, and what features the software should support. The description of the software has been updated through 1993 to reflect the current interests of Geoffrey Fox, now at Syracuse University but still working with many C³P collaborators through the auspices of the NSF Center for Research in Parallel Computation (CRPC).

Many C³P members wrote sections of this book. John Apostolakis wrote Section 7.4; Clive Baillie, Sections 4.3, 4.4, 7.2 and 12.6; Vas Bala, Section 13.2; Ted Barnes, Section 7.3; Roberto Battitti, Sections 6.5, 6.7, 6.8 and 9.9; Rob Clayton, Section 18.2; Dave Curkendall, Section 18.3; Hong Ding, Sections 6.3 and 6.4; David Edelsohn, Section 12.8; Jon Flower, Sections 5.2, 5.3, 5.4 and 13.5; Tom Gottschalk, Sections 9.8 and 18.4; Gary Gutt, Section 4.5; Wojtek Furmanski, Chapter 17; Mark Johnson, Section 14.2; Jeff Koller, Sections 13.4 and 15.2; Aron Kuppermann, Section 8.2; Paulette Liewer, Section 9.3; Vince McKoy, Section 8.3; Paul Messina, Chapter 2; Steve Otto, Sections 6.6, 11.4, 12.7, 13.6 and 14.3; Jean Patterson, Section 9.4; Francois Pepin, Section 12.5; Peter Reiher, Section 15.3; John Salmon, Section 12.4; Tony Skjellum, Sections 9.5, 9.6 and Chapter 16; Michael Speight, Section 7.6; Eric Van de Velde, Section 9.7; David Walker, Sections 6.2 and 8.1; Brad Werner, Section 9.2; Roy Williams, Sections 11.1, 12.2, 12.3 and Chapter 10. Geoffrey Fox wrote the remaining text. Appendix B describes many of the key C³P contributors, with brief biographies.

C³P's research depended on the support of many sponsors; central support for major projects was given by the Department of Energy and the Elec-

tronic Systems Division of the USAF. Other federal sponsors were the Joint Tactical Fusion office, NASA, NSF and the National Security Agency. C³P's start up was only possible due to two private donations from the Parsons and System Development Foundations. Generous corporate support came from ALCOA, Digital Equipment, General Dynamics, General Motors, Hitachi, Hughes, IBM, INTEL, Lockheed, McDonnell Douglas, MOTOROLA, Nippon Steel, nCUBE, Sandia National Laboratories, and Shell.

Production of this book would have been impossible without the dedicated help of Richard Alonso, Lisa Deyo, Keri Arnold, Blaise Canzian and especially Terri Canzian.

Contents

Color Plates	<i>Following page 494</i>
Preface	xvii
1 Introduction	1
1.1 Introduction	1
1.2 The National Vision for Parallel Computation	2
1.3 Caltech Concurrent Computation Program	5
1.4 How Parallel Computing Works	9
2 Technical Backdrop	13
2.1 Introduction	13
2.2 Hardware Trends	14
2.2.1 Parallel Scientific Computers Before 1980	14
2.2.2 Early 1980s	16
2.2.3 Birth of the Hypercube	18
2.2.4 Mid-1980s	21
2.2.5 Late 1980s	23
2.2.6 Parallel Systems—1992	25
2.3 Software	27
2.3.1 Languages and Compilers	27
2.3.2 Tools	29
2.4 Summary	31
3 A Methodology for Computation	33
3.1 Introduction	33
3.2 The Process	36
3.3 Complex Systems and Space-Time Structure	42
3.4 The Temporal Properties of Complex Systems	45
3.5 Spatial Properties of Complex Systems	50
3.6 Compound Complex Systems	55
3.7 Mapping Complex Systems	58

