

A photograph of two runners in motion, blurred to convey speed. The runner in the foreground is wearing a colorful singlet and dark shorts, while the runner behind is in a white singlet and dark shorts. The background is a blurred track.

NETWORK ALGORITHMS

AN INTERDISCIPLINARY APPROACH TO DESIGNING FAST NETWORKED DEVICES



GEORGE VARGHESE

CONTENTS

PREFACE *xix*

PART I The Rules of the Game 1

CHAPTER 1 Introducing Network Algorithmics 3

- 1.1 The Problem: Network Bottlenecks 3**
 - 1.1.1 Endnode Bottlenecks 4
 - 1.1.2 Router Bottlenecks 5
- 1.2 The Techniques: Network Algorithmics 7**
 - 1.2.1 Warm-up Example: Scenting an Evil Packet 8
 - 1.2.2 Strawman Solution 9
 - 1.2.3 Thinking Algorithmically 9
 - 1.2.4 Refining the Algorithm: Exploiting Hardware 10
 - 1.2.5 Cleaning Up 11
 - 1.2.6 Characteristics of Network Algorithmics 13
- 1.3 Exercise 15**

CHAPTER 2 Network Implementation Models 16

- 2.1 Protocols 17**
 - 2.1.1 Transport and Routing Protocols 17
 - 2.1.2 Abstract Protocol Model 17
 - 2.1.3 Performance Environment and Measures 19
- 2.2 Hardware 21**
 - 2.2.1 Combinatorial Logic 21
 - 2.2.2 Timing and Power 22

2.2.3	Raising the Abstraction Level of Hardware Design	23
2.2.4	Memories	25
2.2.5	Memory Subsystem Design Techniques	29
2.2.6	Component-Level Design	30
2.2.7	Final Hardware Lessons	31
2.3	Network Device Architectures	32
2.3.1	Endnode Architecture	32
2.3.2	Router Architecture	34
2.4	Operating Systems	39
2.4.1	Uninterrupted Computation via Processes	39
2.4.2	Infinite Memory via Virtual Memory	41
2.4.3	Simple I/O via System Calls	43
2.5	Summary	44
2.6	Exercises	44

CHAPTER 3 Fifteen Implementation Principles 50

3.1	Motivating the Use of Principles — Updating Ternary Content-Addressable Memories	50
3.2	Algorithms versus Algorithmics	54
3.3	Fifteen Implementation Principles — Categorization and Description	56
3.3.1	Systems Principles	56
3.3.2	Principles for Modularity with Efficiency	61
3.3.3	Principles for Speeding Up Routines	63
3.4	Design versus Implementation Principles	65
3.5	Caveats	66
3.5.1	Eight Cautionary Questions	68
3.6	Summary	70
3.7	Exercises	70

CHAPTER 4 Principles in Action 73

4.1	Buffer Validation of Application Device Channels	74
4.2	Scheduler for Asynchronous Transfer Mode Flow Control	76
4.3	Route Computation Using Dijkstra's Algorithm	77
4.4	Ethernet Monitor Using Bridge Hardware	80

4.5	Demultiplexing in the X-Kernel	81
4.6	Tries with Node Compression	83
4.7	Packet Filtering in Routers	85
4.8	Avoiding Fragmentation of Link State Packets	87
4.9	Policing Traffic Patterns	90
4.10	Identifying a Resource Hog	92
4.11	Getting Rid of the TCP Open Connection List	93
4.12	Acknowledgment Withholding	96
4.13	Incrementally Reading a Large Database	98
4.14	Binary Search of Long Identifiers	100
4.15	Video Conferencing via Asynchronous Transfer Mode	102

