Computer Science and Application 计算机科学与应用, 2014, 4, 175-180
Published Online September 2014 in Hans. http://www.hanspub.org/journal/csa
http://dx.doi.org/10.12677/csa.2014.49025

Hans汉斯

The Application Research of OpenMP on Multicore Processors

Xiaoping Wang, Xiucheng Dong, Weicheng Xie

School of Electrical and Information Engineering, Xihua University, Chengdu
Email: sichuanwww@163.com

Received: Jul. 6th, 2014; revised: Aug. 6th, 2014; accepted: Aug. 15th, 2014

Abstract

Multicore operating system is superior in assigning and scheduling the general processes, but it is unsatisfying in processing parallel data of the application layer. We study the characteristics of OpenMP and propose a mechanism for the development of application-layer multi-core parallel program. We conduct a comprehensive study and compare its performance. Our experimental results show that the mechanism is convenient and flexible to control the multicore parallel granularity of data, take advantage of multicore computing resources, and enhance the real-time of applications. To the ten million of computation, there are greater improvements in running time: the ratio is 0.5:1 between the dual-core and the non-parallel systems; the ratio is 0.38:1 between the quad-core and the non-parallel systems; the ratio is 0.19:1 between the eight-core and the non-parallel systems; the ratio is 0.14:1 between the sixteen-core and the non-parallel systems.

Keywords

Multicore Programming, Parallel Computing, OpenMP

OpenMP在多核处理器上的应用研究

王孝平，董秀成，谢维成

西华大学电气信息学院，成都
Email: sichuanwww@163.com

收稿日期：2014年7月6日；修回日期：2014年8月6日；录用日期：2014年8月15日

摘　要

多核操作系统在解决通用进程的分配和调度等宏观问题上表现优越，而对于应用层的多核数据并行计算没有做到精确控制。本文深入研究了OpenMP的特性，提出了一种应用层开发多核并行程序的机制，对其性能进行了全方位的研究比较。实验结果表明，利用这种机制，可以方便灵活的控制多核数据并行粒度，充分利用多核计算机资源，提升应用程序的实时性。对于千万级别的运算量，运行时间上有较大的改进：双核并行为非并行的0.5，四核并行为非并行的0.38，八核并行为非并行的0.19，十六核并行为非并行的0.14。

关键词

多核编程，并行计算，OpenMP

1. 引言

随着多核处理器的日益普及，对于应用软件来说，有效利用底层的多核就成为了一项迫切要求。多核处理器提供了大幅提升计算机性能的机制，多核软件就是可以真正利用这一特点的策略。多核操作系统主要解决的是通用进程的分配和调度，以便解决负载均衡、实时性等宏观问题[1]。这就决定了多核操作系统不可能针对具体用户的具体数据处理进行很好的并行设计。所以，在多核处理器上的用户数据并行性设计，应该由用户多核程序设计来完成。

本文致力于数据并行粒度划分的策略研究，并通过具体实验对各种策略的效果进行深入比较分析。

2. 多核操作系统对多核的使用

多核操作系统采用核心级线程和用户级线程的两种线程模型。多核操作系统的调度在核心态完成，这样调度器可以专注于核心态线程在处理器上的调度。用户态的线程运行在逻辑机器上，底层的多核 CPU 资源只能由操作系统统一分配调度，如图 1 所示。多核操作系统主要解决的是通用进程的分配和调度，以及如何把进程内的线程整体分配到合理的物理核上，以便解决负载均衡、实时性等宏观问题。这就决定了多核操作系统不可能针对具体用户的具体数据处理进行很好的并行设计，也就不可能充分的利用好多核的资源[2] [3]。

用户应用的多样性和复杂性决定了多核并行设计的多样性和复杂性。用户应用的结构将决定是采用功能分解还是数据分解(或两者并用)来实现多核并行。如果用户应用具备互相独立的功能或任务，那么它们可能并行运行。如果用户应用具备运行于大量数据之上的功能且可能将数据分解成若干能够独立处理的更小单元，那么可能采用数据分解。用户应用的特性还将决定最佳的并行粒度。并行粒度指的是组成应用的任务需要多久互相通信一次。所需的通信频度越低，能够使用的并行粒度越粗，由于所需的通信开销更少，应用也就能够从并行性中获益更多，如图 2 所示。

在多核并行设计之前识别出最大问题出现在哪里非常重要。热点是处理器耗费大量时间的地方，因此可能是针对代码低效而进行优化的最佳目标。然而，也可能热点已经很高效，处理器之所以在此耗费大量时间是因为大量任务正在执行。当处理器需要花费额外的时间是由于低效时，热点就成为瓶颈。如果确定瓶颈能够并行处理，那么它将是进行并行设计的理想位置[4] [5]。

现在的多核操作系统，面对用户如此多样性的特定应用，无法充分发挥多核处理器的功能。

3. OpenMP 对多核的使用

最常使用的三种多核并行编程技术分别为多线程、共享存储模式和消息传递。多线程支持在单个进程中存在多个共享内存和其它资源的线程，从而使多核系统运行更快。共享存储模式支持多个处理器共用一个内存空间，从而提供一个统一地址空间，该空间使用简单。消息传递涉及到进程之间的通信，实施起来可能更加费力，但是它能够通过同步避免竞态条件。OpenMP(Open Multi-Processing)针对共享内存的多核模型，提供了用户级的共享存储并行编程标准。如图 3 所示，它采用 Fork/Join 的并行执行模型，将程序划分为并行区和串行区，不同的处理器之间通过共享变量完成数据交换。