3.8	Parallel Computing Works?	59
4	Synchronous Applications I	65
4.1	QCD and the Beginning of C ³ P	65
4.2	Synchronous Applications	73
4.3	Quantum Chromodynamics	80
4.3.1	Introduction	80
4.3.2	Monte Carlo	83
4.3.3	QCD	85
4.3.4	Lattice QCD	86
4.3.5	Concurrent QCD Machines	89
4.3.6	QCD on the Caltech Hypercubes	93
4.3.7	QCD on the Connection Machine	95
4.3.8	Status and Prospects	99
4.4	Spin Models	99
4.4.1	Introduction	99
4.4.2	Ising Model	101
4.4.3	Potts Model	107
4.4.4	XY Model	112
4.4.5	O(3) Model	115
4.5	An Automata Model of Granular Materials	120
4.5.1	Introduction	120
4.5.2	Comparison to Particle Dynamics Models	121
4.5.3	Comparison to Lattice Gas Models	122
4.5.4	The Rules for the Lattice Grain Model	123
4.5.5	Implementation on a Parallel Computer	125
4.5.6	Simulations	126
4.5.7	Conclusion	129
5	Express and CrOS	131
5.1	Multicomputer Operating Systems	131
5.2	A “Packet” History of Message-passing Systems	133
5.2.1	Prehistory	134
5.2.2	Application-driven Development	134
5.2.3	Collective Communication	135
5.2.4	Automated Decomposition— <i>whoami</i>	136
5.2.5	“Melting”—a Non-crystalline Problem	136
5.2.6	The Mark III	138
5.2.7	Host Programs	139

5.2.8	A Ray Tracer—and an “Operating System”	140
5.2.9	The Crystal Router	143
5.2.10	Portability	146
5.2.11	Express	146
5.2.12	Other Message-passing Systems	149
5.2.13	What Did We Learn?	152
5.2.14	Conclusions	152
5.3	Parallel Debugging	153
5.3.1	Introduction and History	153
5.3.2	Designing a Parallel Debugger	156
5.3.3	Conclusions	166
5.4	Parallel Profiling	167
5.4.1	Missing a Point	168
5.4.2	Visualization	168
5.4.3	Goals in Performance Analysis	170
5.4.4	Overhead Analysis	171
5.4.5	Event Tracing	172
5.4.6	Data Distribution Analysis	173
5.4.7	CPU Usage Analysis	174
5.4.8	Why So Many Separate Tools?	177
5.4.9	Conclusions	178
6	Synchronous Applications II	179
6.1	Computational Issues in Synchronous Problems	179
6.2	Convectively-Dominated Flows	183
6.2.1	An Overview of the FCT Technique	184
6.2.2	Mathematics and the FCT Algorithm	184
6.2.3	Parallel Issues	185
6.2.4	Example Problem	186
6.2.5	Performance and Results	187
6.2.6	Summary	188
6.3	Magnetism	189
6.3.1	Introduction	189
6.3.2	The Computational Algorithm	189
6.3.3	Parallel Implementation and Performance	194
6.3.4	Physics Results	197
6.3.5	Conclusions	202
6.4	Phase Transitions	203
6.4.1	The case of $\mathbf{h} \ll \mathbf{J}$: Antiferromagnetic Transitions . . .	203

6.4.2	The Case of $\mathbf{h} = -\mathbf{J}$: Quantum XY Model and Topological Transition	209
6.5	Surface Reconstruction	215
6.5.1	Multigrid Method with Discontinuities	216
6.5.2	Interacting Line Processes	218
6.5.3	Generic Look-up Table and Specific Parametrization	218
6.5.4	Pyramid on a 2D Mesh of Processors	220
6.5.5	Results for Orientation Constraints	224
6.5.6	Results for Depth Constraints	227
6.5.7	Conclusions	227
6.6	Character Recognition by Neural Nets	231
6.6.1	MLP in General	232
6.6.2	Character Recognition using MLP	233
6.6.3	The Multiscale Technique	236
6.6.4	Results	237
6.6.5	Comments and Variants on the Method	243
6.7	An Adaptive Multiscale Scheme	244
6.7.1	Errors in Computing the Motion Field	245
6.7.2	Adaptive Multiscale Scheme on a Multicomputer	246
6.7.3	Conclusions	250
6.8	Collective Stereopsis	250
7	Independent Parallelism	257
7.1	Embarrassingly Parallel Problem Structure	257
7.2	Dynamically Triangulated Random Surfaces	259
7.2.1	Introduction	259
7.2.2	Discretized Strings	260
7.2.3	Computational Aspects	265
7.2.4	Performance of String Program	269
7.2.5	Conclusion	269
7.3	Numerical Study of High- T_c Spin Systems	271
7.4	Statistical Gravitational Lensing	274
7.5	Parallel Random Number Generators	278
7.6	The GENESIS Project	279
7.6.1	What Is Computational Neurobiology?	279
7.6.2	Parallel Computers?	280
7.6.3	Problems with Most Present Day Parallel Computers	280