PART II Playing With Endnodes 105

CHAPTER 5 Copying Data 107

5.1	Why Data Copies	109
5.2	Reducing Copying via Local Restructuring	111
5.2.1	Exploiting Adaptor Memory	111
5.2.2	Using Copy-on-Write	113
5.2.3	Fbufs: Optimizing Page Remapping	115
5.2.4	Transparently Emulating Copy Semantics	119
5.3	Avoiding Copying Using Remote DMA	121
5.3.1	Avoiding Copying in a Cluster	122
5.3.2	Modern-Day Incarnations of RDMA	123
5.4	Broadening to File Systems	125
5.4.1	Shared Memory	125
5.4.2	IO-Lite: A Unified View of Buffering	126
5.4.3	Avoiding File System Copies via I/O Splicing	128
5.5	Broadening beyond Copies	129
5.6	Broadening beyond Data Manipulations	131
5.6.1	Using Caches Effectively	131
5.6.2	Direct Memory Access versus Programmed I/O	135

5.7 Conclusions 135

5.8 Exercises 137

CHAPTER 6 Transferring Control 139

6.1 Why Control Overhead? 141

6.2 Avoiding Scheduling Overhead in Networking Code 143

6.2.1 Making User-Level Protocol Implementations Real 144

6.3 Avoiding Context-Switching Overhead in Applications 146

6.3.1 Process per Client 147

6.3.2 Thread per Client 148

6.3.3 Event-Driven Scheduler 150

6.3.4 Event-Driven Server with Helper Processes 150

6.3.5 Task-Based Structuring 151

6.4 Fast Select 153

6.4.1 A Server Mystery 153

6.4.2 Existing Use and Implementation of *Select()* 154

6.4.3 Analysis of *Select()* 155

6.4.4 Speeding Up *Select()* without Changing the API 157

6.4.5 Speeding Up *Select()* by Changing the API 158

6.5 Avoiding System Calls 159

6.5.1 The Virtual Interface Architecture (VIA) Proposal 162

6.6 Reducing Interrupts 163

6.6.1 Avoiding Receiver Livelock 164

6.7 Conclusions 165

6.8 Exercises 166

CHAPTER 7 Maintaining Timers 169

7.1 Why Timers? 169

7.2 Model and Performance Measures 171

7.3 Simplest Timer Schemes 172

7.4 Timing Wheels 173

7.5 Hashed Wheels 175

7.6 Hierarchical Wheels 176

7.7 BSD Implementation 178

7.8 Obtaining Fine-Granularity Timers 179

7.9 Conclusions 180

7.10 Exercises 181

CHAPTER 8 Demultiplexing 182

8.1 Opportunities and Challenges of Early Demultiplexing 184

8.2 Goals 184

8.3 CMU/Stanford Packet Filter: Pioneering Packet Filters 185

8.4 Berkeley Packet Filter: Enabling High-Performance Monitoring 186

8.5 Pathfinder: Factoring Out Common Checks 189

8.6 Dynamic Packet Filter: Compilers to the Rescue 192

8.7 Conclusions 195

8.8 Exercises 195

CHAPTER 9 Protocol Processing 197

9.1 Buffer Management 198

9.1.1 Buffer Allocation 199

9.1.2 Sharing Buffers 201

9.2 Cyclic Redundancy Checks and Checksums 203

9.2.1 Cyclic Redundancy Checks 204

9.2.2 Internet Checksums 207

9.2.3 Finessing Checksums 209

9.3 Generic Protocol Processing 209

9.3.1 UDP Processing 212

9.4 Reassembly 213

9.4.1 Efficient Reassembly 214

9.5 Conclusions 216

9.6 Exercises 217

PART III Playing With Routers 219

CHAPTER 10 Exact-Match Lookups 221

10.1 Challenge 1: Ethernet under Fire 222

10.2	Challenge 2: Wire Speed Forwarding	224
10.3	Challenge 3: Scaling Lookups to Higher Speeds	228
10.3.1	Scaling via Hashing	228
10.3.2	Using Hardware Parallelism	230
10.4	Summary	231
10.5	Exercise	232

