Writing High Performance Code

If you're like most developers, you rely heavily on JavaScript to build interactive and quick-responding web applications. The problem is that all of those lines of JavaScript code can slow down your apps. This book reveals techniques and strategies to help you eliminate performance bottlenecks during development. You'll learn how to improve execution time, downloading, interaction with the DOM, page life cycle, and more. Yahoo! frontend engineer Nicholas C. Zakas and five other JavaScript experts—Ross Harmes, Julien Lecomte, Steven Levithan, Stoyan Stefanov, and Matt Sweeney—demonstrate optimal ways to load code onto a page, and offer programming tips to help your JavaScript run as efficiently and quickly as possible. You'll learn the best practices to build and deploy your files to a production environment, and tools that can help you find problems once your site goes live. Identify problem code and use faster alternatives to accomplish the same task Improve scripts by learning how JavaScript stores and accesses data Implement JavaScript code so that it doesn't slow down interaction with the DOM Use optimization techniques to improve runtime performance Learn ways to ensure the UI is responsive at all times Achieve faster client-server communication Use a build system to minify files, and HTTP compression to deliver them to the browser
Thinking Low-Level, Writing High-Level, the second volume in the landmark Write Great Code series by Randall Hyde, covers high-level programming languages (such as Swift and Java) as well as code generation on 64-bit CPUsARM, the Java Virtual Machine, and the Microsoft Common Runtime. Today's programming languages offer productivity and portability, but also make it easy to write sloppy code that isn't optimized for a compiler. Thinking Low-Level, Writing High-Level will teach you to craft source code that results in good machine code once it's run through a compiler. You'll learn: • How to analyze the output of a compiler to verify that your code generates good machine code • The types of machine code statements that compilers generate for common control structures, so you can choose the best statements when writing HLL code • Enough assembly language to read compiler output • How compilers convert various constant and variable objects into machine data With an understanding of how compilers work, you'll be able to write source code that they can translate into elegant machine code. NEW TO THIS EDITION, COVERAGE OF: • Programming languages like Swift and Java • Code generation on modern 64-bit CPUs • ARM processors on mobile phones and tablets • Stack-based architectures like the Java Virtual Machine • Modern language systems like the Microsoft Common Language Runtime
A comprehensive update of the leading algorithms text, with new material on matchings in bipartite graphs, online algorithms, machine learning, and other topics. Some books on algorithms are rigorous but incomplete; others cover masses of material but lack rigor. Introduction to Algorithms uniquely combines rigor and comprehensiveness. It covers a

broad range of algorithms in depth, yet makes their design and analysis accessible to all levels of readers, with self-contained chapters and algorithms in pseudocode. Since the publication of the first edition, Introduction to Algorithms has become the leading algorithms text in universities worldwide as well as the standard reference for professionals. This fourth edition has been updated throughout. New for the fourth edition • New chapters on matchings in bipartite graphs, online algorithms, and machine learning • New material on topics including solving recurrence equations, hash tables, potential functions, and suffix arrays • 140 new exercises and 22 new problems • Reader feedback–informed improvements to old problems • Clearer, more personal, and gender-neutral writing style • Color added to improve visual presentation • Notes, bibliography, and index updated to reflect developments in the field • Website with new supplementary material

Parallel and High Performance Computing offers techniques guaranteed to boost your code's effectiveness. Summary Complex calculations, like training deep learning models or running large-scale simulations, can take an extremely long time. Efficient parallel programming can save hours—or even days—of computing time. Parallel and High Performance Computing shows you how to deliver faster run-times, greater scalability, and increased energy efficiency to your programs by mastering parallel techniques for multicore processor and GPU hardware. About the technology Write fast, powerful, energy efficient programs that scale to tackle huge volumes of data. Using parallel programming, your code spreads data processing tasks across multiple CPUs for radically better performance. With a little help, you can create software that maximizes both speed and efficiency. About the book Parallel and High Performance Computing offers techniques guaranteed to boost your code's effectiveness. You'll learn to evaluate hardware architectures and work with industry standard tools such as OpenMP and MPI. You'll master the data structures and algorithms best suited for high performance computing and learn techniques that save energy on handheld devices. You'll even run a massive tsunami simulation across a bank of GPUs. What's inside Planning a new parallel project Understanding differences in CPU and GPU architecture Addressing underperforming kernels and loops Managing applications with batch scheduling About the reader For experienced programmers proficient with a high-performance computing language like C, C++, or Fortran. About the author Robert Robey works at Los Alamos National Laboratory and has been active in the field of parallel computing for over 30 years. Yuliana Zamora is currently a PhD student and Siebel Scholar at the University of Chicago, and has lectured on programming modern hardware at numerous national conferences. Table of Contents PART 1 INTRODUCTION TO PARALLEL COMPUTING 1 Why parallel computing? 2 Planning for parallelization 3 Performance limits and profiling 4 Data design and performance models 5 Parallel algorithms and patterns PART 2 CPU: THE PARALLEL WORKHORSE 6 Vectorization: FLOPs for free 7 OpenMP that performs 8 MPI: The parallel backbone PART

3 GPUS: BUILT TO ACCELERATE 9 GPU architectures and concepts 10 GPU programming model 11 Directive-based GPU programming 12 GPU languages: Getting down to basics 13 GPU profiling and tools PART 4 HIGH PERFORMANCE COMPUTING ECOSYSTEMS 14 Affinity: Truce with the kernel 15 Batch schedulers: Bringing order to chaos 16 File operations for a parallel world 17 Tools and resources for better code

Pro .NET Memory Management

Optimize Your C# Applications

Parallel and High Performance Computing

High Performance Spark

IPython Interactive Computing and Visualization Cookbook

High Performance Python

As programmers, we've all seen source code that's so ugly and buggy it makes our brain ache. Over the past five years, authors Dustin Boswell and Trevor Foucher have analyzed hundreds of examples of "bad code" (much of it their own) to determine why they're bad and how they could be improved. Their conclusion? You need to write code that minimizes the time it would take someone else to understand it—even if that someone else is you. This book focuses on basic principles and practical techniques you can apply every time you write code. Using easy-to-digest code examples from different languages, each chapter dives into a different aspect of coding, and demonstrates how you can make your code easy to understand. Simplify naming, commenting, and formatting with tips that apply to every line of code Refine your program's loops, logic, and variables to reduce complexity and confusion Attack problems at the function level, such as reorganizing blocks of code to do one task at a time Write effective test code that is thorough and concise—as well as readable "Being aware of how the code you create affects those who look at it later is an important part of developing software. The authors did a great job in taking you through the different aspects of this challenge, explaining the details with instructive examples." —Michael Hunger, passionate Software Developer

