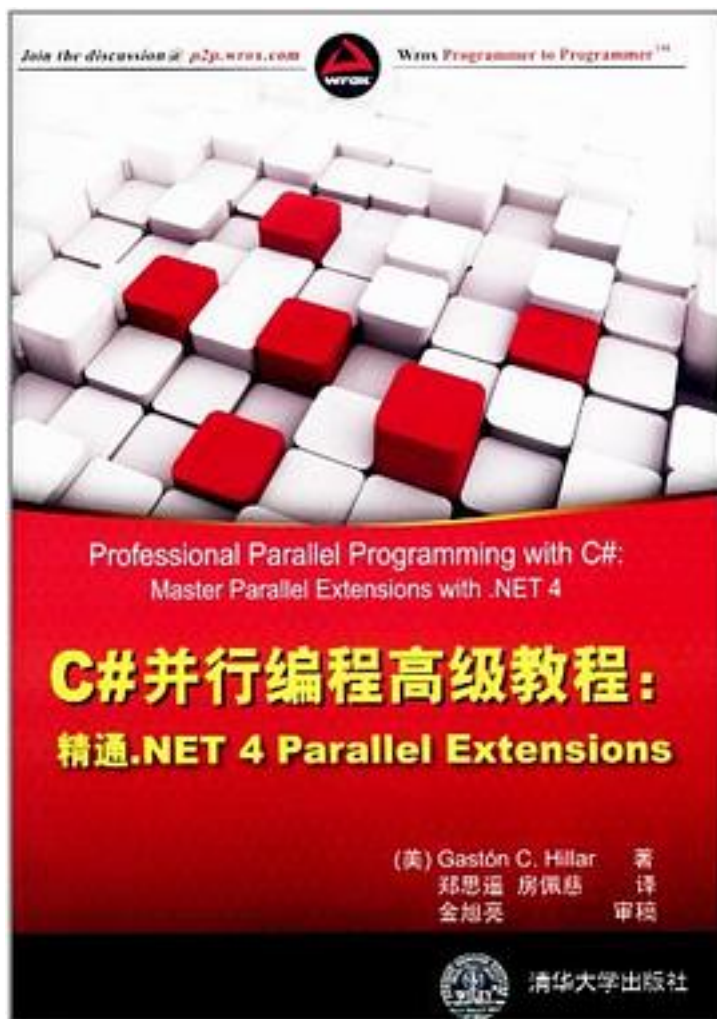


# C#并行编程高级教程



[C#并行编程高级教程 下载链接1](#)

著者:(美)希拉里

出版者:清华大学出版社

出版时间:2012-1

装帧:

isbn:9787302273561

您想淋漓尽致地发挥多核计算机系统的处理能力吗？《C#并行编程高级教程:精通NET 4

Parallel

Extensions》将帮助您实现这一夙愿。这本精品书籍浓墨重彩地描述如何使用C#

4、Visual Studio 2010和.NET Framework

4高效地创建基于任务的并行应用程序，详细讲述最新的单指令、多数据流指令和向量化等并行编程技术，介绍现代并行库，讨论如何珠联璧合地使用高级Intel工具与C#，并指导您巧妙使用新引入的轻型协调结构来开发自己的解决方案并解决最棘手的并发编程问题。

主要内容

- ◆介绍如何基于新Task Parallel Library和.NET 4设计稳定的可扩展并行应用程序。
- ◆讲解命令式数据并行、命令式任务并行、并发集合以及协调数据结构。
- ◆描述PLINQ高级声明式数据并行。
- ◆讨论如何使用新的Visual Studio 2010并行调试功能来调试匿名方法、任务和线程。
- ◆演示如何对数据源进行分区，以便在不同任务和线程之间合理分配工作负荷。

作者介绍:

从8岁起就开始使用计算机了。在20世纪80年代初，他开始在传奇的TexasTI-99/4A和Commodore64家用计算机上编写程序。他作为一名优秀毕业生在UADE大学获得了学士学位，然后又在UCEMA大学凭借出色的毕业论文获得了工商管理硕士学位。

自1997年以来，Gaston在并行编程、多处理器和多核处理器领域进行了深入研究。在设计和开发各种类型复杂的利用多核处理能力的并行解决方案方面，他有着14年的丰富经验，后来，他开始通过C#和.NETFramework编写并行解决方案。