CHAPTER 11 Prefix-Match Lookups 233

11.1	Introduction to Prefix Lookups	234
11.1.1	Prefix Notation	234
11.1.2	Why Variable-Length Prefixes?	235
11.1.3	Lookup Model	236
11.2	Finessing Lookups	238
11.2.1	Threaded Indices and Tag Switching	238
11.2.2	Flow Switching	240
11.2.3	Status of Tag Switching, Flow Switching, and Multiprotocol Label Switching	241
11.3	Nonalgorithmic Techniques for Prefix Matching	242
11.3.1	Caching	242
11.3.2	Ternary Content-Addressable Memories	242
11.4	Unibit Tries	243
11.5	Multibit Tries	245
11.5.1	Fixed-Stride Tries	246
11.5.2	Variable-Stride Tries	247
11.5.3	Incremental Update	250
11.6	Level-Compressed (LC) Tries	250
11.7	Lulea-Compressed Tries	252
11.8	Tree Bitmap	255
11.8.1	Tree Bitmap Ideas	255
11.8.2	Tree Bitmap Search Algorithm	256
11.9	Binary Search on Ranges	257
11.10	Binary Search on Prefix Lengths	259
11.11	Memory Allocation in Compressed Schemes	261
11.11.1	Frame-Based Compaction	262

11.12 Lookup-Chip Model 263

11.13 Conclusions 265

11.14 Exercises 266

CHAPTER 12 Packet Classification 270

12.1 Why Packet Classification? 271

12.2 Packet-Classification Problem 273

12.3 Requirements and Metrics 275

12.4 Simple Solutions 276

12.4.1 Linear Search 276

12.4.2 Caching 276

12.4.3 Demultiplexing Algorithms 277

12.4.4 Passing Labels 277

12.4.5 Content-Addressable Memories 278

12.5 Two-Dimensional Schemes 278

12.5.1 Fast Searching Using Set-Pruning Trees 278

12.5.2 Reducing Memory Using Backtracking 281

12.5.3 The Best of Both Worlds: Grid of Tries 281

12.6 Approaches to General Rule Sets 284

12.6.1 Geometric View of Classification 284

12.6.2 Beyond Two Dimensions: The Bad News 286

12.6.3 Beyond Two Dimensions: The Good News 286

12.7 Extending Two-Dimensional Schemes 287

12.8 Using Divide-and-Conquer 288

12.9 Bit Vector Linear Search 289

12.10 Cross-Producing 292

12.11 Equivalenced Cross-Producing 293

12.12 Decision Tree Approaches 296

12.13 Conclusions 299

12.14 Exercises 300

CHAPTER 13 Switching 302

13.1 Router versus Telephone Switches 304

13.2	Shared-Memory Switches	305
13.3	Router History: From Buses to Crossbars	305
13.4	The Take-a-Ticket Crossbar Scheduler	307
13.5	Head-of-Line Blocking	311
13.6	Avoiding Head-of-Line Blocking via Output Queuing	312
13.7	Avoiding Head-of-Line Blocking by Using Parallel Iterative Matching	314
13.8	Avoiding Randomization with iSLIP	316
13.8.1	Extending iSLIP to Multicast and Priority	320
13.8.2	iSLIP Implementation Notes	322
13.9	Scaling to Larger Switches	323
13.9.1	Measuring Switch Cost	324
13.9.2	Clos Networks for Medium-Size Routers	324
13.9.3	Benes Networks for Larger Routers	328
13.10	Scaling to Faster Switches	333
13.10.1	Using Bit Slicing for Higher-Speed Fabrics	333
13.10.2	Using Short Links for Higher-Speed Fabrics	334
13.10.3	Memory Scaling Using Randomization	335
13.11	Conclusions	336
13.12	Exercises	337

