

Contents

List of Figures	xiii
List of Tables	xv
List of Algorithms	xvii
Preface	xix
Acknowledgments	xxiii
I Fundamentals	1
1 Primer: Notation and Interfaces	3
1.1 Algorithm notation	3
1.1.1 Distributed array notation	4
1.1.2 Distributed data access	5
1.2 Parallel function interfaces	6
1.2.1 Map-based programming	8
1.2.2 Parallel execution	9
2 Introduction to pMatlab	11
2.1 Program: Mandelbrot (fine-grained embarrassingly parallel) . . .	12
2.1.1 Getting started	12
2.1.2 Parallel design	13
2.1.3 Code	15
2.1.4 Debug	18
2.1.5 Test	19
2.2 Program: ZoomImage (coarse-grained embarrassingly parallel) . .	21
2.2.1 Getting started	21
2.2.2 Parallel design	22
2.2.3 Code	25
2.2.4 Debug	27
2.2.5 Test	28

2.3	Program: ParallelIO	29
2.3.1	Getting started	29
2.3.2	Parallel design	30
2.3.3	Code	31
2.3.4	Debug	34
2.3.5	Test	35
2.4	Why these worked	35
3	Interacting with Distributed Arrays	37
3.1	Getting started	38
3.2	Parallel design	39
3.3	Code	43
3.4	Interactive debug and test	46
3.4.1	Serial code correct	46
3.4.2	Parallel code correct	47
3.4.3	Local communication correct	47
3.4.4	Remote communication correct	48
3.4.5	Measuring performance	49
3.5	Advanced topic: Introduction to parallel pipelines	49
II	Advanced Techniques	53
4	Parallel Programming Models	55
4.1	Design: Knowing when to “go parallel”	55
4.1.1	Memory and performance profiling	56
4.1.2	Parallel programming patterns	59
4.1.3	Blurimage parallel design	62
4.2	Coding	67
4.2.1	Manager/worker	67
4.2.2	Message passing	69
4.2.3	Distributed arrays	70
4.3	Debug	71
4.4	Testing	72
4.5	Performance summary	74
4.6	Summary	74
5	Advanced Distributed Array Programming	77
5.1	Introduction	77
5.2	Pure versus fragmented distributed arrays	78
5.3	MATLAB distributed array interface and architecture design	80
5.4	Maps and distributions	80
5.5	Parallel support functions	83
5.5.1	Cyclic distributions and parallel load balancing	83
5.6	Concurrency versus locality	87
5.7	Ease of implementation	89

5.8	Implementation specifics	90
5.8.1	Program execution	92
5.9	Array redistribution	93
5.10	Message passing layer	95
5.11	Performance concepts	98
5.11.1	Performance, performance, performance	98
5.11.2	Benchmarks	99
5.11.3	Minimize overhead	99
5.11.4	Embarrassingly parallel implies linear speedup	100
5.11.5	Algorithm and mapping are orthogonal	100
5.11.6	Do no harm	100
5.11.7	Watch the SLOC	100
5.11.8	No free lunch	100
5.11.9	Four easy steps	100
5.12	User results	102
6	Performance Metrics and Software Architecture	107
6.1	Introduction	107
6.2	Characterizing a parallel application	108
6.2.1	Characteristics of the example programs	108
6.2.2	Serial performance metrics	112
6.2.3	Degrees of parallelism	113
6.2.4	Parallel performance metrics (no communication)	115
6.2.5	Parallel performance metrics (with communication)	116
6.2.6	Amdahl's Law	117
6.2.7	Characterizing speedup	119
6.2.8	Spatial and temporal locality	121
6.3	Standard parallel computer	123
6.3.1	Network model	125
6.3.2	Kuck hierarchy	128
6.4	Parallel programming models	129
6.5	System metrics	131
6.5.1	Performance	131
6.5.2	Form factor	132
6.5.3	Efficiency	133
6.5.4	Software cost	135
6.5.5	Software productivity	137
III	Case Studies	141
7	Parallel Application Analysis	143
7.1	Historical overview	143
7.2	Characterizing the application space	145
7.2.1	Physical memory hierarchy	146
7.2.2	Spatial/temporal locality	147
7.2.3	Logical memory hierarchy	148

7.3	HPC Challenge: Spanning the application space	148
7.3.1	Stream	149
7.3.2	FFT	149
7.3.3	RandomAccess	151
7.3.4	HPL	151
7.4	Intrinsic algorithm performance	152
7.4.1	Data structures	152
7.4.2	Computational complexity	152
7.4.3	Degrees of parallelism	154
7.4.4	Communication complexity	155
7.5	Hardware performance	155
7.5.1	Spatial and temporal locality	156
7.5.2	Performance efficiency	156
7.5.3	Estimating performance	157
7.5.4	Analysis results	159
7.5.5	Performance implications	159
7.6	Software performance	160
7.6.1	Stream	161
7.6.2	RandomAccess	161
7.6.3	FFT	162
7.6.4	HPL	163
7.7	Performance versus effort	163
8	Stream	169
8.1	Getting started	169
8.2	Parallel design	171
8.3	Code	173
8.4	Debug	176
8.5	Test	177
9	RandomAccess	181
9.1	Getting started	182
9.2	Parallel design	183
9.2.1	Spray algorithm	184
9.2.2	Tree algorithm	187
9.3	Code	189
9.3.1	Spray code	191
9.3.2	Tree code	193
9.3.3	Coding summary	194
9.4	Debug	195
9.5	Test	197
9.5.1	Multicore performance	197
9.5.2	Multinode performance	198
10	Fast Fourier Transform	201
10.1	Getting started	202
10.2	Parallel design	203

10.3	Code	206
10.4	Debug	208
10.5	Test	209
	10.5.1 Multicore performance	209
	10.5.2 Multinode performance	211
11	High Performance Linpack	215
11.1	Getting started	216
11.2	Parallel design	217
	11.2.1 Parallel LU	219
	11.2.2 Critical path analysis	221
11.3	Code	224
11.4	Debug	227
11.5	Test	229
	11.5.1 Multicore performance	229
	11.5.2 Multinode performance	230
Appendix	Notation for Hierarchical Parallel Multicore Algorithms	233
A.1	Introduction	233
A.2	Data parallelism	234
	A.2.1 Serial algorithm	234
	A.2.2 Parallel algorithm	235
	A.2.3 Block parallel algorithm	238
	A.2.4 Hierarchical parallel algorithm	239
	A.2.5 Hierarchical block parallel algorithm	240
A.3	Pipeline parallelism	242
	A.3.1 Implicit pipeline parallel	242
	A.3.2 Task pipeline parallel	243
	A.3.3 Fine-grained task pipeline parallel	246
	A.3.4 Generic hierarchical block parallel algorithm	247
Index		251