

<<多核计算与程序设计>>

图书基本信息

书名：<<多核计算与程序设计>>

13位ISBN编号：9787560950969

10位ISBN编号：7560950965

出版时间：2009-3

出版时间：华中科技大学

作者：周伟明

页数：656

版权说明：本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问：<http://www.tushu007.com>

<<多核计算与程序设计>>

前言

我的工作要求我跟踪、了解软件技术各领域的重要进展。

为此，除了频繁阅读最新的技术新闻，请教于各路高人，我还必须经常留心市面上的技术图书。

大约是在2006年5月，我在中关村图书大厦看到一本《多任务下的数据结构与算法》，作者周伟明，这本书当即引起了我的兴趣。

自从2005年初Herb Sutter发表名为《免费午餐已经结束》的著名文章以来，多核计算就一直是整个技术社群关注的热点问题之一。

不过，大家的注意力更多地集中在诸如Java . concurrent、MPI、OpenMP、Intel TBB等程序库和Pipelining、MapReduce等编程模式上，希望能够借助一些工具来规避并发程序设计所带来的智力挑战。

毫无疑问，由于并发程序所具有的复杂性和各种“诡异”问题，在软件项目实施中，应尽可能利用现成的模式和工具，而要对“土法炼钢”式的“创新”保持审慎。

然而，这并不意味着开发者可以轻轻地“站在巨人肩膀上”；恰恰相反，如果没有自下而上地对并发多任务程序设计下一番苦功进行研究的话，这些模式和工具在我们手中反而可能成为无知和错误的放大器，贻害无穷。

《多任务下的数据结构与算法》这本书的出发点，即是以并发和多任务的角度重新审视算法与数据结构这样最基本、最重要的知识，从根本上帮助读者建立对多任务程序本质的正确观念、知识体系和关键技能，对此我是非常认可的。

<<多核计算与程序设计>>

内容概要

本书主要介绍适应于多核（或多处理器）计算机系统的算法和程序，共分为五个部分进行讲解。

第1部分介绍多核编程的基础知识，包括多核编程常见问题、锁竞争、加速比、负载均衡等基本概念，多线程退出算法、读写锁、旋转锁、原子操作等多线程编程基础知识，基于OpenMP标准的并行程序设计基础等；第2部分介绍基础的数据结构与算法，包括数组、链表、哈希表、二叉树、AVL树、复合二叉树等基本数据结构，在链表那章中还讲解了多线程并行遍历的基本方法。

第3部分介绍多核并行计算方面的基础知识，并行编程包括常用的编程模式如分治模式、流水线模式、任务图分解与调度模式、动态任务调度模式等，并行搜索包括顺序搜索及终止检测算法，并行最短路径搜索等，并行排序包括并行快速排序、并行归并排序、并行基数排序等，并行数值计算包括并行矩阵乘法、并行前缀和计算等方面的内容。

本部分介绍的各种并行算法和程序中，重点介绍如何解决多核系统中的计算随CPU核数的扩展性，CPU Cache伪共享方面的问题。

第4部分介绍多核共享资源计算方面的内容，也是本书中最重要的内容，讲解了分布式计算设计模式如线程分组竞争模式、条件同步模式、批量私有化处理模式、数据本地化模式等。

这部分中讲解了本书中几个最重要的程序：分布式队列中实现了自动让每个线程带有一个本地队列、分布式查找中介绍了分段锁的哈希表、动态负载均衡的分布式查找等，分布式内存管理则介绍了适应多核的内存管理方案，尤其是基于抢夺式的分布式内存管理算法，在分配和释放共享内存时也几乎不需要使用锁，性能优异。

第5部分介绍任务分解与调度方面的知识，这也是本书中最重要的内容，包括任务图分解与调度的实现方法，动态任务分解与调度的实现方法等。

其中还介绍了使用动态嵌套任务调度进行并行计算的方法，给出了用动态嵌套任务调度实现ParallelForo、并行快速排序、并行归并的实例。

最后一章中还介绍了Lock-Free编程（使用CAS原子操作进行编程）的基础知识，如ABA问题，内存删除问题等，并给出了一个Lock-Free的队列的实现实例。

<<多核计算与程序设计>>

作者简介

周伟明，1994年毕业于上海交通大学，曾就职于美国加州的DASCOM Inc. 公司和华为技术有限公司等企业。
担任过网络安全软件、网络服务器软件、机器翻译软件、工具软件、嵌入式系统软件等研发工作，亲自编写过的源代码逾40万行。
著有《多任务下的数据结构与算法》（华中科技大学