*Write efficient, clean, and powerful Scala code and create high-performing applications that your users will loveAbout This Book*This is the first book that explores Scala performance techniques in depth, including how to benchmark your performance so you can understand where to make gains*It provides a first-principles examination of what performance means in a Scala context*This book was written by industry experts Vincent Theron and Michael DiamantWho This Book Is ForIf you are a Scala developer with experience in programming Scala applications and know the basics in Scala, syntax, and frameworks such as Lift or Play, this book is for you. This book will also be useful if you are a Java developer who is interested in switching to Scala, but you don't want to give up the performance of Java code. No knowledge of anything outside Scala is required.What You Will Learn*Find out about performance and how to evaluate the behavior of an application*Analyze the performance of your application on JVM*Use Scala features to achieve a high performance benchmark for your application*Enhance the performance of your application with the Collection API*Explore*

*asynchronous programming to achieve concurrency and parallelism*Achieve a deeper understanding of high performance using advanced toolsIn DetailScala is a statically and strongly typed language that tries to elegantly blend both functional and object-oriented paradigms. It has experienced growing popularity in the past few years as both an appealing and pragmatic choice to write production-ready software in the functional paradigm. Scala lets you solve problems with less code than the alternatives. However, this programmatic gain can come at the cost of performance if you aren't careful.Scala High Performance Programming is written to arm you with the knowledge you need to create highly efficient, clean Scala applications. Starting with the basics of understanding what performance is in a Scala context, we'll look at how to benchmark your performance so you can see the results of your optimizations in action. We'?ll also take a deep dive into type specialization, concurrency, and parallel programming. By the end of the book, you'll be able to code efficient, optimized, solutions in Scala.*

A comprehensive guide to help aspiring and professional C++ developers elevate the performance of their apps by allowing them to run faster and consume fewer resources Key FeaturesUpdated to C++20 with completely revised code and more content on error handling, benchmarking, memory allocators, and concurrent programmingExplore the latest C++20 features including concepts, ranges, and coroutinesUtilize C++ constructs and techniques to carry out effective data structure optimization and memory managementBook Description C++ High Performance, Second Edition guides you through optimizing the performance of your C++ apps. This allows them to run faster and consume fewer resources on the device they're running on without compromising the readability of your codebase. The book begins by introducing the C++ language and some of its modern concepts in brief. Once you are familiar with the fundamentals, you will be ready to measure, identify, and eradicate bottlenecks in your C++ codebase. By following this process, you will gradually improve your style of writing code. The book then explores data structure optimization, memory management, and how it can be used efficiently concerning CPU caches. After laying the foundation, the book trains you to leverage algorithms, ranges, and containers from the standard library to achieve faster execution, write readable code, and use customized iterators. It provides hands-on examples of C++ metaprogramming, coroutines, reflection to reduce boilerplate code, proxy objects to perform optimizations under the hood, concurrent programming, and lock-free data structures. The book concludes with an overview of parallel algorithms. By the end of this book, you will have the ability to use every tool as needed to boost the efficiency of your C++ projects. What you will learnWrite specialized data structures for performance-critical codeUse modern metaprogramming techniques to reduce runtime calculationsAchieve efficient memory management using custom memory allocatorsReduce boilerplate code using reflection techniquesReap the benefits of lock-free concurrent programmingGain insights into subtle optimizations used by standard library algorithmsCompose algorithms using ranges libraryDevelop the ability to apply metaprogramming aspects such as constexpr, constraints, and conceptsImplement lazy generators and asynchronous tasks using C++20 coroutinesWho this book is for If you're a C++ developer looking to improve the efficiency of your code or just keen to upgrade your skills to the next level, this book is for you.

Apache Spark is amazing when everything clicks. But if you haven't seen the performance improvements you expected, or

still don't feel confident enough to use Spark in production, this practical book is for you. Authors Holden Karau and Rachel Warren demonstrate performance optimizations to help your Spark queries run faster and handle larger data sizes, while using fewer resources. Ideal for software engineers, data engineers, developers, and system administrators working with large-scale data applications, this book describes techniques that can reduce data infrastructure costs and developer hours. Not only will you gain a more comprehensive understanding of Spark, you'll also learn how to make it sing. With this book, you'll explore: How Spark SQL's new interfaces improve performance over SQL's RDD data structure The choice between data joins in Core Spark and Spark SQL Techniques for getting the most out of standard RDD transformations How to work around performance issues in Spark's key/value pair paradigm Writing high-performance Spark code without Scala or the JVM How to test for functionality and performance when applying suggested improvements Using Spark MLlib and Spark ML machine learning libraries Spark's Streaming components and external community packages

The Art of Writing Efficient Programs

Write Lean, High-Performance Code without the JVM

C# 7 and .NET Core 2.0 High Performance

Master the art of optimizing the functioning of your C++ code, 2nd Edition

Boost and optimize the performance of your C++17 code

Efficient R Programming

Your Python code may run correctly, but you need it to run faster. Updated for Python 3, this expanded edition shows you how to locate performance bottlenecks and significantly speed up your code in high-data-volume programs. By exploring the fundamental theory behind design choices, High Performance Python helps you gain a deeper understanding of Python's implementation. How do you take advantage of multicore architectures or clusters? Or build a system that scales up and down without losing reliability? Experienced Python programmers will learn concrete solutions to many issues, along with war stories from companies that use high-performance Python for social media analytics, productionized machine learning, and more. Get a better grasp of NumPy, Cython, and profilers Learn how Python abstracts the underlying computer architecture Use profiling to find bottlenecks in CPU time and memory usage Write efficient programs by choosing appropriate data structures Speed up matrix and vector computations Use tools to compile Python down to machine code Manage multiple I/O and computational operations concurrently Convert multiprocessing code to run on local or remote clusters Deploy code faster using tools like Docker