目录: 目录

第1章 基于任务的程序设计 1

1.1 使用共享内存的多核系统 2

1.1.1 共享内存多核系统与分布式内存系统之间的区别 3

1.1.2 并程序设计和多核程序设计 4

1.2 理解硬件线程和软件线程 5

1.3 理解Amdahl法则 8

1.4 考虑Gustafson法则 11

1.5 使用轻量级并发模型 14

1.6 创建成功的基于任务的设计 15

1.6.1 以并发的思想指导设计 16

1.6.2 理解交错并发、并发和并行之间的区别 17

1.6.3 并行化任务 18

1.6.4 尽量减少临界区 18

1.6.5 理解多核并行程序的设计原则 19

1.7 为NUMA架构和更高的可扩展性做好准备 20

1.8 判断是否适合并行化 24

1.9 小结 25

第2章 命令式数据并行 27

2.1 加载并行任务 27

2.1.1 System.Threading.Tasks.Parallel类 29

2.1.2 Parallel.Invoke 30

- 2.2 将串行代码转换为并行代码 37
  - 2.2.1 检测可并行化的热点 37
  - 2.2.2 测量并行执行的加速效果 40
  - 2.2.3 理解并发执行 42
- 2.3 循环并行化 43
  - 2.3.1 Parallel.For 43
  - 2.3.2 Parallel.ForEach 49
  - 2.3.3 从并行循环中退出 56
- 2.4 指定并行度 62
  - 2.4.1 ParallelOptions 63
  - 2.4.2 计算硬件线程 65
  - 2.4.3 逻辑内核并不是物理内核 66
- 2.5 通过甘特图检测临界区 67
- 2.6 小结 68
- 第3章 命令式任务并行 69
  - 3.1 创建和管理任务 70
    - 3.1.1 System.Threading.Tasks.Task 71
    - 3.1.2 理解Task状态和生命周期 72
    - 3.1.3 通过使用任务来对代码进行并行化 74
    - 3.1.4 等待任务完成 80
    - 3.1.5 忘记复杂的线程 81
    - 3.1.6 通过取消标记取消任务 82
    - 3.1.7 从任务返回值 88
    - 3.1.8 TaskCreationOptions 90
    - 3.1.9 通过延续串联多个任务 90
    - 3.1.10 编写适应并发和并行的代码 95
  - 3.2 小结 96
- 第4章 并发集合 97
  - 4.1 理解并发集合提供的功能 98
    - 4.1.1 System.Collections.Concurrent 100
    - 4.1.2 ConcurrentQueue 101
    - 4.1.3 理解并行的生产者-消费者模式 104
    - 4.1.4 ConcurrentStack 116
    - 4.1.5 将使用数组和不安全集合的代码转换为使用并发集合的代码 121
    - 4.1.6 ConcurrentBag 122
    - 4.1.7 IProducerConsumerCollection 129
    - 4.1.8 BlockingCollection 129
    - 4.1.9 ConcurrentDictionary 143
  - 4.2 小结 147
- 第5章 协调数据结构 149
  - 5.1 通过汽车和车道理解并发难题 150
    - 5.1.1 非预期的副作用 150
    - 5.1.2 竞争条件 151
    - 5.1.3 死锁 152
    - 5.1.4 使用原子操作的无锁算法 153
    - 5.1.5 使用本地存储的无锁算法 154
  - 5.2 理解新的同步机制 156
  - 5.3 使用同步原语 157
    - 5.3.1 通过屏障同步并发任务 158
    - 5.3.2 屏障和ContinueWhenAll 164
    - 5.3.3 在所有的参与者任务中捕捉异常 165
    - 5.3.4 使用超时 166
    - 5.3.5 使用动态数目的参与者 171
  - 5.4 使用互斥锁 172

- 5.4.1 使用Monitor 176
- 5.4.2 使用锁超时 177
- 5.4.3 将代码重构为避免使用锁 180
- 5.5 将自旋锁用作互斥锁原语 183
  - 5.5.1 使用超时 186
  - 5.5.2 使用基于自旋的等待 187
  - 5.5.3 自旋和处理器出让 190
  - 5.5.4 使用volatile修饰符 193
- 5.6 使用轻量级的手动重置事件 194
  - 5.6.1 使用ManualResetEventSlim进行自旋和等待 194
  - 5.6.2 使用超时和取消 199
  - 5.6.3 使用ManualResetEvent 203
- 5.7 限制资源的并发访问 204
  - 5.7.1 使用SemaphoreSlim 205
  - 5.7.2 使用超时和取消 209
  - 5.7.3 使用 Semaphore 209
- 5.8 通过CountdownEvent简化动态fork和join场景 211
- 5.9 使用原子操作 215
- 5.10 小结 220
- 第6章 PLINQ：声明式数据并行 221
  - 6.1 从LINQ转换到PLINQ 222
    - 6.1.1 ParallelEnumerable及其AsParallel方法 224
    - 6.1.2 AsOrdered和orderby子句 225
  - 6.2 指定执行模式 228
  - 6.3 理解PLINQ中的数据分区 229
  - 6.4 通过PLINQ执行归约操作 234
  - 6.5 创建自定义的PLINQ聚合函数 235
  - 6.6 并发PLINQ任务 240
  - 6.7 取消PLINQ 243
  - 6.8 指定所需的并行度 245
    - 6.8.1 WithDegreeOfParallelism 245
    - 6.8.2 测量可扩展性 247
  - 6.9 使用ForAll 249
    - 6.9.1 foreach和ForAll的区别 250
    - 6.9.2 测量可扩展性 251
  - 6.10 通过WithMergeOptions配置返回结果的方式 253
  - 6.11 处理PLINQ抛出的异常 255
  - 6.12 使用PLINQ执行MapReduce算法 257
  - 6.13 使用PLINQ设计串行多步操作 259
  - 6.14 小结 261
- 第7章 Visual Studio 2010的任务调试能力 263
  - 7.1 充分利用多显示器的支持 264
  - 7.2 理解并行任务调试器窗口 267
  - 7.3 查看Parallel Stacks图 273
  - 7.4 跟踪并发代码 280
    - 7.4.1 调试匿名方法 288
    - 7.4.2 查看方法 290
    - 7.4.3 在源代码中查看线程 292
  - 7.5 检测死锁 294
  - 7.6 小结 300
- 第8章 线程池 301
  - 8.1 探究任务的底层技术 301
  - 8.2 理解新的CLR 4线程池引擎 303
    - 8.2.1 理解全局队列 303

