

Paweł Czarnul

Parallel Programming for Modern High Performance Computing Systems



CRC Press
Taylor & Francis Group

A CHAPMAN & HALL BOOK

*Parallel Programming
for Modern
High Performance
Computing Systems*

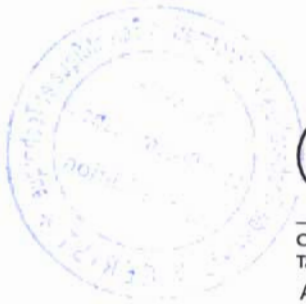
IST 33 56

BIBLIOTHEQUE DU CERIST

Parallel Programming for Modern High Performance Computing Systems

Paweł Czarnul

Gdańsk University of Technology, Poland



CRC Press

Taylor & Francis Group
Boca Raton London New York

CRC Press is an imprint of the
Taylor & Francis Group, an **informa** business
A CHAPMAN & HALL BOOK

Contents

List of figures	xiii
List of tables	xvii
List of listings	xix
Preface	xxiii
CHAPTER 1 ■ Understanding the need for parallel computing	1
1.1 INTRODUCTION	1
1.2 FROM PROBLEM TO PARALLEL SOLUTION – DEVELOPMENT STEPS	2
1.3 APPROACHES TO PARALLELIZATION	4
1.4 SELECTED USE CASES WITH POPULAR APIS	6
1.5 OUTLINE OF THE BOOK	7
CHAPTER 2 ■ Overview of selected parallel and distributed systems for high performance computing	11
2.1 GENERIC TAXONOMY OF PARALLEL COMPUTING SYSTEMS	11
2.2 MULTICORE CPUS	12
2.3 GPUS	14
2.4 MANYCORE CPUS/COPROCESSORS	17
2.5 CLUSTER SYSTEMS	19
2.6 GROWTH OF HIGH PERFORMANCE COMPUTING SYSTEMS AND RELEVANT METRICS	20
2.7 VOLUNTEER-BASED SYSTEMS	22
2.8 GRID SYSTEMS	25

CHAPTER 3 ■ Typical paradigms for parallel applications	29
<hr/>	
3.1 ASPECTS OF PARALLELIZATION	30
3.1.1 Data partitioning and granularity	30
3.1.2 Communication	32
3.1.3 Data allocation	32
3.1.4 Load balancing	33
3.1.5 HPC related metrics	34
3.2 MASTER-SLAVE	35
3.3 GEOMETRIC SPMD	39
3.4 PIPELINING	55
3.5 DIVIDE-AND-CONQUER	56
<hr/>	
CHAPTER 4 ■ Selected APIs for parallel programming	69
<hr/>	
4.1 MESSAGE PASSING INTERFACE (MPI)	74
4.1.1 Programming model and application structure	74
4.1.2 The world of MPI processes and threads	75
4.1.3 Initializing and finalizing usage of MPI	75
4.1.4 Communication modes	76
4.1.5 Basic point-to-point communication routines	76
4.1.6 Basic MPI collective communication routines	78
4.1.7 Packing buffers and creating custom data types	83
4.1.8 Receiving a message with wildcards	85
4.1.9 Receiving a message with unknown data size	86
4.1.10 Various send modes	87
4.1.11 Non-blocking communication	88
4.1.12 One-sided MPI API	90
4.1.13 A sample MPI application	95
4.1.14 Multithreading in MPI	97
4.1.15 Dynamic creation of processes in MPI	99
4.1.16 Parallel MPI I/O	101
4.2 OPENMP	102
4.2.1 Programming model and application structure	102
4.2.2 Commonly used directives and functions	104
4.2.3 The number of threads in a parallel region	109

4.2.4	Synchronization of threads within a parallel region and single thread execution	109
4.2.5	Important environment variables	111
4.2.6	A sample OpenMP application	112
4.2.7	Selected SIMD directives	115
4.2.8	Device offload instructions	115
4.2.9	Tasking in OpenMP	117
4.3	PTHREADS	118
4.3.1	Programming model and application structure	118
4.3.2	Mutual exclusion	121
4.3.3	Using condition variables	123
4.3.4	Barrier	124
4.3.5	Synchronization	125
4.3.6	A sample Pthreads application	125
4.4	CUDA	127
4.4.1	Programming model and application structure	127
4.4.2	Scheduling and synchronization	131
4.4.3	Constraints	134
4.4.4	A sample CUDA application	134
4.4.5	Streams and asynchronous operations	137
4.4.6	Dynamic parallelism	141
4.4.7	Unified Memory in CUDA	143
4.4.8	Management of GPU devices	145
4.5	OPENCL	147
4.5.1	Programming model and application structure	147
4.5.2	Coordinates and indexing	155
4.5.3	Queuing data reads/writes and kernel execution	156
4.5.4	Synchronization functions	157
4.5.5	A sample OpenCL application	158
4.6	OPENACC	167
4.6.1	Programming model and application structure	167
4.6.2	Common directives	168
4.6.3	Data management	169
4.6.4	A sample OpenACC application	171
4.6.5	Asynchronous processing and synchronization	171
4.6.6	Device management	172

4.7	SELECTED HYBRID APPROACHES	172
4.7.1	MPI+Pthreads	173
4.7.2	MPI+OpenMP	177
4.7.3	MPI+CUDA	180
CHAPTER 5 ■ Programming parallel paradigms using selected APIs		185
<hr/>		
5.1	MASTER-SLAVE	185
5.1.1	MPI	186
5.1.2	OpenMP	190
5.1.3	MPI+OpenMP	197
5.1.4	MPI+Pthreads	199
5.1.5	CUDA	207
5.1.6	OpenMP+CUDA	213
5.2	GEOMETRIC SPMD	218
5.2.1	MPI	218
5.2.2	MPI+OpenMP	220
5.2.3	OpenMP	225
5.2.4	MPI+CUDA	225
5.3	DIVIDE-AND-CONQUER	229
5.3.1	OpenMP	229
5.3.2	CUDA	232
5.3.3	MPI	235
	5.3.3.1 Balanced version	236
	5.3.3.2 Version with dynamic process creation	240
CHAPTER 6 ■ Optimization techniques and best practices for parallel codes		251
<hr/>		
6.1	DATA PREFETCHING, COMMUNICATION AND COMPUTATIONS OVERLAPPING AND INCREASING COMPUTATION EFFICIENCY	252
6.1.1	MPI	253
6.1.2	CUDA	256
6.2	DATA GRANULARITY	257
6.3	MINIMIZATION OF OVERHEADS	258

6.3.1	Initialization and synchronization overheads	258
6.3.2	Load balancing vs cost of synchronization	260
6.4	PROCESS/THREAD AFFINITY	260
6.5	DATA TYPES AND ACCURACY	261
6.6	DATA ORGANIZATION AND ARRANGEMENT	261
6.7	CHECKPOINTING	262
6.8	SIMULATION OF PARALLEL APPLICATION EXECUTION	264
6.9	BEST PRACTICES AND TYPICAL OPTIMIZATIONS	265
6.9.1	GPUs/CUDA	265
6.9.2	Intel Xeon Phi	266
6.9.3	Clusters	269
6.9.4	Hybrid systems	270
APPENDIX A ■ Resources		273
<hr/>		
A.1	SOFTWARE PACKAGES	273
APPENDIX B ■ Further reading		275
<hr/>		
B.1	CONTEXT OF THIS BOOK	275
B.2	OTHER RESOURCES ON PARALLEL PROGRAMMING	275
Index		297
<hr/>		