Learn how to write R code with fewer bugs. The problem with programming is that you are always one typo away from writing something silly. Likewise with data analysis, a small mistake in your model can lead to a big mistake in your results. Combining the two disciplines means that it is all too easy for a missed minus sign to generate a false prediction that you don't spot until it's too late. Testing is the only way to be sure that your code, and your results, are correct. Testing R Code teaches you how to perform development-time testing using the testthat package, allowing you to ensure that your code works as intended. The book also teaches run-time testing using the assertive package; enabling your users to correctly run your code. After beginning with an introduction to testing in R, the book explores more advanced cases such as integrating tests into R packages; testing code that

accesses databases; testing C++ code with Rcpp; and testing graphics. Each topic is explained with real-world examples, and has accompanying exercises for readers to practise their skills — only a small amount of experience with R is needed to get started! Ready to build mobile apps that out-perform the rest? If you're an iOS developer with app-building experience, this practical guide provides tips and best practices to help you solve many common performance issues. You'll learn how to design and optimize iOS apps that deliver a smooth experience even when the network is poor and memory is low. Today's picky users want fast and responsive apps that don't hog resources. In this book, author Gaurav Vaish demonstrates methods for writing optimal code from an engineering perspective, using reusable Objective-C code that you can use right away. Up your game and create high-performance native iOS apps that truly stand out from the crowd. Measure key performance indicators—attributes that constitute and affect app performance Write efficient apps by minimizing memory and power consumption, and explore options for using available CPU cores Optimize your app's lifecycle and UI, as well as its networking, data sharing, and security features Learn about application testing, debugging and analysis tools, and monitoring your app in the wild Collect data from real users to analyze app usage, identify bottlenecks, and provide fixes Use iOS 9 upgrades to improve your app's performance Design and develop high performing programs with Julia About This Book Learn to code high reliability and high performance programs Stand out from the crowd by developing code that runs faster than your peers' codes This book is intended for developers who are interested in high performance technical programming. Who This Book Is For This book is for beginner and intermediate Julia programmers who are interested in high performance technical computing. You will have a basic familiarity with Julia syntax, and have written some small programs in the language. What You Will Learn Discover the secrets behind Julia's speed Get a sense of the possibilities and limitations of Julia's performance Analyze the performance of Julia programs Measure the time and memory taken by Julia programs Create fast machine code using Julia's type information Define and call functions without compromising Julia's performance Understand number types in Julia Use Julia arrays to write high performance code Get an overview of Julia's distributed computing capabilities In Detail Julia is a high performance, high-level dynamic language designed to address the requirements of high-level numerical and scientific computing. Julia brings solutions to the complexities faced by developers while developing elegant and high performing code. Julia High Performance will take you on a journey to understand the performance characteristics of your Julia programs, and enables you to utilize the promise of near C levels of performance in Julia. You will learn to analyze and measure the performance of Julia code, understand how to avoid bottlenecks, and design your program for the highest possible performance. In this book, you will also see how Julia uses type information to achieve its performance goals, and how to use multuple dispatch to help the compiler to emit high performance machine code. Numbers and their arrays are obviously the key structures in scientific computing – you will see how Julia's design makes them fast. The last chapter will give you a taste of Julia's distributed computing capabilities. Style and approach This is a hands-on manual that will give you good explanations about the important concepts related to Julia programming.

Optimize Your Code for Better Apps
C++ High Performance
Thinking Low-Level, Writing High-Level
Fast and Scalable Designs

The Art of Readable Code

Writing Efficient Programs

Maximizing the performance of your algorithms and applications is extremely important and can give you a competitive advantage, a l[...] ownership, and happier users. Pro .NET Performance explains the internals of Windows, the CLR, and the physical hardware that affec[...] of your applications, and gives you the knowledge and tools to measure how your code performs in isolation from external factors. Th[...] code samples and tips to help you squeeze every bit of juice from your application—lower memory utilization, consistent CPU usage, a[...] operations across the network and disk. Pro .NET Performance will change the way you think about .NET application development. Gui[...] performance measurement with a variety of profilers and other tools Explains how OS and CLR internals affect your application's perf[...] unexpected ways Provides you with tips and real-life case studies for improving application performance

This book is for programmers and developers who want to improve the performance of their R programs by making them run faster v[...] who are trying to solve a pesky performance problem.

Do you want your .NET code to have the absolute best performance it can? This book demystifies the CLR, teaching you how and why[...] optimum performance. Learn critical lessons from a person who helped design and build one of the largest high-performance .NET sys[...] This book does not just teach you how the CLR works-it teaches you exactly what you need to do now to obtain the best performan[...] guide you through the nuts and bolts of extreme performance optimization in .NET, complete with in-depth examinations of CLR funct[...] recommendations and tutorials, useful anecdotes, and step-by-step guides to measure and improve performance. Among the topics yo[...] to: Choose what to measure and why Use many amazing tools, freely available, to solve problems quickly Understand the .NET garbage[...] effect on your application Use effective coding patterns that lead to optimal garbage collection performance Diagnose common GC-re[...] costs of JITting Use multiple threads sanely and effectively, avoiding synchronization problems Know which .NET features and APIs to[...] avoid Use code generation to avoid performance problems Measure everything and expose hidden performance issues Instrument your[...] performance counters and ETW events Use the latest and greatest .NET features Ensure your code can run on mobile devices without[...] performance-minded team …and much more.

Write code that scales across CPU registers, multi-core, and machine clusters Key Features Explore concurrent programming in C++ lo[...] management problems Use SIMD and STL containers for performance improvement Book Description C++ is a highly portable language[...] to write both large-scale applications and performance-critical code. It has evolved over the last few years to become a modern and e[...] This book will guide you through optimizing the performance of your C++ apps by allowing them to run faster and consume fewer res[...] they're running on without compromising the readability of your code base. The book begins by helping you measure and identify bottl[...] code base. It then moves on by teaching you how to use modern C++ constructs and techniques. You'll see how this affects the way[...] you'll see the importance of data structure optimization and memory management, and how it can be used efficiently with respect to[...] you'll see how STL algorithm and composable Range V3 should be used to both achieve faster execution and more readable code, follo[...] STL containers and how to write your own specialized iterators. Moving on, you'll get hands-on experience in making use of modern C[...] metaprogramming and reflection to reduce boilerplate code as well as in working with proxy objects to perform optimizations under t[...] you'll learn concurrent programming and understand lock-free data structures. The book ends with an overview of parallel algorithms[...] policies, Boost Compute, and OpenCL to utilize both the CPU and the GPU. What you will learn Benefits of modern C++ constructs and[...]

Identify hardware bottlenecks, such as CPU cache misses, to boost performance Write specialized data structures for performance-cr
modern metaprogramming techniques to reduce runtime calculations Achieve efficient memory management using custom memory allo
boilerplate code using reflection techniques Reap the benefits of lock-free concurrent programming Perform under-the-hood optimizat
readability using proxy objects Gain insights into subtle optimizations used by STL algorithms Utilize the Range V3 library for expressiv
Parallelize your code over CPU and GPU, without compromising readability Who this book is for If you're a C++ developer looking to im
of your code or simply wanting to take your skills up to the next level, then this book is perfect for you.
Pro .NET Performance
High Performance IOS Apps
Build Faster Web Application Interfaces
A Practical Guide to Smarter Programming
High Performance JavaScript
Practical Performant Programming for Humans

Intended to anyone interested in numerical computing and data science: students, researchers, teachers, engineers
hobbyists… Basic knowledge of Python/NumPy is recommended. Some skills in mathematics will help you understar
theory behind the computational methods.