<<多核计算与程序设计>>

书籍目录

第1部分 基础知识 1 多核计算概述 1.1 多核CPU概述 1.1.1 多核计算将成为发展趋势 1.1.2 多核CPU硬件架构介绍 1.1.3 多核给程序员带来的机遇和挑战 1.2 多核编程会遇到那些问题 1.2.1 并发性问题 1.2.2 CPU饥饿问题 1.2.3 任务的分解与调度问题 1.2.4 加速比性能问题 1.2.5 节能环保问题 1.2.6 扩展性问题 1.3 多核编程与单核多线程编程的区别 1.3.1 锁竞争导致的串行化的区别 1.3.2 线程分解与执行的区别 1.3.3 CPU核负载均衡的区别 1.3.4 任务调度策略的区别 1.3.5 CPU Cache存取的区别(伪共享问题) 1.3.6 任务优先级抢占的区别 1.3.7 串行计算与并行及分布式计算的区别 1.4 多核编程与多机分布式编程的区别 1.4.1 共享存储与分布式存储的区别 1.4.2 分布式计算的区别 1.4.3 编程环境上的区别 1.5 加速比系数 1.5.1 阿姆达尔定律 1.5.2 Gustafson定律 1.5.3 阿姆达尔定律和Gustafson定律的等价性 1.5.4 Karp-Flatt度量 1.5.5 实际情况中影响加速比系数的因素 1.5.6 并行计算开销情况下的加速比 1.6 锁竞争问题及对加速比的影响 1.6.1 线程粒度因子与锁粒度因子 1.6.2 锁竞争的性能情况 1.6.3 集中式锁竞争中的加速比分析 1.6.4 随机锁竞争中的加速比分析 1.6.5 分布式锁竞争的加速比分析 1.6.6 无锁编程的加速比分析 1.7 负载均衡问题对加速比的影响 1.7.1 影响负载均衡的主要因素 1.7.2 负载均衡的评价指标 1.7.3 负载均衡情况下的加速比 1.8 参考文献 2