7.6.4	What is GENESIS?	281
7.6.5	Task Farming	282
7.6.6	Distributed Modelling via the Postmaster Element . .	283
8	Full Matrix Algorithms	289
8.1	Full and Banded Matrix Algorithms	289
8.1.1	Matrix Decomposition	290
8.1.2	Basic Matrix Arithmetic	293
8.1.3	Matrix Multiplication for Banded Matrices	296
8.1.4	Systems of Linear Equations	299
8.1.5	The Gauss-Jordan Method	303
8.1.6	Other Matrix Algorithms	304
8.1.7	Concurrent Linear Algebra Libraries	305
8.1.8	Problem Structure	307
8.1.9	Conclusions	307
8.2	Quantum Mechanical Reactive Scattering	308
8.2.1	Introduction	308
8.2.2	Methodology	309
8.2.3	Parallel Algorithm	311
8.2.4	Results and Discussion	313
8.3	Electron-Molecule Collisions	320
8.3.1	Introduction	320
8.3.2	The SMC Method and Its Implementation	321
8.3.3	Parallel Implementation	323
8.3.4	Performance	325
8.3.5	Selected Results	328
8.3.6	Conclusion	331
9	Loosely Synchronous Problems	333
9.1	Problem Structure	333
9.2	Micromechanical Simulations	338
9.3	Plasma Particle-in-Cell Simulation	341
9.3.1	Introduction	341
9.3.2	GPIC Algorithm	342
9.3.3	Electron Beam Plasma Instability	343
9.3.4	Performance Results for 1D Electrostatic Code	344
9.3.5	One-Dimensional Electromagnetic Code	345
9.3.6	Dynamic Load Balancing	347
9.3.7	Summary	348

9.4	Computational Electromagnetics	349
9.5	LU Factorization	354
9.5.1	Introduction	355
9.5.2	Design Overview	357
9.5.3	Reduced-Communication Pivoting	359
9.5.4	New Data Distributions	362
9.5.5	Performance Versus Scattering	366
9.5.6	Performance	368
9.5.7	Conclusions	371
9.6	Concurrent DASSL	373
9.6.1	Introduction	373
9.6.2	Mathematical Formulation	374
9.6.3	proto-Cdyn – Simulation Layer	376
9.6.4	Concurrent Formulation	380
9.6.5	Chemical Engineering Example	387
9.6.6	Conclusions	389
9.7	Adaptive Multigrid	390
9.7.1	Introduction	390
9.7.2	The Basic Algorithm	391
9.7.3	The Adaptive Algorithm	392
9.7.4	The Concurrent Algorithm	393
9.7.5	Summary	395
9.8	Munkres Algorithm for Assignment	395
9.8.1	Introduction	395
9.8.2	The Sequential Algorithm	396
9.8.3	The Concurrent Algorithm	402
9.9	Optimization Methods for Neural Nets	407
9.9.1	Deficiencies of Steepest Descent	408
9.9.2	The “Bold Driver” Network	409
9.9.3	The Broyden-Fletcher-Goldfarb-Shanno One-Step Memoryless Quasi-Newton Method	411
9.9.4	Parallel Optimization	412
9.9.5	Experiment: the Dichotomy Problem	414
9.9.6	Experiment: Time Series Prediction	414
9.9.7	Summary	414