Do you want your .NET code to have the absolute best performance it can? This book demystifies the CLR, teachin
and why to write code with optimum performance. Learn critical lessons from a person who helped design and bui
the largest high-performance .NET systems in the world.This book does not just teach you how the CLR works--it
you exactly what you need to do now to obtain the best performance today. It will expertly guide you through the
bolts of extreme performance optimization in .NET, complete with in-depth examinations of CLR functionality, free
recommendations and tutorials, useful anecdotes, and step-by-step guides to measure and improve performance.Ar
topics you will learn are how to:- Choose what to measure and why- Use many amazing tools, freely available, to s
problems quickly- Understand the .NET garbage collector and its effect on your application- Use effective coding pa
that lead to optimal garbage collection performance- Diagnose common GC-related issues- Reduce costs of JITting-
multiple threads sanely and effectively, avoiding synchronization problems- Know which .NET features and APIs to u
which to avoid- Use code generation to avoid performance problems- Measure everything and expose hidden perfo
issues- Instrument your program with performance counters and ETW events- Use the latest and greatest .NET fea
Ensure your code can run on mobile devices without problems- Build a performance-minded team…and much more.
How can you help your Drupal website continue to perform at the highest level as it grows to meet demand? This
comprehensive guide provides best practices, examples, and in-depth explanations for solving several performance

scalability issues. You'll learn how to apply coding and infrastructure techniques to Drupal internals, application performance, databases, web servers, and performance analysis. Covering Drupal versions 7 and 8, this book is the reference for everything from site deployment to implementing specific technologies such as Varnish, memcache, o you have a basic understanding of Drupal and the Linux-Apache-MySQL-PHP (LAMP) stack, you're ready to get star Establish a performance baseline and define goals for improvement Optimize your website's code and front-end per Get best and worst practices for customizing Drupal core functionality Apply infrastructure design techniques to la expand a site Use tools to configure, monitor, and optimize MySQL performance Employ alternative storage and bac search options as your site grows Tune your web servers through httpd and PHP configuration Monitor services a load tests to catch problems before they become critical

Understand .NET memory management internal workings, pitfalls, and techniques in order to effectively avoid a wid of performance and scalability problems in your software. Despite automatic memory management in .NET, there ar advantages to be found in understanding how .NET memory works and how you can best write software that inter efficiently and effectively. Pro .NET Memory Management is your comprehensive guide to writing better software b understanding and working with memory management in .NET. Thoroughly vetted by the .NET Team at Microsoft, th book contains 25 valuable troubleshooting scenarios designed to help diagnose challenging memory problems. Read also benefit from a multitude of .NET memory management "rules" to live by that introduce methods for writing me aware code and the means for avoiding common, destructive pitfalls. What You'll LearnUnderstand the theoretical underpinnings of automatic memory management Take a deep dive into every aspect of .NET memory management, including detailed coverage of garbage collection (GC) implementation, that would otherwise take years of experien acquire Get practical advice on how this knowledge can be applied in real-world software development Use practic knowledge of tools related to .NET memory management to diagnose various memory-related issuesExplore various of advanced memory management, including use of Span and Memory types Who This Book Is For .NET developers, architects, and performance engineers

Rust High Performance
Introduction to Algorithms, fourth edition
Optimized C++
Proven Techniques for Heightened Performance
Write Great Code, Volume 2, 2nd Edition
Advanced R

Build powerful and fast applications with F# About This Book Explore the advanced concurrency support in F# and .NET TPL Covers major optimization techniques in F# to improve the performance of applications Use Struct, Class and Record model, Interop with C# and VB without sacrificing performance. Who This Book Is For This book is for F# developers who want to build high-performance applications. Knowledge of functional programming would be helpful. What You Will Learn Understand how the execution of functions in F# works Identify common performance bottlenecks Implement best practices to optimize performance Use the available tooling to help measure performance Combine the best practice of asynchronous and synchronous Optimize further using various F# language constructs In Detail F# is a functional programming language and is used in enterprise applications that demand high performance. It has its own unique trait: it is a functional programming language and has OOP support at the same time. This book will help you make F# applications run faster with examples you can easily break down and take into your own work. You will be able to assess the performance of the program and identify bottlenecks. Beginning with a gentle overview of concurrency features in F#, you will get to know the advanced topics of concurrency optimizations in F#, such as F# message passing agent of MailboxProcessor and further interoperation with .NET TPL. Based on this knowledge, you will be able to enhance the performance optimizations when implementing and using other F# language features. The book also covers optimization techniques by using F# best practices and F# libraries. You will learn how the concepts of concurrency and parallel programming will help in improving the performance. With this, you would be able to take advantage of multi-core processors and track memory leaks, root causes, and CPU issues. Finally, you will be able to test their applications to achieve scalability. Style and approach This easy-to-follow guide is full of hands-on examples of real-world multithreading tasks. Each topic is explained and placed in context, and for the more inquisitive, there are also more in-depth details of the concepts used.

Design and develop high-performance programs in Julia 1.O Key FeaturesLearn the characteristics of high-performance Julia codeUse the power of the GPU to write efficient numerical codeSpeed up your computation with the help of newly introduced shared memory multi-threading in Julia 1.OBook Description Julia is a high-level, high-performance dynamic programming language for numerical computing. If you want to understand how to avoid bottlenecks and design your programs for the highest possible performance, then this book is for you. The book starts with how Julia uses type information to achieve its performance goals, and how to use multiple dispatches to help the compiler emit high-performance machine code. After that, you will learn how to analyze Julia programs and identify issues with time and memory consumption. We teach you how to use Julia's typing facilities accurately to write high-performance code and describe how the Julia compiler uses type information to create fast machine code. Moving ahead, you'll master design constraints and learn how to use the power of the GPU in your Julia

code and compile Julia code directly to the GPU. Then, you'll learn how tasks and asynchronous IO help you create responsive programs and how to use shared memory multithreading in Julia. Toward the end, you will get a flavor of Julia's distributed computing capabilities and how to run Julia programs on a large distributed cluster. By the end of this book, you will have the ability to build large-scale, high-performance Julia applications, design systems with a focus on speed, and improve the performance of existing programs. What you will learnUnderstand how Julia code is transformed into machine codeMeasure the time and memory taken by Julia programs Create fast machine code using Julia's type information Define and call functions without compromising Julia's performance Accelerate your code via the GPUUse tasks and asynchronous IO for responsive programsRun Julia programs on large distributed clustersWho this book is for This book is for beginners and intermediate Julia programmers who are interested in high-performance technical programming. A basic knowledge of Julia programming is assumed.

