

Contents

LIST OF PROGRAMS AND SCRIPTS XI

TABLE OF FIGURES XIV

PREFACE XXIII

WHY THIS BOOK XXIII

WHOM THIS BOOK IS FOR XXIV

HOW THIS BOOK IS ORGANIZED XXIV

SOFTWARE AND HARDWARE XXV

HOW TO USE THIS BOOK XXV

TYPOGRAPHIC CONVENTIONS XXVI

HOW TO REACH THE AUTHOR XXVI

THE BOOK'S WEB SITE XXVI

ACKNOWLEDGEMENTS XXVII

PART I JAVA PERFORMANCE AND SCALABILITY: BASICS 1

1 SOFTWARE PERFORMANCE AND SCALABILITY 3

1.1 PERFORMANCE AND SCALABILITY METRICS 3

II CONTENTS

1.1.1	<i>An Analogy between Electrical Circuits and Computer Systems</i>	4
1.1.2	<i>Ohm's Law versus Little's Law</i>	5
1.2	A PRIMER ON QUEUING THEORY	6
1.2.1	<i>Response Time Performance Laws</i>	6
1.2.2	<i>The Throughput Performance Law</i>	8
1.2.3	<i>Jackson's Theorem</i>	9
1.3	A REAL WORLD CASE STUDY	11
1.3.1	<i>Test Environment</i>	11
1.3.2	<i>CPU as a Critical Performance Factor</i>	12
1.3.3	<i>Hyper-Threaded or Non-Hyper-Threaded</i>	15
1.4	PERFORMANCE VERSUS SCALABILITY	16
1.5	LINEAR SCALABILITY?	19
1.6	SYSTEM PERFORMANCE COUNTERS	20
1.6.1	<i>Windows</i>	21
1.6.2	<i>Linux</i>	23
1.7	SUMMARY	27
	EXERCISES	27
	REFERENCES	28
2	COMPUTER HARDWARE ARCHITECTURE	29
2.1	HISTORY OF INTEL PROCESSORS	30
2.1.1	<i>4-bit to 32-bit computing</i>	31
2.1.2	<i>CPU GHz racing</i>	32
2.1.3	<i>Multicore processors</i>	34
2.2	INTERNALS OF THE INTEL 80386 MICROPROCESSORS	35
2.3	PERFORMANCE FEATURES OF THE INTEL CORE I7 PROCESSORS	38
2.3.1	<i>Intel's New Microarchitecture</i>	38
2.3.2	<i>Intel's QuickPath Interconnect Technology</i>	39
2.3.3	<i>Intel's Turbo Boost Technology</i>	40

2.3.4	<i>Smart L3 Cache</i>	41
2.3.5	<i>Hyper-Threading Re-Introduced</i>	41
2.4	CPU BENCHMARKING	42
2.5	SUMMARY	44
	EXERCISES	44
	REFERENCES	45
3	JAVA VIRTUAL MACHINE FUNDAMENTALS	47
3.1	JAVA VIRTUAL MACHINES (JVMs)	49
3.1.1	<i>Overview</i>	49
3.1.2	<i>Java Program Execution</i>	50
3.1.3	<i>JVM Runtime Data Areas</i>	51
3.1.4	<i>JVM Instruction Set</i>	53
3.1.5	<i>The HotSpot JVM</i>	56
3.2	JAVA MEMORY MANAGEMENT	56
3.3	A SCALAR GARBAGE COLLECTION ALGORITHM	57
3.3.1	<i>Reference Counting</i>	57
3.3.2	<i>Cyclic Reference Structures</i>	59
3.4	VECTORIAL GARBAGE COLLECTION ALGORITHMS	60
3.4.1	<i>Simple Mark-and-Sweep</i>	60
3.4.2	<i>Tri-Colour Marking</i>	62
3.5	IMPLEMENTATION STRATEGIES	63
3.5.1	<i>Serial versus Parallel</i>	64
3.5.2	<i>Non-Compacting versus Compacting versus Copying</i>	65
3.5.3	<i>Stop-and-Copy (or Two Space Collection) versus Concurrent</i>	67
3.5.4	<i>Generational Strategy</i>	68
3.5.5	<i>Incremental strategy</i>	68
3.6	SUMMARY	68
	EXERCISES	69

