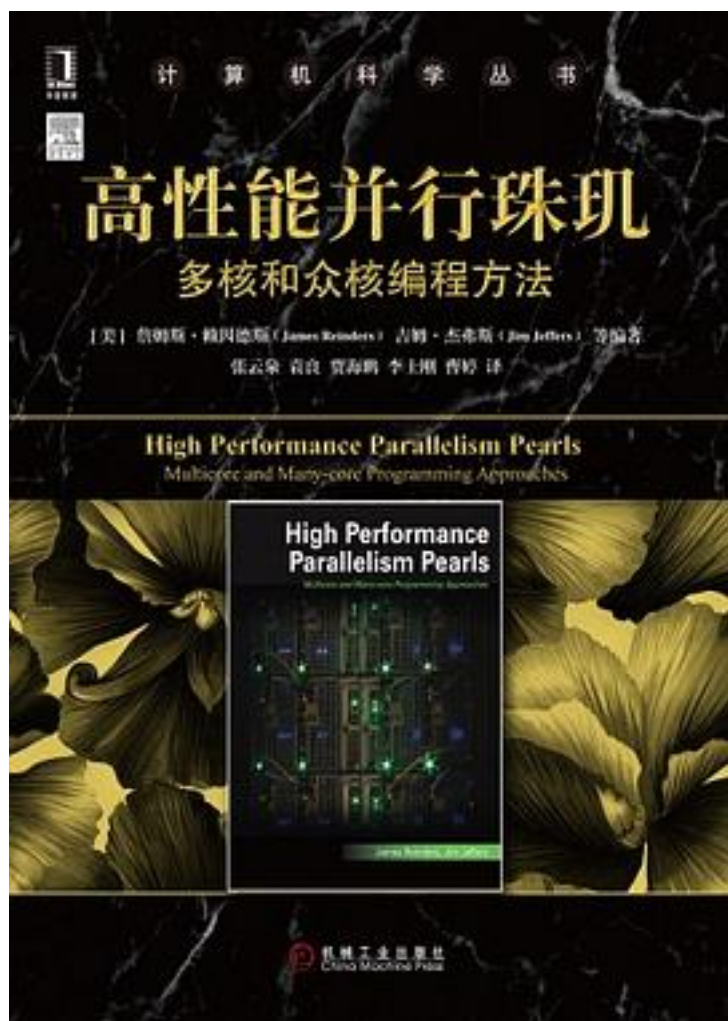


高性能并行珠玑



[高性能并行珠玑_下载链接1](#)

著者:James Reinders

出版者:机械工业出版社

出版时间:2017-11-3

装帧:平装

isbn:9787111580805

本书由英特尔的技术专家撰写，全面、系统地讲解在英特尔至强处理器和至强融核协处

理器上进行并行处理和编程的方法和技术。书中展示了如何在处理器和协处理器上进行并行处理和编程——展示了更好利用Intel Xeon Phi协处理器和Intel Xeon处理器或其他多核处理器的系统计算潜力的最有效的方法。全书包括大量来自多个行业 and 不同领域的成功并行编程工作的例子。每章既详细讲述所采用的编程技术，同时展示了其在Intel Xeon Phi协处理器和多核处理器上的高性能结果。几十个新的例子和案例显示的“成功经验”不仅展现了这些强大系统的主要特征，而且展示出如何在这些异构系统上保持并行化。

本书将使为Intel Xeon Phi产品开发高层并行性（包括最优编程）更加简单。Intel Xeon和Intel Xeon Phi系列之间的通用编程方法对整个科学和工程领域都是有利的，相同的程序可以实现多核和众核的并行可扩展性和向量化。

—— 选自推荐序，Sverre Jarpe, CERN

本书展示了如何借助同种编程方法来利用处理器和协处理器上的并行性，展示了如何有效地利用配备Intel Xeon Phi协处理器和Intel Xeon处理器或者其他多核处理器的系统的能力。本书包括多个行业和领域（如化学、工程以及环境科学）中成功的编程示例。每一章都包括所使用的编程技巧的详细讲解，并且展示了在Intel Xeon Phi协处理器和多核处理器上的高性能结果。这些示例不仅展示了这些高性能系统的特征，还给出了利用这些新型异构系统间并行性的方法。