There are many excellent R resources for visualization, data science, and package development. Hundreds of scattered vignettes, web pages, and forums explain how to use R in particular domains. But little has been written on how to simply make R work effectively—until now. This hands-on book teaches novices and experienced R users how to write efficient R code. Drawing on years of experience teaching R courses, authors Colin Gillespie and Robin Lovelace provide practical advice on a range of topics—from optimizing the set-up of RStudio to leveraging C++—that make this book a useful addition to any R user's bookshelf. Academics, business users, and programmers from a wide range of backgrounds stand to benefit from the guidance in Efficient R Programming. Get advice for setting up an R programming environment Explore general programming concepts and R coding techniques Understand the ingredients of an efficient R workflow Learn how to efficiently read and write data in R Dive into data carpentry—the vital skill for cleaning raw data Optimize your code with profiling, standard tricks, and other methods Determine your hardware capabilities for handling R computation Maximize the benefits of collaborative R programming Accelerate your transition from R hacker to R programmer

In today's fast and competitive world, a program's performance is just as important to customers as the features it provides. This practical guide teaches developers performance-tuning principles that enable optimization in C++. You'll learn how to make code that already embodies best practices of C++ design run faster and consume fewer resources on any computer--whether it's a watch, phone, workstation, supercomputer, or globe-spanning network of servers. Author Kurt Guntheroth provides several running examples that demonstrate how to apply these principles incrementally to improve existing code so it meets customer requirements for responsiveness and throughput. The advice in this book will prove itself the first time you hear a colleague exclaim, "Wow, that was fast. Who fixed something?"Locate performance hot spots using the profiler and software timersLearn to perform repeatable experiments to measure performance of code changesOptimize use of dynamically allocated variablesImprove performance of

hot loops and functionsSpeed up string handling functionsRecognize efficient algorithms and optimization patternsLearn the strengths--and weaknesses--of C++ container classesView searching and sorting through an optimizer's eyeMake efficient use of C++ streaming I/O functionsUse C++ thread-based concurrency features effectively

Scala High Performance Programming

Best Practices for Scaling and Optimizing Apache Spark

An advanced programmer's guide to efficient hardware utilization and compiler optimizations using C++ examples

Boost and optimize the performance of your Golang applications at scale with resilience

Hands-On High Performance with Go

Software Optimization for High-performance Computing

Classic on practical methods of optimizing programs: This book gives practical advice on improving the efficiency (optimizing) programs and the limits there of. While showing how to trade off speed for space or vice-versa, the author points out the limits that can be expected to gain. His list of techniques is a collection of practical approaches rather than theoretical possibilities. At 158 pages (not counting index) this book is eminently readable, accessable and useful. Clearly written and well organized this is a book to keep on your shelf for when a program needs improving. It is also a book to read before a program as a reminder not to make things complicated with optimization that aren't needed

Take performance to the next level!

This book does not just teach you how the CLR works---it teaches you exactly what you need to do now to obtain the best performance today. It will expertly guide you through the nuts and bolts of extreme performance optimization in .NET, complete with in-depth examinations of CLR functionality, free tool recommendations and tutorials, useful anecdotes, and step-by-step guides to measure and improve performance.

This second edition incorporates the advances and improvements in .NET over the last few years, as well as greatly expanded coverage of tools, more topics, more tutorials, more tips, and improvements throughout the entire book.

New in the 2nd Edition:

- **50% increase in content!**
- **New examples, code samples, and diagrams throughout entire book**
- **More ways to analyze the heap and find memory problems**
- **More tool coverage, including expanded usage of Visual Studio**
- **More benchmarking**
- **New GC configuration options**
- **Code warmup techniques**
- **New .NET features such as ref-returns, value tuples, SIMD, and more**
- **More detailed analysis of LINQ**
- **Tips for high-level feature areas such as ASP.NET, ADO.NET, and WPF**

Also find expanded coverage and discover new tips and tricks for:

- **Profiling with multiple tools to quickly find problem areas**
- **Detailed description of the garbage collector, how to optimize your code for it, and how to diagnose difficult memory-related issues**
- **How to analyze JIT and diagnose warmup problems**
- **Effective use of the Task Parallel Library to maximize throughput**
- **Which .NET features and APIs to use and which to avoid**
- **Instrument your program with performance counters and ETW events**
- **Use the latest and greatest .NET features**
- **Build a performance-minded team**
- **...and so much more**

An Essential Reference for Intermediate and Advanced R Programmers Advanced R presents useful tools and techniques for attacking many types of R programming problems, helping you avoid mistakes and dead ends. With more than ten years of experience programming in R, the author illustrates the

elegance, beauty, and flexibility at the heart of R. The book develops the necessary skills to produce quality code that can be used in a variety of circumstances. You will learn: The fundamentals of R, including standard data types and functions Functional programming as a useful framework for solving wide classes of problems The positives and negatives of metaprogramming How to write fast, memory-efficient code This book not only helps current R users become R programmers but also shows existing programmers what's special about R. Intermediate R programmers can dive deeper into R and learn new strategies for solving diverse problems while programmers from other languages can learn the details of R and understand why R works the way it does.