- 8.2.2 等待工作线程完成工作 313
- 8.2.3 跟踪动态数目的工作线程 319
- 8.2.4 使用任务(而不是线程)将作业加入队列 323
- 8.2.5 理解任务和线程池之间的关系 326
- 8.2.6 理解局部队列和工作窃取算法 330
- 8.2.7 指定自定义的任务调度器 335
- 8.3 小结 340
- 第9章 异步编程模型 341
  - 9.1 结合使用异步编程和任务 342
    - 9.1.1 使用TaskFactory.FromAsync 343
    - 9.1.2 编写异步方法执行结束之后的延续 348
    - 9.1.3 合并多个并发异步操作的结果 349
    - 9.1.4 执行异步WPF UI更新 351
    - 9.1.5 执行异步Windows Forms UI更新 358
    - 9.1.6 创建执行EAP操作的任务 365
    - 9.1.7 使用TaskCompletionSource 373
  - 9.2 小结 377
- 第10章 并行测试和调优 379
  - 10.1 准备并行测试 379
    - 10.1.1 使用性能剖析功能 383
    - 10.1.2 测量并发性 385
  - 10.2 常见问题模式的解决方案 394
    - 10.2.1 串行化的执行 395
    - 10.2.2 锁争用 397
    - 10.2.3 锁封护 398
    - 10.2.4 申请超额 401
    - 10.2.5 申请不足 404
    - 10.2.6 分区问题 406
    - 10.2.7 工作站垃圾回收开销 408
    - 10.2.8 使用服务器垃圾回收 411
    - 10.2.9 I/O瓶颈 412
    - 10.2.10 主线程过载 413
  - 10.3 理解伪共享 416
  - 10.4 小结 419
- 第11章 向量化、SIMD指令以及其他并行库 421
  - 11.1 理解SIMD和向量化 421
  - 11.2 从MMX到SSE4.X和AVX 423
  - 11.3 使用Intel Math Kernel Library 425
    - 11.3.1 使用适用于多核的高度优化的软件函数 433
    - 11.3.2 将基于任务的编程和外部优化的库混合使用 434
    - 11.3.3 并行生成伪随机数 434
  - 11.4 使用Intel Integrated Performance Primitives 439
  - 11.5 小结 445
- 附录A .NET 4中与并行相关的类图 447
- 附录B 并发UML模型 461
- 附录C Parallel Extensions Extras 469
  - • • • • [\(收起\)](#)

[C#并行编程高级教程\\_下载链接1](#)

## 标签

C

#并行编程

C#并行编程

编程

计算机

程序设计

Programming

程序员修炼

## 评论

很详细的.Net并行编程书籍。多谢实例就好了。

-----  
国庆期间陪老爸输液，闲着无聊把这本书翻完了。不错的一本书，理论联系实际，看起来也不要吃力，就是例子有点傻。并行测试和调优那章不错。

-----  
讲述的深入浅出，受益匪浅

-----  
[C#并行编程高级教程 下载链接1](#)

## 书评

书内涵盖的内容： 现代多核和众核共享内存体系结构 介绍如何基于新task parallel library和.net 4设计稳定的可扩展并行应用程序。  
讲解命令式数据并行、命令式任务并行、并发集合以及协调数据结构。  
描述plinq高级声明式数据并行。 讨论如何使用新的visual stu...

-----  
[C#并行编程高级教程\\_下载链接1\\_](#)