本书特色

推动一致的基于标准的编程，展示多核处理器和Intel Xeon Phi协处理器上高性能编码的细节。

涵盖多个垂直领域的示例，展示真实应用代码的并行优化。

源代码可供下载，以便于未来进一步研究。

作者介绍:

詹姆斯·瑞恩德斯（James Reinders）Intel公司软件总监，首席技术布道师。参与多个旨在加强并行编程在工业界应用的工程研究和教育项目。他对多个项目做出了贡献，包括世界上首例 Teraflop 级超级计算机（ASCI Red）和世界上首例 Teraflop 级微处理器（Intel Xeon Phi协处理器）。吉姆·杰弗斯（Jim Jeffers）Intel公司技术计算组的首席工程师和工程经理。Jim与人合著了Intel Xeon Phi《Coprocesor High Performance Programming》协处理器高性能编程指南（Morgan Kaufmann, 2013）。目前，Jim是Intel技术计算可视化团队的领导

目录: 出版者的话

译者序

推荐序

前言

作者简介

第1章 引言 1

1.1 学习成功经验 1

1.2 代码现代化	1
1.3 并发算法现代化	1
1.4 向量化和数据局部性现代化	2
1.5 理解功耗使用	2
1.6 ISPC和OpenCL	2
1.7 Intel Xeon Phi协处理器特性	2
1.8 众核和新异构系统	2
1.9 书名中没有Xeon Phi与新异构架构编程	3
1.10 众核的未来	3
1.11 下载	3
1.12 更多信息	4
第2章 从正确到正确&高效：Godunov	
格式的Hydro2D案例学习	5
2.1 现代计算机上的科学计算	5
2.1.1 现代计算环境	6
2.1.2 CEA的Hydro2D	6
2.2 冲击流体动力学的一种数值方法	7
2.2.1 欧拉方程	7
2.2.2 Godunov方法	7
2.2.3 哪里需要优化	9
2.3 现代计算机架构的特征	9
2.3.1 面向性能的架构	9
2.3.2 编程工具和运行时	10
2.3.3 计算环境	11
2.4 通向高性能的路	11
2.4.1 运行Hydro2D	11
2.4.2 Hydro2D的结构	12
2.4.3 优化	15
2.4.4 内存使用	16
2.4.5 线程级并行	17
2.4.6 算术效率和指令级并行	24
2.4.7 数据级并行	26
2.5 总结	32
2.5.1 协处理器与处理器	32
2.5.2 水涨船高	32
2.5.3 性能策略	33
2.6 更多信息	34
第3章 HBM上的SIMD与并发优化	36
3.1 应用程序：HIROMB-BOOS-MODEL	36
3.2 关键应用：DMI	36
3.3 HBM执行配置文件	37
3.4 HBM优化综述	38
3.5 数据结构：准确定位位置	38
3.6 HBM上的线程并行	41
3.7 数据并行：SIMD向量化	45
3.7.1 零散的可优化部分	46
3.7.2 过早抽象是万恶之源	48
3.8 结果	50
3.9 详情分析	51
3.10 处理器与协处理器可扩展性对比	52
3.11 CONTIGUOUS属性	53
3.12 总结	54
3.13 参考文献	54
3.14 更多信息	55