Access the power of bare-metal systems programming with Scala Native, an ahead-of-time Scala compiler. Without the baggage of legacy frameworks and virtual machines, Scala Native lets you re-imagine how your programs interact with your operating system. Compile Scala code down to native machine instructions; seamlessly invoke operating system APIs for low-level networking and IO; control pointers, arrays, and other memory management techniques for extreme performance; and enjoy instant start-up times. Skip the JVM and improve your code performance by getting close to the metal. Developers generally build systems on top of the work of those who came before, accumulating layer upon layer of abstraction. Scala Native provides a rare opportunity to remove layers. Without the JVM, Scala Native uses POSIX and ANSI C APIs to build concise, expressive programs that run unusually close to bare metal. Scala Native compiles Scala code down to native machine instructions instead of JVM bytecode. It starts up fast, without the sluggish warm-up phase that's common for just-in-time compilers. Scala Native programs can seamlessly invoke operating system APIs for low-level networking and IO. And Scala Native lets you control pointers, arrays, and other memory layout types for extreme performance. Write practical, bare-metal code with Scala Native, step by step. Understand the foundations of systems programming, including pointers, arrays, strings, and memory management. Use the UNIX socket API to write network client and server programs without the sort of frameworks higher-level languages rely on. Put all the pieces together to design and implement a modern, asynchronous microservice-style HTTP framework from scratch. Take advantage of Scala Native's clean, modern syntax to write lean, high-performance code without the JVM. What You Need: A modern Windows, Mac OS, or Linux system capable of running Docker. All code examples in the book are designed to run on a portable Docker-based build environment that runs anywhere. If you don't have Docker yet, see the

Appendix for instructions on how to get it.
R High Performance Programming
Clojure High Performance Programming
For Better Code, Performance, and Scalability
Optimizations, distributed computing, multithreading, and GPU programming with Julia 1.0 and beyond, 2nd Edition
Python High Performance

Learn how to use Python to create efficient applications About This Book Identify the bottlenecks in your applications and solve them using the best profiling techniques Write efficient numerical code in NumPy, Cython, and Pandas Adapt your programs to run on multiple processors and machines with parallel programming Who This Book Is For The book is aimed at Python developers who want to improve the performance of their application. Basic knowledge of Python is expected What You Will Learn Write efficient numerical code with the NumPy and Pandas libraries Use Cython and Numba to achieve native performance Find bottlenecks in your Python code using profilers Write asynchronous code using Asyncio and RxPy Use Tensorflow and Theano for automatic parallelism in Python Set up and run distributed algorithms on a cluster using Dask and PySpark In Detail Python is a versatile language that has found applications in many industries. The clean syntax, rich standard library, and vast selection of third-party libraries make Python a wildly popular language. Python High Performance is a practical guide that shows how to leverage the power of both native and third-party Python libraries to build robust applications. The book explains how to use various profilers to find performance bottlenecks and apply the correct algorithm to fix them. The reader will learn how to effectively use NumPy and Cython to speed up numerical code. The book explains concepts of concurrent programming and how to implement robust and responsive applications using Reactive programming. Readers will learn how to write code for parallel architectures using Tensorflow and Theano, and use a cluster of computers for large-scale computations using technologies such as Dask and PySpark. By the end of the book, readers will have learned to achieve performance and scale from their Python applications. Style and approach A step-by-step practical guide filled with real-world use cases and examples

Understand what every developer should know about performance when building Windows Store apps. Not designed as a comprehensive reference, this book instead zeroes in on the essentials of planning for great performance and provides a solid starting point for building fast apps. This concise, performance-focused guide: Provides an introduction to the Windows platform from a performance point of view Describes how to set performance goals, establish tests to track performance, and covers tools to instrument code and analyze performance Explains why common techniques such as micro benchmarks and ad hoc testing often fall short in verifying performance Focuses on managed C#/XAML apps Although tools and techniques also apply to Visual Basic/XAML apps, all

code examples use C# HTML5/JavaScript and C++/XAML are not covered

Improve the speed of your code and optimize the performance of your apps Key Features Understand the common performance pitfalls and improve your application's performance Get to grips with multi-threaded and asynchronous programming in C# Develop highly performant applications on .NET Core using microservice architecture Book Description While writing an application, performance is paramount. Performance tuning for realworld applications often involves activities geared toward fnding bottlenecks; however, this cannot solve the dreaded problem of slower code. If you want to improve the speed of your code and optimize an application's performance, then this book is for you. C# 7 and .NET Core 2.0 High Performance begins with an introduction to the new features of what?explaining how they help in improving an application's performance. Learn to identify the bottlenecks in writing programs and highlight common performance pitfalls, and learn strategies to detect and resolve these issues early. You will explore multithreading and asynchronous programming with .NET Core and learn the importance and effcient use of data structures. This is followed with memory management techniques and design guidelines to increase an application's performance. Gradually, the book will show you the importance of microservices architecture for building highly performant applications and implementing resiliency and security in .NET Core. After reading this book, you will learn how to structure and build scalable, optimized, and robust applications in C#7 and .NET. What you will learn Measure application performance using BenchmarkDotNet Explore the techniques to write multithreaded applications Leverage TPL and PLinq libraries to perform asynchronous operations Get familiar with data structures to write optimized code Understand design techniques to increase your application's performance Learn about memory management techniques in .NET Core Develop a containerized application based on microservices architecture Learn tools and techniques to monitor application performance Who this book is for This book is for .NET developers looking at improving the speed of their code or simply wanting to take their skills to the next level. Basic C# knowledge is assumed.

Become an expert at writing fast and high performant code in Clojure 1.7.0 About This Book Enhance code performance by using appropriate Clojure features Improve the efficiency of applications and plan their deployment A hands-on guide to designing Clojure programs to get the best performance Who This Book Is For This book is intended for intermediate Clojure developers who are looking to get a good grip on achieving optimum performance. Having a basic knowledge of Java would be helpful. What You Will Learn Identify performance issues in Clojure programs using different profiling tools Master techniques to achieve numerical performance in Clojure Use Criterium library to measure latency of Clojure expressions Exploit Java features in Clojure code to enhance performance Avoid reflection and boxing with type hints Understand Clojure's concurrency and state-management primitives in depth Measure and monitor performance, and understand optimization techniques In Detail Clojure treats code as data and has a macro system. It focuses on programming with immutable values and explicit progression-of-time constructs, which are intended to facilitate the development of more robust programs, particularly multithreaded ones. It is built with performance, pragmatism, and simplicity in mind. Like most general purpose languages, various Clojure features have different performance characteristics that one should know in order to write high performance code. This book shows you how to evaluate the performance implications of various

Clojure abstractions, discover their underpinnings, and apply the right approach for optimum performance in real-world programs. It starts by helping you classify various use cases and the need for them with respect to performance and analysis of various performance aspects. You will also learn the performance vocabulary that experts use throughout the world and discover various Clojure data structures, abstractions, and their performance characteristics. Further, the book will guide you through enhancing performance by using Java interoperability and JVM-specific features from Clojure. It also highlights the importance of using the right concurrent data structure and Java concurrency abstractions. This book also sheds light on performance metrics for measuring, how to measure, and how to visualize and monitor the collected data. At the end of the book, you will learn to run a performance profiler, identify bottlenecks, tune performance, and refactor code to get a better performance. Style and approach An easy-to-follow guide full of real-world examples and self-sufficient code snippets that will help you get your hands dirty with high performance programming with Clojure.