10 DIME Programming Environment	415
10.1 DIME	415
10.1.1 Applications and Extensions	417
10.1.2 The Components of DIME	418
10.1.3 Domain Definition	419
10.1.4 Mesh Structure	421
10.1.5 Refinement	422
10.1.6 Load Balancing	423
10.1.7 Summary	423
10.2 DIMEFEM	425
10.2.1 Memory Allocation	425
10.2.2 Operations and Elements	426
10.2.3 Navier-Stokes Solver	427
10.2.4 Results	428
10.2.5 Summary	428
11 Load Balancing and Optimization	431
11.1 Load Balancing as an Optimization Problem	431
11.1.1 Load Balancing a Finite-Element Mesh	436
11.1.2 The Optimization Problem and Physical Analogy	439
11.1.3 Algorithms for Load Balancing	443
11.1.4 Simulated Annealing	445
11.1.5 Recursive Bisection	453
11.1.6 Eigenvalue Recursive Bisection	454
11.1.7 Testing Procedure	458
11.1.8 Test Results	460
11.1.9 Conclusions	466
11.2 Physical Analogy	468
11.3 Physical Optimization	471
11.4 The Travelling Salesman Problem	473
11.4.1 Background on Local Search Heuristics	474
11.4.2 Background on Markov Chains and Simulated Annealing	475
11.4.3 The New Algorithm—Large-Step Markov Chains	476
11.4.4 Results	479
12 Irregular LS Problems	481
12.1 Irregular Loosely Synchronous Problems Are Hard	481
12.2 The Electrosensory System of the Fish	483

12.2.1	Physical Model	484
12.2.2	Mathematical Theory	485
12.2.3	Results	486
12.2.4	Summary	489
12.3	Transonic Flow	489
12.3.1	Compressible Flow Algorithm	490
12.3.2	Adaptive Refinement	491
12.3.3	Examples	491
12.3.4	Performance	492
12.3.5	Summary	492
12.4	Tree Codes for N-body Simulations	494
12.4.1	Oct-Trees	495
12.4.2	Computing Forces	497
12.4.3	Parallelism in Tree Codes	499
12.4.4	Acquiring Locally Essential Data	501
12.4.5	Comments on Performance	507
12.5	Fast Vortex Algorithm	507
12.5.1	Vortex Methods	508
12.5.2	Fast Algorithms	509
12.5.3	Hypercube Implementation	510
12.5.4	Efficiency of Parallel Implementation	512
12.5.5	Results	514
12.6	Cluster Algorithms for Spin Models	516
12.6.1	Monte Carlo Calculations of Spin Models	516
12.6.2	Cluster Algorithms	517
12.6.3	Parallel Cluster Algorithms	518
12.6.4	Self-labelling	519
12.6.5	Global Equivalencing	520
12.6.6	Other Algorithms	522
12.6.7	Summary	522
12.7	Sorting	522
12.7.1	The Merge Strategy	524
12.7.2	The Bitonic Algorithm	524
12.7.3	Shellsort or Diminishing Increment Algorithm	527
12.7.4	Quicksort or Samplesort Algorithm	528
12.8	Hierarchical Tree-Structures	534
12.8.1	Introduction	534
12.8.2	Adaptive Structures	538
12.8.3	Tree as Grid	540

12.8.4 Conclusion	543
13 Data Parallel C and Fortran	545
13.1 High-Level Languages	545
13.1.1 High Performance Fortran Perspective	545
13.1.2 Problem Architecture and Message-Passing Fortran	552
13.1.3 Problem Architecture and Fortran 77	554
13.2 A Software Tool	556
13.2.1 Is Any Assistance Really Needed?	557
13.2.2 Overview of the Tool	559
13.2.3 Dependence-based Data Partitioning	560
13.2.4 Mapping Data to Processors	563
13.2.5 Communication Analysis and Performance Improvement Transformations	564
13.2.6 Communication Analysis Algorithm	566
13.2.7 Static Performance Estimator	571
13.2.8 Conclusion	574
13.3 Fortran 90 Experiments	575
13.4 Optimizing Compilers by Neural Networks	576
13.5 ASPAR	580
13.5.1 Degrees of Difficulty	580
13.5.2 Various Parallelizing Technologies	582
13.5.3 The Local View	582
13.5.4 The "Global" View	588
13.5.5 Global Strategies	589
13.5.6 Dynamic Data Distribution	591
13.5.7 Conclusions	592
13.6 Coherent Parallel C	593
13.7 Hierarchical Memory	595
14 Asynchronous Applications	603
14.1 Asynchronous Problems	603
14.2 Melting in Two Dimensions	611
14.2.1 Problem Description	611
14.2.2 Solution Method	611
14.2.3 Concurrent Update Procedure	613
14.2.4 Performance Analysis	615
14.3 Computer Chess	616