第4章 流体动力学方程优化	56
4.1 开始	56
4.2 1.0版本：基础版本	57
4.3 2.0版本：线程盒	59
4.4 3.0版本：栈内存	63
4.5 4.0版本：分块	63
4.6 5.0版本：向量化	64
4.7 Intel Xeon Phi协处理器上的运行结果	68
4.8 总结	69
4.9 更多信息	70
第5章 分阶段准同步栅栏	71
5.1 如何改善代码	74
5.2 如何进一步改善代码	74
5.3 超线程方阵	74
5.4 关于该方案哪些地方不是最优的	75
5.5 超线程方阵编码	76
5.5.1 如何确定内核间兄弟线程和内核内HT线程	77
5.5.2 超线程方阵手动分区方法	77
5.5.3 吸取教训	79
5.6 回到工作	80
5.7 数据对齐	81
5.7.1 尽可能使用对齐的数据	81
5.7.2 冗余未必是件坏事	81
5.8 深入讨论分阶段准同步栅栏	84
5.9 如何节省时间	86
5.10 几个留给读者的优化思考	90
5.11 类似Xeon Phi协处理器的Xeon主机性能优化	91
5.12 总结	92
5.13 更多信息	92
第6章 故障树表达式并行求解	93
6.1 动机和背景	93
6.1.1 表达式	93
6.1.2 表达式选择：故障树	93
6.1.3 程序实例中的故障树：基本模拟	93
6.2 实例实现	94
6.3 其他因素	101
6.4 总结	101
6.5 更多信息	101
第7章 深度学习的数值优化	102
7.1 拟合目标函数	102
7.2 目标函数与主成分分析	105
7.3 软件及样例数据	106
7.4 训练数据	109
7.5 运行时间	109
7.6 扩展结果	111
7.7 总结	111
7.8 更多信息	112
第8章 优化聚集/分散模式	113
8.1 聚集/分散在Intel架构下的说明	114
8.2 聚集/分散模式在分子动力学中的应用	115
8.3 优化聚集/分散模式	117
8.3.1 提高时间和空间的局部性	117
8.3.2 选择一种适当的数据布局：AoS与SoA	118
8.3.3 AoS和SoA之间的动态转换	119

- 8.3.4 分摊聚集/分散和转换的开销 122
- 8.4 总结 123
- 8.5 更多信息 123
- 第9章 N体问题直接法的众核实现 125
 - 9.1 N体模拟 125
 - 9.2 初始解决方案 125
 - 9.3 理论极限 126
 - 9.4 降低开销和对齐数据 128
 - 9.5 优化存储层次 131
 - 9.6 改进分块 133
 - 9.7 主机端的优化 135
 - 9.8 总结 136
 - 9.9 更多信息 136
- 第10章 N体方法 137
 - 10.1 快速N体方法和直接N体内核 137
 - 10.2 N体方法的应用 138
 - 10.3 直接N体代码 138
 - 10.4 性能结果 141
 - 10.5 总结 142
 - 10.6 更多信息 142
- 第11章 使用OpenMP 4.0实现动态负载均衡 144
 - 11.1 最大化硬件利用率 144
 - 11.2 N体内核 146
 - 11.3 卸载版本 149
 - 11.4 第一个处理器与协处理器协作版本 150
 - 11.5 多协处理器版本 152
 - 11.6 更多信息 155
- 第12章 并发内核卸载 156
 - 12.1 设定上下文 156
 - 12.1.1 粒子动力学 156
 - 12.1.2 本章结构 157
 - 12.2 协处理器上的并发内核 158
 - 12.2.1 协处理器设备划分和线程关联 158
 - 12.2.2 并发数据传输 163
 - 12.3 在PD中使用并发内核卸载进行作用力计算 166
 - 12.3.1 使用牛顿第三定律并行评估作用力 166
 - 12.3.2 实现作用力并发计算 167
 - 12.3.3 性能评估：之前与之后 171
 - 12.4 总结 173
 - 12.5 更多信息 174
- 第13章 MPI和异构计算 175
 - 13.1 现代集群中的MPI 175
 - 13.2 MPI任务地点 176
 - 13.3 DAPL提供者的选择 180
 - 13.3.1 第一个提供者OFA-V2-MLX4_0-1U 180
 - 13.3.2 第二个提供者ofa-v2-scif0以及对节点内部结构的影响 180
 - 13.3.3 最后一个提供者 181
 - 13.3.4 混合程序的可扩展性 182
 - 13.3.5 负载均衡 184
 - 13.3.6 任务和线程映射 184
 - 13.4 总结 185
 - 13.5 致谢 185
 - 13.6 更多信息 185
- 第14章 Intel Xeon Phi协处理器功耗分析 186