Mastering Python High Performance

High Performance Drupal

Testing R Code

High-Performance Windows Store Apps

F# High Performance

Writing High-Performance .NET Code

Find bottlenecks, identify the proper algorithm to use, optimize performance, and create really efficient Rust applications Key Features Understand common performance pitfalls and improve the performance of your applications. Get to grips with parallel programming and multithreading with Rust. Learn metaprogramming in Rust. Book Description At times, it is difficult to get the best performance out of Rust. This book teaches you to optimize the speed of your Rust code to the level of languages such as C/C++. You'll understand and fix common pitfalls, learn how to improve your productivity by using metaprogramming, and speed up your code by concurrently executing parts of it safely and easily. You will master the features of the language which will make you stand out and use them to really improve the efficiency of your algorithms The book begins with a gentle introduction to help you identify bottlenecks when programming in Rust. We highlight common performance pitfalls, along with strategies to detect and resolve these issues early. We move on to mastering Rust's type system, which will enable us to create impressive optimizations in both performance and safety at compile time. You will then learn how to effectively manage memory in Rust, mastering the borrow checker. We move on to measuring performance and you will see how this affects the way you write code. Moving ahead, you will perform metaprogramming in Rust to boost the performance of your code and your productivity. You will finally learn parallel programming in Rust, which enables efficient and faster execution by using multithreading and asynchronous programming. What you will learn Master tips and tricks to make your code faster. Learn how to identify bottlenecks in your Rust applications Discover how to profile your Rust software. Understand the type system to

create compile-time optimizations. Master the borrow checker . Learn metaprogramming in Rust to avoid boilerplate code. Discover multithreading and work stealing in Rust. Understand asynchronous programming in Rust. Who this book is for This book is for Rust developers keen to improve the speed of their code or simply to take their skills to the next level. The hands-on guide to high-performance coding and algorithm optimization. This hands-on guide to software optimization introduces state-of-the-art solutions for every key aspect of software performance - both code-based and algorithm-based. Two leading HP software performance experts offer comparative optimization strategies for RISC and for the new Explicitly Parallel Instruction Computing (EPIC) design used in Intel IA-64 processors. Using many practical examples, they offer specific techniques for: Predicting and measuring performance - and identifying your best optimization opportunities Storage optimization: cache, system memory, virtual memory, and I/0 Parallel processing: distributed-memory and shared-memory (SMP and ccNUMA) Compilers and loop optimization Enhancing parallelism: compiler directives, threads, and message passing Mathematical libraries and algorithms Whether you're a developer, ISV, or technical researcher, if you need to optimize high-performance software on today's leading processors, one book delivers the advanced techniques and code examples you need: Software Optimization for High Performance Computing.

Proven methodologies and concurrency techniques that will help you write faster and better code with Go programming Key FeaturesExplore Go's profiling tools to write faster programs by identifying and fixing bottlenecksAddress Go-specific performance issues such as memory allocation and garbage collectionDelve into the subtleties of concurrency and discover how to successfully implement it in everyday applicationsBook Description Go is an easy-to-write language that is popular among developers thanks to its features such as concurrency, portability, and ability to reduce complexity. This Golang book will teach you how to construct idiomatic Go code that is reusable and highly performant. Starting with an introduction to performance concepts, you'll understand the ideology behind Go's performance. You'll then learn how to effectively implement Go data structures and algorithms along with exploring data manipulation and organization to write programs for scalable software. This book covers channels and goroutines for parallelism and concurrency to write high-performance code for distributed systems. As you advance, you'll learn how to manage memory effectively. You'll explore the compute unified device architecture (CUDA) application programming interface (API), use containers to build Go code, and work with the Go build cache for quicker compilation. You'll also get to grips with profiling and tracing Go code for detecting bottlenecks in your system. Finally, you'll evaluate clusters and job queues for performance optimization and monitor the application for performance regression. By the end of this Go programming book, you'll be able to improve existing code and fulfill customer requirements by writing efficient programs. What you will learnOrganize and manipulate data effectively with clusters and job queuesExplore commonly applied Go data structures and algorithmsWrite anonymous functions in Go to build reusable appsProfile and trace Go apps to reduce bottlenecks and improve efficiencyDeploy, monitor, and iterate Go programs with a focus on performanceDive into memory management and CPU and GPU parallelism in GoWho this book is for This Golang book

is a must for developers and professionals who have an intermediate-to-advanced understanding of Go programming, and are interested in improving their speed of code execution.

Boost the performance of your Haskell applications using optimization, concurrency, and parallel programming About This Book Explore the benefits of lazy evaluation, compiler features, and tools and libraries designed for high performance Write fast programs at extremely high levels of abstraction Work through practical examples that will help you address the challenges of writing efficient code Who This Book Is For To get the most out of this book, you need to have a working knowledge of reading and writing basic Haskell. No knowledge of performance, optimization, or concurrency is required. What You Will Learn Program idiomatic Haskell that's also surprisingly efficient Improve performance of your code with data parallelism, inlining, and strictness annotations Profile your programs to identify space leaks and missed opportunities for optimization Find out how to choose the most efficient data and control structures Optimize the Glasgow Haskell Compiler and runtime system for specific programs See how to smoothly drop to lower abstractions wherever necessary Execute programming for the GPU with Accelerate Implement programming to easily scale to the cloud with Cloud Haskell In Detail Haskell, with its power to optimize the code and its high performance, is a natural candidate for high performance programming. It is especially well suited to stacking abstractions high with a relatively low performance cost. This book addresses the challenges of writing efficient code with lazy evaluation and techniques often used to optimize the performance of Haskell programs. We open with an in-depth look at the evaluation of Haskell expressions and discuss optimization and benchmarking. You will learn to use parallelism and we'll explore the concept of streaming. We'll demonstrate the benefits of running multithreaded and concurrent applications. Next we'll guide you through various profiling tools that will help you identify performance issues in your program. We'll end our journey by looking at GPGPU, Cloud and Functional Reactive Programming in Haskell. At the very end there is a catalogue of robust library recommendations with code samples. By the end of the book, you will be able to boost the performance of any app and prepare it to stand up to real-world punishment. Style and approach This easy-to-follow guide teaches new practices and techniques to optimize your code, and then moves towards more advanced ways to effectively write efficient Haskell code. Small and simple practical examples will help you test the concepts yourself, and you will be able to easily adapt them for any application.