多线程编程基础 2.1 多线程编程基本概念 2.1.1 线程 2.1.2 锁 2.1.3 各种系统中常用的锁操作及信号量操作函数 2.1.4 用C++实现锁的自动释放 2.1.5 原子操作 2.1.6 锁与原子操作的区别 2.1.7 有锁计算、无锁计算与本地计算的概念 2.2 各种锁性能比较 2.2.1 各种锁在单线程情况下的性能 2.2.2 各种锁在多线程集中式锁竞争情况下的性能 2.2.3 各种锁在多线程分布式锁竞争情况下的性能 2.3 读写锁算法 2.3.1 读写锁概念的引出 2.3.2 读写锁算法的分析和实现 2.3.3 读写锁的编码实现 2.4 多线程退出算法 2.4.1 单个子线程退出算法 2.4.2 多个线程访问共享资源时的退出 2.4.3 有锁的多线程资源释放退出算法实现 2.4.4 无锁的退出算法 2.4.5 多线程退出算法的使用 2.5 参考文献 3 OpenMP程序设计 3.1 OpenMP基本概念 3.1.1 fork/join并行执行模式的概念 3.1.2 内存模型 3.1.3 性能例子 3.1.4 编译器对OpenMP的支持 3.2 OpenMP编程模型 3.2.1 OpenMP编译指导语句格式 3.2.2 OpenMP主要命令 3.2.3 OpenMP主要子句 3.2.4 OpenMP主要库函数 3.3 线程创建与工作分摊 3.3.1 parallel命令 3.3.2 for和parallel for命令 3.3.3 if子句(条件执行并行) 3.3.4 动态设置并行循环的线程数量 3.3.5 循环并行化的问题 3.3.6 sections和section命令 3.3.7 single命令 3.3.8 master命令 3.4 数据处理 3.4.1 private子句 3.4.2 firstprivate子句 3.4.3 lastprivate子句 3.4.4 threadprivate子句 3.4.5 shared子句 3.4.6 default子句 3.4.7 reduction子句 3.4.8 copyin子句 3.4.9 copyprivate子句 3.5 任务调度 3.5.1 Schedule子句用法 3.5.2 静态调度(static) 3.5.3 动态调度(dynamic) 3.5.4 guided调度(guided) 3.5.5 runtime调度(runtime) 3.5.6 任务调度与伪共享问题 3.6 线程间的同步 3.6.1 barrier命令 3.6.2 critical命令 3.6.3 atomic命令 3.6.4 ordered命令和子句 3.6.5 nowait子句 3.6.6 flush命令 3.7 OpenMP库函数详解 3.7.1 执行环境函数 3.7.2 锁操作函数 3.7.3 时间操作函数 3.8 OpenMP环境变量 3.8.1 OMP_DYNAMIC 3.8.2 OMP_NUM_THREADS 3.8.3 OMP_NESTED 3.8.4 OMP_SCHEDULE 3.9 OpenMP内部控制变量及相关流程 3.9.1 内部控制变量 3.9.2 任务调度流程 3.9.3 线程数量决定流程 3.10 参考文献 第2部份 基础数据结构与算法 4 数组 4.1 栈 4.1.1 栈的基本概念 4.1.2 栈的编码实现 4.1.3 多线程栈的实现 4.2 对数组进行快速排序 4.2.1 排序算法介绍 4.2.2 串行快速排序基本思想 4.2.3 串行快速排序的代码实现 4.2.4 非递归的快速排序算法 4.2.5 快速排序算法的复杂度分析 4.3 对数组进行查找 4.3.1 顺序查找 4.3.2 二分查找 4.4 实例:用数组管理一个HOOK功能 4.4.1 单个函数的HOOK实现 4.4.2 多个函数的HOOK实现 4.4.3 HOOK功能的应用简介 4.4.4 HOOK使用的注意事项 4.5 参考文献 5 链表 5.1 单向链表 5.1.1 存储表示 5.1.2 接口设计 5.1.3 添加节点到链表头部 5.1.4 基本功能编码实现 5.2 单向链表的排序 5.2.1 插入排序 5.2.2 归并插入排序 5.3 双向链表 5.3.1 双向链表的基本概念 5.3.2 双向链表的设计 5.3.3 双向链表的操作接口 5.3.4 双向链表的编码实现 5.4 链表的逐个节点遍历 5.4.1 逐个节点遍历基本概念 5.4.2 逐个节点遍历编码实现