- 14.1 功耗分析 186
- 14.2 用软件测量功耗和温度 187
 - 14.2.1 创建功耗和温度监控脚本 188
 - 14.2.2 使用micsmc工具创建功耗和温度记录器 189
 - 14.2.3 使用IPMI进行功耗分析 190
- 14.3 基于硬件的功耗分析方法 192
- 14.4 总结 196
- 14.5 更多信息 196
- 第15章 集成Intel Xeon Phi协处理器至集群环境 197
 - 15.1 早期探索 197
 - 15.2 Beacon系统的历史 197
 - 15.3 Beacon系统的架构 198
 - 15.3.1 硬件环境 198
 - 15.3.2 软件环境 198
 - 15.4 Intel MPSS安装步骤 199
 - 15.4.1 系统准备 199
 - 15.4.2 安装Intel MPSS栈 200
 - 15.4.3 生成和定制配置文件 201
 - 15.4.4 MPSS升级 204
 - 15.5 建立资源和工作负载管理器 204
 - 15.5.1 TORQUE 204
 - 15.5.2 序言程序 205
 - 15.5.3 尾声程序 206
 - 15.5.4 TORQUE/协处理器集成 207
 - 15.5.5 Moab 207
 - 15.5.6 提高网络局部性 207
 - 15.5.7 Moab/协处理器集成 207
 - 15.6 健康检查和监控 208
 - 15.7 常用命令脚本化 209
 - 15.8 用户软件环境 210
 - 15.9 今后的方向 211
 - 15.10 总结 212
 - 15.11 致谢 212
 - 15.12 更多信息 212
- 第16章 在Intel Xeon Phi协处理器上支持集群文件系统 214
 - 16.1 网络配置概念和目标 214
 - 16.1.1 网络选项概览 215
 - 16.1.2 设置集群启用协处理器的步骤 216
 - 16.2 协处理器文件系统支持 217
 - 16.2.1 支持NFS 217
 - 16.2.2 支持Lustre文件系统 218
 - 16.2.3 支持Fraunhofer BeeGFS文件系统 219
 - 16.2.4 支持Panasas PanFS文件系统 220
 - 16.2.5 集群文件系统的选择 220
 - 16.3 总结 220
 - 16.4 更多信息 221
- 第17章 NWChem: 大规模量子化学仿真 222
 - 17.1 引言 222
 - 17.2 回顾单线程CC形式 222
 - 17.3 NWChem软件架构 225
 - 17.3.1 全局数组 225
 - 17.3.2 张量收缩引擎 226
 - 17.4 设计卸载解决方案 226
 - 17.5 卸载架构 229

- 17.6 内核优化 230
- 17.7 性能评估 232
- 17.8 总结 233
- 17.9 致谢 235
- 17.10 更多信息 235
- 第18章 大规模多系统上的高效嵌套并行 238
 - 18.1 动机 238
 - 18.2 基准测试 238
 - 18.3 基线基准测试 239
 - 18.4 流水线方法——Flat_arena类 240
 - 18.5 Intel TBB用户管理任务调度平台 241
 - 18.6 分层方法——Hierarchical_arena类 243
 - 18.7 性能评估 243
 - 18.8 对NUMA架构的影响 245
 - 18.9 总结 246
 - 18.10 更多信息 246
- 第19章 Black-Scholes定价的性能优化 248
 - 19.1 金融市场模型基础及Black-Scholes公式 248
 - 19.1.1 金融市场数学模型 248
 - 19.1.2 欧式期权和公平价格概念 249
 - 19.1.3 Black-Scholes公式 250
 - 19.1.4 期权定价 250
 - 19.1.5 测试平台架构 250
 - 19.2 案例研究 251
 - 19.2.1 初始版本——检验正确性 251
 - 19.2.2 参照版本——选择合适的数据结构 251
 - 19.2.3 参照版本——不要混合使用数据类型 252
 - 19.2.4 循环向量化 253
 - 19.2.5 使用快速数学函数：erff()与cdfnormf() 255
 - 19.2.6 代码等价变换 256
 - 19.2.7 数组对齐 257
 - 19.2.8 尽可能降低精度 258
 - 19.2.9 并行工作 259
 - 19.2.10 使用热身 260
 - 19.2.11 使用Intel Xeon Phi协处理器实现轻松移植 261
 - 19.2.12 使用Intel Xeon Phi协处理器实现并行工作 261
 - 19.2.13 使用Intel Xeon Phi协处理器和流存储 262
 - 19.3 总结 263
 - 19.4 更多信息 264
- 第20章 使用Intel COI库传输数据 265
 - 20.1 使用Intel COI库的第一步 265
 - 20.2 COI缓冲区种类和传输性能 266
 - 20.3 应用程序 269
 - 20.4 总结 270
 - 20.5 更多信息 270
- 第21章 高性能光线追踪 271
 - 21.1 背景 272
 - 21.2 向量化的光线遍历 272
 - 21.3 Embree光线追踪内核 273
 - 21.4 在应用程序中使用Embree 274
 - 21.5 性能 276
 - 21.6 总结 277
 - 21.7 更多信息 278
- 第22章 OpenCL程序的可移植性能 279