14.3.1	Sequential Computer Chess	617
14.3.2	Parallel Computer Chess: The Hardware	624
14.3.3	Parallel Alpha-Beta Pruning	625
14.3.4	Load Balancing	630
14.3.5	Speedup Measurements	631
14.3.6	Real-time Graphical Performance Monitoring	633
14.3.7	Speculation	633
14.3.8	Summary	634
15	High-Level Asynchronous Software	635
15.1	Asynchronous Software Paradigms	635
15.2	MOOS II	638
15.2.1	Design of MOOSE	638
15.2.2	Dynamic Load-Balancing Support	639
15.2.3	What We Learned	641
15.3	Time Warp	642
16	The Zipcode Message-Passing System	647
16.1	Overview of <i>Zipcode</i>	647
16.2	Low-Level Primitives	652
16.2.1	CE/RK Overview	652
16.2.2	Interface with the CE/RK system	652
16.2.3	CE Functions	653
16.2.4	RK Calls	654
16.2.5	<i>Zipcode</i> Calls	655
16.3	High-Level Primitives	660
16.3.1	Invoices	660
16.3.2	Packing and Unpacking	665
16.3.3	The Packed-Message Functions	666
16.3.4	Fortran Interface	666
16.4	Details of Execution	668
16.4.1	Initialization/Termination	669
16.4.2	Process Creation/Destruction	669
16.5	Conclusions	670
17	MOVIE	671
17.1	Introduction	671
17.1.1	The Beginning	671
17.1.2	Towards the MOVIE System	673

17.1.3	Current Status and Outlook	674
17.2	System Overview	675
17.2.1	The MOVIE System in a Nutshell	675
17.2.2	MovieScript as Virtual Machine Language	676
17.2.3	Data-Parallel Computing	680
17.2.4	Model for MIMD-parallelism	684
17.2.5	Distributed Computing	687
17.2.6	Object Orientation	688
17.2.7	Integrated Visualization Model	694
17.2.8	“In Large” Extensibility Model	701
17.2.9	CASE Tools	703
17.2.10	Planned MOVIE Applications	707
17.3	Map Separates	711
17.3.1	Problem Specification	711
17.3.2	Test Case	713
17.3.3	Segmentation via RGB Clustering	715
17.3.4	Comparison with JPL Neural Net Results	722
17.3.5	Edge Detection via Zero Crossing	725
17.3.6	Towards the Map Expert System	728
17.3.7	Summary	729
17.4	The Ultimate User Interface	729
17.4.1	Overall Assessment	729
17.4.2	Markets and Application Areas	731
17.4.3	VR at Syracuse University	732
17.4.4	MOVIE as VR Operating Shell	733
18	Simulation and Analysis	739
18.1	MetaProblems and MetaSoftware	739
18.1.1	Applications	739
18.1.2	Asynchronous versus Loosely Synchronous?	742
18.1.3	Software for Compound Problems	746
18.2	ISIS: An Interactive Seismic Imaging System	748
18.2.1	Introduction	748
18.2.2	Concepts of Interactive Imaging	749
18.2.3	Geologist-As-Analyst	750
18.2.4	Why Interactive Imaging?	751
18.2.5	System Design	752
18.2.6	Performance Considerations	753
18.2.7	Trace Manager	754