REFERENCES 69

4 HOTSPOT JVMs 71

- 4.1 ORACLE JDK PERFORMANCE FEATURES 71
 - 4.1.1 *Oracle JDK Version History* 71
 - 4.1.2 *JIT Compilation* 72
 - 4.1.3 *Adaptive Optimization* 72
 - 4.1.4 *New Input/Output (NIO)* 72
 - 4.1.5 *Concurrent Utilities* 73
 - 4.1.6 *Java Quick Starter* 73
 - 4.1.7 *Cold Start versus Warm Start* 74
- 4.2 GARBAGE COLLECTORS IN HOTSPOT JVMs 74
 - 4.2.1 *Generations* 74
 - 4.2.2 *GC Performance Metrics* 75
 - 4.2.3 *Garbage Collectors with the HotSpot JVMs* 76
 - 4.2.4 *Concurrent Mark and Sweep (CMS)* 77
 - 4.2.5 *Garbage First (G1) GC* 79
- 4.3 HELPING GARBAGE COLLECTORS 80
- 4.4 GARBAGE COLLECTION TUNING 80
 - 4.4.1 *HotSpot JVM Command Line Options* 80
 - 4.4.2 *Sizing a Heap* 81
 - 4.4.3 *Tuning a Throughput Garbage Collector* 81
 - 4.4.4 *Adaptive Heap Sizing* 82
 - 4.4.5 *Tuning a CMS Garbage Collector* 82
 - 4.4.6 *Dealing With OutOfMemoryErrors* 83
- 4.5 GARBAGE COLLECTION MONITORING 84
 - 4.5.1 *Java Ergonomic Defaults* 84
 - 4.5.2 *JConsole* 85
 - 4.5.3 *VisualVM* 89

4.5.4 *Overhead of VisualVM* 97

4.5.5 *Sampling Snapshots* 98

4.5.6 *Application Snapshots* 99

4.6 SUMMARY 100

EXERCISES 101

REFERENCES 101

PART II JAVA PERFORMANCE AND SCALABILITY: PRACTICE 103

5 64-BIT ENTERPRISE JAVA COMPUTING 105

5.1 THE BITS 105

5.2 MEMORY BLOAT FROM 32-BIT TO 64-BIT 108

5.3 64-BIT JVMs 108

5.4 COMPRESSING ORDINARY OBJECT POINTERS WITH USECOMPRESSED_OOPS FLAG 109

5.5 CASE STUDY: USECOMPRESSED_OOPS 110

5.5.1 *The 64-bit System Under Test: AMD64 Opteron* 110

5.5.2 *The Principle of “Comparable Workloads, One Single Variable”* 111

5.5.3 *The Measurements and Comparisons* 112

5.6 SUMMARY 123

EXERCISES 123

REFERENCES 123

6 TUNING JAVA FOR BEST POSSIBLE PERFORMANCE AND SCALABILITY 125

6.1 QUANTIFYING PERFORMANCE AND SCALABILITY ISSUES 126

6.1.1 *Application Architecture* 126

6.1.2 *Test Environment* 126

6.1.3 *Hardware Specs* 127

6.1.4 *Load Testing* 127

6.1.5 *Performance and Scalability Issues* 127

6.2 APPLICATION PROFILING 130

6.2.1 *Profiling the 400 Concurrent User Run with VisualVM* 130

VI CONTENTS

6.2.2	<i>Profiling in Single Threaded Mode</i>	133
6.3	RESOLUTION	136
6.3.1	<i>Newer JDKs</i>	137
6.3.2	<i>Faster Hardware</i>	138
6.4	JVM TUNING	143
6.4.1	<i>Quantifying the Performance and Scalability Issues</i>	144
6.4.2	<i>Baseline with the Throughput GC</i>	144
6.4.3	<i>Switching to the CMS GC</i>	148
6.4.4	<i>Verifying the Parameters Used In Conjunction with – XX:+UseConcMarkSweepGC</i>	151
6.5	SYSTEMATICAL TUNINGS FOR BEST POSSIBLE PERFORMANCE AND SCALABILITY	153
6.5.1	<i>Establishing a Solid Baseline</i>	153
6.5.2	<i>Quantifying the Baseline</i>	154
6.5.3	<i>JDBC Tunings</i>	158
6.5.4	<i>Sizing Thread Pools with PSI-Probe and VisualVM</i>	159
6.5.5	<i>JVM Statistics</i>	161
6.6	SCALABILITY TESTING	162
6.6.1	<i>Average Response Times</i>	163
6.6.2	<i>System Resource Utilizations</i>	164
6.6.3	<i>JVM Activity Analysis</i>	165
6.6.4	<i>Thread Pool Analysis</i>	166
6.6.5	<i>Database Activity Analysis</i>	167
6.6.6	<i>Fine-Tuning JDBC Connection Pool</i>	168
6.6.7	<i>Further Tuning with JVM options -XX:+AggressiveOpts and – XX:+DoEscapeAnalysis</i>	171
6.6.8	<i>Scaling Beyond</i>	175
6.6.9	<i>Trials with G1 GC</i>	176
6.7	SUMMARY	179
	EXERCISES	180

REFERENCES 181

7 JAVA PERFORMANCE AND SCALABILITY BY DESIGN 183

- 7.1 THE CPUCHECK PROGRAM 183
 - 7.1.1 *The Admin Server 184*
 - 7.1.2 *The Client 191*
- 7.2 THE SERVICEDIRECTORY PROGRAM 194
 - 7.2.1 *The Static Algorithm 195*
 - 7.2.2 *The Recursive Algorithm 195*
 - 7.2.3 *The Dynamic Algorithm 195*
- 7.3 TEST RESULTS 196
 - 7.3.1 *Performance Gaps with the Static, Recursive and Dynamic Algorithms 196*
 - 7.3.2 *Part versus Whole in Evaluating Performance 197*
- 7.4 BOTTLENECK ANALYSIS 198
- 7.5 ARRAY PROCESSING TO BOOST PERFORMANCE AND SCALABILITY SIGNIFICANTLY 199
- 7.6 PERFORMANCE AND SCALABILITY ANTI-PATTERNS 201
- 7.7 SUMMARY 202