<<多核计算与程序设计>>

- 5.5 多线程遍历算法 5.5.1 多线程链表的设计和编码实现 5.5.2 多线程链表的4种遍历方案
5.5.3 多个线程同时遍历的情况 5.6 实例：使用链表管理短信息系统的CACHE 5.6.1 短信息系统的CACHE管理基本概念 5.6.2 短信息系统的发送和接收分析 5.6.3 短信息系统CACHE管理的编码实现 6 哈希表 6.1 哈希表 6.1.1 哈希表的基本概念 6.1.2 哈希表的索引方法 6.1.3 哈希表的冲突解决方法 6.1.4 哈希表基本操作的源代码 6.2 哈希链表 6.2.1 哈希表和数组、链表的效率比较 6.2.2 时间效率和空间效率的关系 6.2.3 哈希链表的基本概念 6.2.4 哈希链表的操作 6.2.5 哈希链表的编码实现 6.3 实例：WebServer的动态CACHE文件管理 6.3.1 WebServer的动态CACHE文件管理基本概念 6.3.2 CACHE文件管理功能的设计 6.3.3 CACHE文件管理功能的编码实现 6.4 参考文献 7 普通树与二叉树 7.1 普通树 7.1.1 普通树的描述方法 7.1.2 树的操作接口设计 7.1.3 树的遍历算法 7.1.4 树的编码实现 7.1.5 使用树的遍历算法来实现Xcopy功能 7.2 二叉树 7.2.1 二叉树的基本概念 7.2.2 二叉树的树梢及二叉树的高度 7.2.3 二叉树的描述方法 7.3 二叉排序树 7.3.1 二叉排序树的基本概念 7.3.2 二叉排序树的查找 7.3.3 二叉排序树的插入 7.3.4 二叉排序树的删除 7.3.5 二叉排序树的遍历 7.3.6 二叉排序树的旋转操作 8 AVL搜索树 8.1 AVL搜索树的基本概念 8.2 AVL搜索树的插入 8.2.1 插入操作需要考虑的问题 8.2.2 不存在不平衡节点的情况分析 8.2.3 不平衡A节点的情况分析 8.2.4 存在不平衡节点的四种情况分析 8.2.5 LL型不平衡情况的调整 8.2.6 LR型不平衡情况的调整 8.2.7 插入操作的伪代码描述 8.3 AVL搜索树的删除 8.3.1 A节点的确定 8.3.2 几种不平衡情况的分析 8.3.3 L0型调整分析 8.3.4 L-1型调整分析 8.3.5 L1型调整分析 8.3.6 删除操作的伪代码描述 8.4 负载平衡的AVL树 8.4.1 基本概念的引出 8.4.2 插入操作中负载因子的调整 8.4.3 删除操作中负载因子的调整 8.4.4 L0和L-1型调整分析 8.4.5 L1型调整分析 8.5 AVL树的源代码 8.5.1 数据结构定义 8.5.2 创建、释放、查找等操作 8.5.3 旋转操作函数 8.5.4 插入操作函数 8.5.5 删除操作函数 8.6 参考文献 9 复合二叉树 9.1 哈希红黑树 9.1.1 哈希红黑树的基本概念 9.1.1 哈希红黑树的查找 9.1.3 哈希红黑树的插入 9.1.4 哈希红黑树的删除 9.1.5 哈希红黑树的释放 9.1.6 哈希红黑树的遍历 9.1.7 哈希红黑树的编码实现 9.1.8 哈希红黑树的效率分析 9.2 哈希AVL树 9.2.1 哈希AVL树的基本概念 9.2.2 哈希AVL树的查找 9.2.3 哈希AVL树的插入 9.2.4 哈希AVL树的删除 9.2.5 哈希AVL树的释放 9.2.6 哈希AVL树的遍历 9.2.7 哈希AVL树的编码实现 9.3 复合数据结构的分类 9.4 抗DoS/DdoS攻击的实例 9.4.1 DoS/DdoS攻击的概念 9.4.2 常见DoS/DdoS攻击手段及防范策略 9.4.3 抗DoS/DdoS攻击的实现 9.4.4 抗DoS/DdoS攻击的编码实现 第3部分 并行计算 10 并程序序设计模式 10.1 基本概念 10.1.1 强并行计算与弱并行计算 10.1.2 并程序序设计模式的基本思路 10.2 模式数据分解模式 10.3 分治模式 10.3.1 子问题求解时的负载平衡问题 10.3.2 子问题的解的合并可能引起的串行化问题 10.4 流水线模式 10.5 任务并行模式 10.6 任务调度模式 10.6.1 任务图调度模式 10.6.2 动态任务调度模式 11 并行搜索 11.1 并行顺序搜索 11.1.1 并行搜索指定数据 11.1.2 并行搜索最大数 11.1.3 终止检测算法 11.2 串行Dijkstra最短路径搜索 11.2.1 Dijkstra最短路径算法的描述 11.2.2 Dijkstra最短路径算法的过程图解 11.2.3 伪代码描述 11.2.4 算法流程图 11.2.5 C/C++代码实现 11.3 并行最短路径算法 11.3.1 Dijkstra算法的并行化 11.3.2 并行Dijkstra算法的代码实现 11.3.3 其他并行最短路径算法的介绍和分析 11.4 参考文献 12 并行排序 12.1 并行排序概述 12.2 冒泡排序 12.2.1 串行冒泡排序 12.2.2 奇偶排序 12.3 快速排序 12.3.1 串行快速排序基本思想 12.3.2 串行快速排序的代码实现 12.3.3 快速排序并行化方法 12.3.4 开源项目mcsstl中的并行快速排序 12.3.5 基于任务窃取的快速排序 12.4 并行归并排序 12.4.1 串行归并算法 12.4.2 Cole并行归并算法 12.4.3 并行快速归并排序 12.5 基数排序 12.5.1 串行链式基数排序 12.5.2 串行数组基数排序 12.5.3 一步到位的分层排序 12.5.4 负载平衡的并行基数排序 12.5.5 分区的并行基数排序 13 并行数值计算 13.1 多核并行数值计算面临的