CHAPTER 14 Scheduling Packets 339

14.1	Motivation for Quality of Service	340
14.2	Random Early Detection	342
14.3	Token Bucket Policing	345
14.4	Multiple Outbound Queues and Priority	346
14.5	A Quick Detour into Reservation Protocols	347
14.6	Providing Bandwidth Guarantees	348
14.6.1	The Parochial Parcel Service	348
14.6.2	Deficit Round-Robin	350
14.6.3	Implementation and Extensions of Deficit Round-Robin	351
14.7	Schedulers That Provide Delay Guarantees	354
14.8	Scalable Fair Queuing	358
14.8.1	Random Aggregation	359

14.8.2	Edge Aggregation	359
14.8.3	Edge Aggregation with Policing	360
14.9	Summary	361
14.10	Exercises	361

CHAPTER 15 Routers as Distributed Systems 362

15.1	Internal Flow Control	363
15.1.1	Improving Performance	364
15.1.2	Rescuing Reliability	365
15.2	Internal Striping	368
15.2.1	Improving Performance	368
15.2.2	Rescuing Reliability	369
15.3	Asynchronous Updates	371
15.3.1	Improving Performance	372
15.3.2	Rescuing Reliability	373
15.4	Conclusions	373
15.5	Exercises	374

PART IV Endgame 377

CHAPTER 16 Measuring Network Traffic 379

16.1	Why Measurement Is Hard	381
16.1.1	Why Counting Is Hard	381
16.2	Reducing SRAM Width Using DRAM Backing Store	382
16.3	Reducing Counter Width Using Randomized Counting	384
16.4	Reducing Counters Using Threshold Aggregation	385
16.5	Reducing Counters Using Flow Counting	387
16.6	Reducing Processing Using Sampled NetFlow	388
16.7	Reducing Reporting Using Sampled Charging	389
16.8	Correlating Measurements Using Trajectory Sampling	390
16.9	A Concerted Approach to Accounting	392
16.10	Computing Traffic Matrices	393
16.10.1	Approach 1: Internet Tomography	394

16.10.2 Approach 2: Per-Prefix Counters 394

16.10.3 Approach 3: Class Counters 395

16.11 Sting as an Example of Passive Measurement 395

16.12 Conclusion 396

16.13 Exercises 397

CHAPTER 17 Network Security 399

17.1 Searching for Multiple Strings in Packet Payloads 401

17.1.1 Integrated String Matching Using Aho–Corasick 402

17.1.2 Integrated String Matching Using Boyer–Moore 403

17.2 Approximate String Matching 405

17.3 IP Traceback via Probabilistic Marking 406

17.4 IP Traceback via Logging 409

17.4.1 Bloom Filters 410

17.4.2 Bloom Filter Implementation of Packet Logging 412

17.5 Detecting Worms 413

17.6 Conclusion 415

17.7 Exercises 415

CHAPTER 18 Conclusions 417

18.1 What This Book Has Been About 418

18.1.1 Endnode Algorithmics 418

18.1.2 Router Algorithmics 419

18.1.3 Toward a Synthesis 420

18.2 What Network Algorithmics Is About 423

18.2.1 Interdisciplinary Thinking 423

18.2.2 Systems Thinking 424

18.2.3 Algorithmic Thinking 425

18.3 Network Algorithmics and Real Products 427

18.4 Network Algorithmics: Back to the Future 429

18.4.1 New Abstractions 429

18.4.2 New Connecting Disciplines 430

18.4.3 New Requirements 431

18.5 The Inner Life of a Networking Device 431

APPENDIX Detailed Models 433

A.1 TCP and IP 433

A.1.1 Transport Protocols 433

A.1.2 Routing Protocols 436

A.2 Hardware Models 437

A.2.1 From Transistors to Logic Gates 437

A.2.2 Timing Delays 439

A.2.3 Hardware Design Building Blocks 439

A.2.4 Memories: The Inside Scoop 440

A.2.5 Chip Design 441

A.3 Switching Theory 442

A.3.1 Matching Algorithms for Clos Networks with $k = n$ 442

A.4 The Interconnection Network Zoo 443

Bibliography 445

Index 457