EXERCISES 202

REFERENCES 203

8 JAVA PERFORMANCE AND SCALABILITY ON LINUX 205

- 8.1 THE EVOLUTION OF UNIX 205
- 8.2 LINUX AND ITS VARIANTS 206
 - 8.2.1 *Linux Kernel Features 206*
 - 8.2.2 *Debian/Ubuntu 209*
 - 8.2.3 *Fedora 209*
 - 8.2.4 *Red Hat Enterprise Linux (RHEL) 209*
 - 8.2.5 *OpenSUSE/SUSE Linux Enterprise 210*
- 8.3 LINUX ARCHITECTURE 210
 - 8.3.1 *The Standard C Library 210*

8.3.2	<i>System Calls</i>	212
8.3.3	<i>Responsibilities of the Kernel</i>	212
8.3.4	<i>Virtual Memory</i>	212
8.3.5	<i>Virtual and Physical Address Spaces</i>	213
8.3.6	<i>Page Tables</i>	214
8.3.7	<i>Swapping and Page Reclaiming</i>	216
8.3.8	<i>UMA versus NUMA</i>	216
8.3.9	<i>Processes</i>	217
8.3.10	<i>Threads</i>	219
8.4	KNOWING ABOUT YOUR LINUX SYSTEMS	219
8.4.1	<i>Version of a Linux System</i>	219
8.4.2	<i>CPU model</i>	220
8.4.3	<i>Memory</i>	223
8.4.4	<i>Disk</i>	226
8.4.5	<i>Network</i>	226
8.4.6	<i>Checking and Configuring Kernel Parameters</i>	227
8.4.7	<i>Configuring Shell Limits for a User</i>	230
8.4.8	<i>Linux Performance Diagnostic Utilities</i>	232
8.5	CASE STUDY: JAVA PERFORMANCE AND SCALABILITY ON LINUX	237
8.5.1	<i>The Linux System Activity Monitoring Procedure</i>	237
8.5.2	<i>Test Results</i>	238
8.5.3	<i>Post-Analysis</i>	240
8.5.4	<i>Linux versus Windows</i>	248
8.6	SUMMARY	252
	EXERCISES	253
	REFERENCES	253
9	JAVA PERFORMANCE AND SCALABILITY ON VMWARE VSPHERE	255
9.1	INTRODUCTION TO THE VMWARE VSPHERE PLATFORM	255

9.1.1	<i>What is a Hypervisor?</i>	256
9.1.2	<i>VMware vSphere Virtualization Platform</i>	257
9.1.3	<i>VMware vMotion and DRS</i>	260
9.2	VMWARE VSPHERE RESOURCE MANAGEMENT	261
9.2.1	<i>Resource Allocation Shares, Reservations and Limits</i>	261
9.2.2	<i>Admission Control</i>	264
9.3	VMWARE CPU VIRTUALIZATION	264
9.3.1	<i>Software-Based CPU Virtualization</i>	265
9.3.2	<i>Hardware-Assisted CPU Virtualization</i>	265
9.3.3	<i>Multicore Processors</i>	266
9.3.4	<i>Hyperthreading</i>	267
9.3.5	<i>CPU Affinity</i>	267
9.4	VMWARE MEMORY VIRTUALIZATION	268
9.4.1	<i>Memory Over-Commitment</i>	269
9.4.2	<i>Memory Sharing</i>	269
9.4.3	<i>Software-Based Memory Virtualization</i>	269
9.4.4	<i>Hardware-Assisted Memory Virtualization</i>	270
9.5	VMWARE VM CONFIGURATION MAXIMUMS	270
9.6	CASE STUDY: JAVA PERFORMANCE AND SCALABILITY ON VMWARE VSPHERE (VIRTUAL VERSUS PHYSICAL)	271
9.6.1	<i>A 4 vCPU VM versus a 4 Physical CPU Physical System</i>	271
9.6.2	<i>An 8 vCPU VM versus an 8 Physical CPU Physical System</i>	275
9.7	SCALABILITY (VM VERSUS PHYSICAL)	279
9.8	DOES SETTING CPU SHARES TO HIGH REALLY HELP?	280
9.9	SUMMARY	283
	EXERCISES	283
	REFERENCES	284
	APPENDIX A CPUCHECK: AN CLIENT/SERVER JAVA PROGRAM FOR BENCHMARKING CPUS	285

x CONTENTS

A.1 HOW TO RUN CPUCHECK 285

A.2 MORE ABOUT CPUCHECK 288

A.2.1 The Admin Server 288

A.2.2 The Client 289

A.2.3 ServiceDirectory.java 291

APPENDIX B THE DISKCHECK PROGRAM FOR BENCHMARKING IO 311

APPENDIX C SETTING UP X11 FOR LINUX 315

INDEX 317