22.1 两难的困境	279
22.2 OpenCL简介	280
22.3 OpenCL示例：矩阵乘	282
22.4 OpenCL与Intel Xeon Phi协处理器	285
22.5 性能评估	285
22.6 案例研究：分子对接算法	287
22.7 性能评估：性能可移植性	289
22.8 相关工作	291
22.9 总结	291
22.10 更多信息	291
第23章 应用到Stencil计算中的特性提取和优化方法	292
23.1 引言	292
23.2 性能评估	293
23.2.1 测试平台的AI	293
23.2.2 内核的AI	294
23.3 标准优化	296
23.3.1 自动应用调试	301
23.3.2 自动调试工具	304
23.3.3 结果	305
23.4 总结	305
23.5 更多信息	307
第24章 剖析指导优化	308
24.1 计算机科学中的矩阵转置	308
24.2 工具和方法	309
24.3 串行：初始的就地转置实现	310
24.4 并行：使用OpenMP增加并行度	313
24.5 分块：提高数据局部性	315
24.6 规范化：多版本微内核	319
24.7 预组织：释放更多的并行性	322
24.8 总结	326
24.9 更多信息	327
第25章 基于ITAC的异构MPI应用优化	328
25.1 亚式期权定价	328
25.2 应用设计	329
25.3 异构集群中的同步	330
25.4 通过ITAC寻找性能瓶颈	331
25.5 建立ITAC	331
25.6 非均衡的MPI运行	332
25.7 手动负载均衡	335
25.8 动态老板-工人负载均衡	337
25.9 结论	339
25.10 更多信息	340
第26章 集群上可扩展OOC解法器	341
26.1 引言	341
26.2 基于ScaLAPACK的OOC分解算法	342
26.2.1 核内分解	342
26.2.2 OOC分解	343
26.3 从NVIDIA GPU移植到Intel Xeon Phi协处理器	344
26.4 数值结果	346
26.5 结论和展望	350
26.6 致谢	350
26.7 更多信息	350
第27章 稀疏矩阵向量乘：并行化和向量化	352
27.1 引言	352

27.2 稀疏矩阵数据结构	353
27.2.1 压缩后的数据结构	354
27.2.2 分块	356
27.3 并行SpMV乘法	356
27.3.1 部分分布式并行SpMV	356
27.3.2 完全分布式并行SpMV	357
27.4 Intel Xeon Phi协处理器的向量化	358
27.5 评估	362
27.5.1 Intel Xeon Phi协处理器	363
27.5.2 Intel Xeon处理器	365
27.5.3 性能比较	366
27.6 总结	366
27.7 致谢	367
27.8 更多信息	367
第28章 基于Morton排序的性能优化	368
28.1 通过数据重排提高缓存局部性	368
28.2 性能改进	368
28.3 矩阵转置	369
28.4 矩阵乘法	373
28.5 总结	377
28.6 更多信息	378
• • • • •	(收起)

[高性能并行珠玑 下载链接1](#)

标签

多核编程

计算机科学

数值计算

并行计算

并行

Concurrency

评论

整本书不能说没有价值，但第一章翻译的那么垃圾，负责这章的译者可以开除了，你就算看不懂学着原版排版的有结构一点总能做到吧。。。顺便说一下这本书原版还有第二卷

[高性能并行珠玑 下载链接1](#)

书评

[高性能并行珠玑 下载链接1](#)