<<多核计算与程序设计>>

问题 13.1.1 Cache的命中率问题 13.1.2 伪共享问题 13.2 求和及前缀求和 13.3 矩阵相加
13.4 矩阵相乘 13.4.1 基本概念 13.4.2 串行算法 13.4.3 并行算法 13.5 矩阵向量相乘 13.6
并行随机数生成 13.7 参考文献 第4部分 共享资源分布式计算 14 分布式计算设计模式 14.1 基本概
念 14.1.1 共享资源的计算分解 14.1.2 共享资源计算的负载均衡问题 14.1.3 共享资源计算的算
法设计思路与方法 14.2 线程分组竞争模式 14.2.1 标准的线程分组竞争模式 14.2.2 线程分组竞
争模式的变种 14.3 线程随机竞争模式 14.3.1 基本概念 14.3.2 加速比性能的保证 14.4 数据
本地化模式 14.4.1 取得比单核多线程更好的性能 14.4.2 数据本地化模式 14.4.3 优缺点分析
14.5 分布式数据结构设计 14.5.1 复合数据结构设计方法 14.5.2 分布式数据结构设计 14.5.3 分
布式数据结构主要问题 14.6 参考文献 15 分布式队列 15.1 串行队列 15.1.1 简单环形队列
15.1.2 STL中的Deque 15.1.3 动态环形队列 15.2 队列池 15.2.1 共享队列 15.2.2 消息队列
15.2.3 队列池 15.2.4 队列池的几种实现方案 15.2.5 队列池的使用实例 15.3 带本地计算的分布
式队列 15.3.1 基本思想 15.3.2 本地化队列的实现 15.3.3 任务偷取队列的实现 15.3.4 分布
式队列的实现 15.3.5 线程池CThreadPool的实现 15.3.6 线程池CThreadPool的代码实现 15.3.7
CDistributedQueue源代码 15.3.8 CDistributedQueue的使用实例 16 分布式查找 16.1 多核中查找的
问题与主要思路 16.2 静态负载平衡的二级查找结构设计 16.2.1 二级查找结构设计 16.2.2 分布
式哈希AVL树 16.2.3 分布式顺序AVL树 16.3 动态负载平衡的多级查找结构设计 16.3.1 分布式
查找中的负载均衡问题 16.3.2 多级查找结构设计方法 16.3.3 多级查找表的查找算法 16.3.4 多
级查找表的插入操作算法 16.3.5 多级查找表的删除操作算法 16.3.6 多级顺序表 16.3.7 多级索
引AVL树 16.3.8 分布式哈希多级AVL树 16.3.9 分布式顺序多级AVL树 16.4 多核环境中查找算
法的选用方法 16.5 动态WebCache设计实例 17 分布式内存管理 17.1 多核内存管理的基本思想
17.1.1 内存管理方面的需求 17.1.2 多核系统中的内存管理思路 17.2 等尺寸内存管理 17.2.1
Freelist内存管理基本概念 17.2.2 Freelist编码实现 17.2.3 FreeLists内存管理 17.3 Intel 开源项
目TBB中的内存管理 17.3.1 伪共享问题 17.3.2 Cache对齐的内存管理 17.3.3 数据结构 17.3.4
将内存管理器映射到线程 17.3.5 分配和释放算法 17.3.6 线程退出时的内存回收 17.4 抢夺式内
存管理算法 17.4.1 算法基本思想 17.4.2 碎片重组回收利用技术 17.4.3 抢夺式算法的详细算法
流程 17.4.4 代码实现 17.5 伪共享问题的深入分析 17.5.1 内存释放时的伪共享问题 17.5.2
伪共享问题的概率分析 17.5.3 用户程序使用内存过程中的伪共享问题 17.5.4 分布式内存管理的
进一步改进措施 17.6 参考文献 第5部分 任务分解与调度 18 任务图分解与调度 18.1 任务分解与调
度的问题 18.1.1 使用OpenMP调度的问题 18.1.2 任务图调度模型 18.1.3 任务图调度算法简介
18.2 任务组调度算法 18.2.1 基本思路 18.2.2 任务组调度算法 18.2.3 算法流程图 18.2.4 数
据结构与接口设计 18.2.5 代码实现 18.2.6 任务组调度的应用分析 18.2.7 误差下降调度算法
18.3 任务图调度算法 18.3.1 任务图的分层算法 18.3.2 分层算法过程图解 18.3.3 数据结构和接
口设计 18.3.4 分层算法的代码实现 18.3.5 任务调度器的代码实现 18.3.6 实例：任务图调度器
的使用 18.4 手工任务分解的原则和方法 18.4.1 任务间负载均衡的影响因素 18.4.2 任务分解原
则和方法 18.5 参考文献 19 动态任务分解与调度 19.1 动态任务分解的两种类型 19.2 非嵌套型
动态任务调度 19.2.1 网络服务器软件中的任务调度 19.2.2 使用分布式队列的调度方法 19.2.3
CTaskScheduler的设计 19.2.4 CTaskScheduler的代码实现 19.3 嵌套型动态任务调度 19.3.1 基本
思想 19.3.2 CNestTaskScheduler的设计 19.3.3 CNestTaskScheduler的代码实现 19.3.4
CNestTaskScheduler使用方法 19.4 实例：用任务调度器实现parallel_for 19.4.1 parallel_for的实现
19.4.2 用parallel_for进行并行快速排序 19.4.3 用parallel_for进行并行归并 19.5 参考文献 20 Lock-Free
编程基础 20.1 Lock-Free编程基本概念和问题 20.1.1 CAS原子操作 20.1.2 ABA问题 20.1.3
ABA问题的解决方法 20.1.4 内存删除问题 20.1.5 数据竞争问题 20.2 Lock-Free的队列 20.2.1
无锁队列的链式实现方法 20.2.2 串行实现方法 20.2.3 出队操作的Lock-Free实现 20.2.4 进队操
作的Lock-Free实现 20.2.5 CLockFreeQueue的实现代码 20.3 Lock-Free程序的问题分析 20.4 参考
文献 附录1 本书代码和CAPI开源项目源文件对照表附录2 多核编程的四层境界