Learn to skyrocket the performance of your Rust applications
Writing High-Performance .NET Code, 2nd Edition
Haskell High Performance Programming
Performance Optimization of Numerically Intensive Codes
Writing High-Performance .Net Code
Julia High Performance

Become a better programmer with performance improvement techniques such as concurrency, lock-free programming, atomic

operations, parallelism, and memory management Key FeaturesLearn proven techniques from a heavyweight and recognized expert in C++ and high-performance computingUnderstand the limitations of modern CPUs and their performance impactFind out how you can avoid writing inefficient code and get the best optimizations from the compilerLearn the tradeoffs and costs of writing high-performance programsBook Description The great free lunch of "performance taking care of itself" is over. Until recently, programs got faster by themselves as CPUs were upgraded, but that doesn't happen anymore. The clock frequency of new processors has almost peaked, and while new architectures provide small improvements to existing programs, this only helps slightly. To write efficient software, you now have to know how to program by making good use of the available computing resources, and this book will teach you how to do that. The Art of Efficient Programming covers all the major aspects of writing efficient programs, such as using CPU resources and memory efficiently, avoiding unnecessary computations, measuring performance, and how to put concurrency and multithreading to good use. You'll also learn about compiler optimizations and how to use the programming language (C++) more efficiently. Finally, you'll understand how design decisions impact performance. By the end of this book, you'll not only have enough knowledge of processors and compilers to write efficient programs, but you'll also be able to understand which techniques to use and what to measure while improving performance. At its core, this book is about learning how to learn. What you will learnDiscover how to use the hardware computing resources in your programs effectivelyUnderstand the relationship between memory order and memory barriersFamiliarize yourself with the performance implications of different data structures and organizationsAssess the performance impact of concurrent memory accessed and how to minimize itDiscover when to use and when not to use lock-free programming techniquesExplore different ways to improve the effectiveness of compiler optimizationsDesign APIs for concurrent data structures and high-performance data structures to avoid inefficienciesWho this book is for This book is for experienced developers and programmers who work on performance-critical projects and want to learn new techniques to improve the performance of their code. Programmers in algorithmic trading, gaming, bioinformatics, computational genomics, or computational fluid dynamics communities will get the most out of the examples in this book, but the techniques are fairly universal. Although this book uses the C++ language, the concepts demonstrated in the book can be easily transferred or applied to other compiled languages such as C, Java, Rust, Go, and more.

Take performance to the next level! This book does not just teach you how the CLR works---it teaches you exactly what you need to do now to obtain the best performance today. It will expertly guide you through the nuts and bolts of extreme performance optimization in .NET, complete with in-depth examinations of CLR functionality, free tool recommendations and tutorials, useful anecdotes, and step-by-step guides to measure and improve performance. This second edition incorporates the advances and improvements in .NET over the last few years, as well as greatly expanded coverage of tools, more topics, more tutorials, more tips, and improvements throughout the entire book. New in the 2nd Edition: 50% increase in content! New examples, code samples, and

diagrams throughout entire book More ways to analyze the heap and find memory problems More tool coverage, including expanded usage of Visual Studio More benchmarking New GC configuration options Code warmup techniques New .NET features such as ref-returns, value tuples, SIMD, and more More detailed analysis of LINQ Tips for high-level feature areas such as ASP.NET, ADO.NET, and WPF Also find expanded coverage and discover new tips and tricks for: Profiling with multiple tools to quickly find problem areas Detailed description of the garbage collector, how to optimize your code for it, and how to diagnose difficult memory-related issues How to analyze JIT and diagnose warmup problems Effective use of the Task Parallel Library to maximize throughput Which .NET features and APIs to use and which to avoid Instrument your program with performance counters and ETW events Use the latest and greatest .NET features Build a performance-minded team ...and so much more

Performance Optimization of Numerically Intensive Codes offers a comprehensive, tutorial-style, hands-on, introductory and intermediate-level treatment of all the essential ingredients for achieving high performance in numerical computations on modern computers. The authors explain computer architectures, data traffic and issues related to performance of serial and parallel code optimization exemplified by actual programs written for algorithms of wide interest. The unique hands-on style is achieved by extensive case studies using realistic computational problems. The performance gain obtained by applying the techniques described in this book can be very significant. The book bridges the gap between the literature in system architecture, the one in numerical methods and the occasional descriptions of optimization topics in computer vendors' literature. It also allows readers to better judge the suitability of certain computer architecture to their computational requirements. In contrast to standard textbooks on computer architecture and on programming techniques the book treats these topics together at the level necessary for writing high-performance programs. The book facilitates easy access to these topics for computational scientists and engineers mainly interested in practical issues related to efficient code development.

Measure, optimize, and improve the performance of your Python code with this easy-to-follow guide About This Book Master the do's and don'ts of Python performance programming Learn how to use exiting new tools that will help you improve your scripts A step-by-step, conceptual guide to teach you how to optimize and fine-tune your critical pieces of code Who This Book Is For If you're a Python developer looking to improve the speed of your scripts or simply wanting to take your skills to the next level, then this book is perfect for you. What You Will Learn Master code optimization step-by-step and learn how to use different tools Understand what a profiler is and how to read its output Interpret visual output from profiling tools and improve the performance of your script Use Cython to create fast applications using Python and C Take advantage of PyPy to improve performance of Python code Optimize number-crunching code with NumPy, Numba, Parakeet, and Pandas In Detail Simply knowing how to code is not enough; on mission-critical pieces of code, every bit of memory and every CPU cycle counts, and knowing how to squish every bit of processing power out of your code is a crucial and sought-after skill. Nowadays, Python is used for many scientific projects, and sometimes the

calculations done in those projects require some serious fine-tuning. Profilers are tools designed to help you measure the performance of your code and help you during the optimization process, so knowing how to use them and read their output is very handy. This book starts from the basics and progressively moves on to more advanced topics. You'll learn everything from profiling all the way up to writing a real-life application and applying a full set of tools designed to improve it in different ways. In the middle, you'll stop to learn about the major profilers used in Python and about some graphic tools to help you make sense of their output. You'll then move from generic optimization techniques onto Python-specific ones, going over the main constructs of the language that will help you improve your speed without much of a change. Finally, the book covers some number-crunching-specific libraries and how to use them properly to get the best speed out of them. After reading this book, you will know how to take any Python code, profile it, find out where the bottlenecks are, and apply different techniques to remove them. Style and approach This easy-to-follow, practical guide will help you enhance your optimization skills by improving real-world code.
Modern Systems Programming with Scala Native
Build highly performant, multi-threaded, and concurrent applications using C# 7 and .NET Core 2.0
Simple and Practical Techniques for Writing Better Code