<<多核计算与程序设计>>

章节摘录

插图：第1部分 基础知识1 多核计算概述 1.1 多核CPU概述 1.1.1 多核计算将成为发展趋势 1.单核CPU的发展限制在过去的几十年里，个人PC的CPU速度的发展一直按照摩尔定律进行——半导体厂商能够集成在芯片中的晶体管数量每18 ~ 24个月翻一番，反映到实际使用中就是处理器的时钟频率每18 ~ 24个月增加一倍。

因此，长期以来提高处理器的主频成为提高CPU速度的不二法则。

目前，单核CPU的速度已超过3 GHz。

提高主频会增加处理器的功耗和设计的复杂度，其中最大的困难就是提高主频所带来的高发热问题；如果继续提高主频，高发热问题将成为不可克服的障碍，它会导致芯片运行不稳定，因此目前主频的提升空间已经不大。

在单核时代，提高性能的另一手段是用superscalar（超标量）处理器的方式，让处理器在一个时钟周期内执行多条指令。

超标量处理器通常有两个或多个处理单元，利用这些硬件资源，需要对软件进行精心设计；要适应多流水线，需要对软件进行大量的修改，这些都会影响软件的可移植性。

<<多核计算与程序设计>>

编辑推荐

《多核计算与程序设计》特色内容：并行遍历的基本方法，常见并行算法如并行搜索、并行排序、并行数值计算等在多核系统中的实现。

共享资源分布式计算的基本编程模式和方法，分布式队列，它能自动给每个线程赋予一个本地队列，它是基于偷取的共享队列和队列池来实现的。

分布式查找，包括分段锁的哈希表，动态负载平衡的多级查找等。

分布式内存管理，它自动给每个线程生成一个本地的内存管理器，并且几乎不需要使用锁进行内存分配和释放（抢夺式内存管理）。

任务图分解与调度及实现方法，非嵌套任务调度，可用于网络服务器软件等地方进行任务调度。

嵌套任务调度，是另一种更广泛的任务调度方法，可以用它实现各种并行计算。

各种程序和算法中的伪共享问题的处理，Lock-Free编程基础知识。

<<多核计算与程序设计>>

版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:<http://www.tushu007.com>