多线程应用的并行任务粒度会对其并行性能产生很大影响。为了实现较好的负载平衡而取得最优的性能,就必须对循环进行调度与分块。OpenMP 提供了 4 种不同调度策略:静态调度(Static)、动态调度(Dynamic)、指导调度(Guided)和运行时调度。其中，运行时调度是指在程序运行时，通过环境变量指定其余 3 种调度策略中的一种，因此本文仅讨论前 3 种调度策略。

静态调度是指将循环迭代划分成相等大小的块。当循环迭代次数不能整除线程数时，尽可能地划分成相等大小的块。在默认情况下，没有指定迭代块的大小，静态调度会尽可能为每个线程分配数量相同的迭代块。如果指定输入参数 chunkSize 的值，静态调度将会为每个线程依次分配由 chunkSize 指定的迭

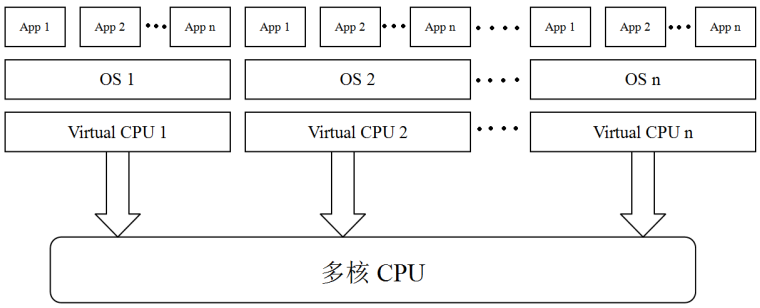

Figure 1. The technology of multicore and multithreading in operating system
图 1. 操作系统的多核多线程技术

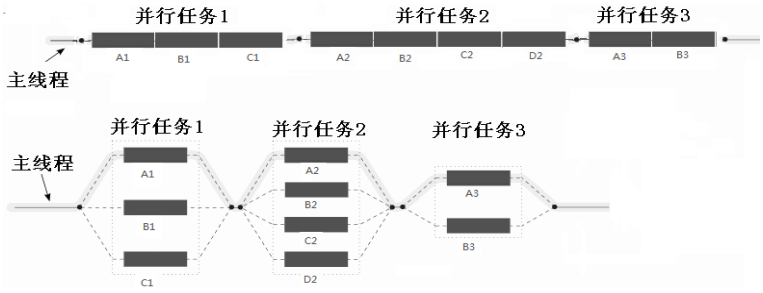

Figure 2. The granularity division of serial and parallel tasks
图 2. 串行与并行粒度划分

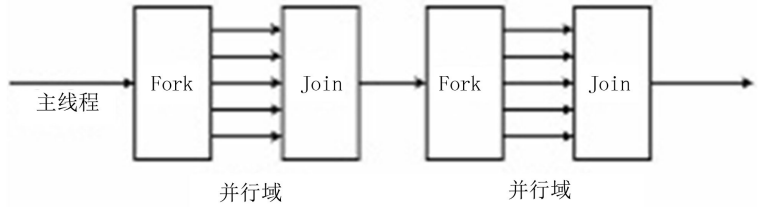

Figure 3. The parallel execution model of OpenMP
图 3. OpenMP 的并行执行模型

代次数，直到所有迭代块都分配完毕。

动态调度一般采用一个内部的任务队列，当某个线程可用时，为其分配由参数 chunkSize 指定的迭代次数。当线程完成当前分配的迭代块以后，将从任务队列头部取出下一组迭代。在默认情况下，块大小为 1，此时每个迭代逐次地分配到各个线程，调度开销也会变得很大。

指导调度是一种采用指导性的启发式自调度方法。开始时每个线程会分配到较大的迭代块，之后分配到的迭代块会逐渐递减。迭代块的大小会按指数级下降到指定的 chunkSize 大小。因此，指导调度又被称为指数调度。可选的参数 chunkSize 可以指定所使用的迭代块的最小值，默认为 1。每次调度的迭代块数是由 NCk = LCk/(p + NP)计算的，其中 LCk 表示剩余的未调度的循环迭代次数，NP 是线程个数，NCk 表示第 k 个迭代块的大小，p 是一个比例因子，推荐值为 1 或者 2。

多线程应用的并行任务粒度会对其并行性能产生很大影响。在分解一项应用使之适用于多线程处理时，开发人员通常采用的方法是从逻辑上将问题分割成尽量多的并行任务，或者在并行任务内根据共享数据与执行顺序决定进行哪些必要的通信。由于分割任务、将任务分配给线程以及在任务之间进行数据通信涉及到一定的成本，开发人员通常需要聚合或整合分割的任务，用于避免随之产生的开销，尽量实现应用高效运行。通过聚合分割的任务可确定并行任务的最佳粒度。

粒度通常与工作负载在线程之间的均衡程度有关。尽管平衡大量小型任务的工作负载更容易，但这样做却可能导致通信和同步等方面的并行开销过高。此时，开发人员可以通过将小型任务整合成一项任务，增加每项任务的粒度来减少并行开销[6]。

4. 方案设计与实验

我们通过求解两千万(20,000,000)以内质数的个数为实验标的，进行两种实验设计。一种是在八核计算机上，采用不同的数据并行粒度划分，寻求到一种相对最好的数据粒度划分方法。另一种就是分别在双核、四核、八核和十六核目标机上，测试我们的 OpenMP 并行实验程序，充分探索多核并行程序的规律。

质数的判断，我们采用最通用的快速判定方法：对于数字 n，如果在[2, sqrt(n)]没有能整除 n 的数存在，则 n 为质数。

4.1. 八核并行程序实验

对于八核处理器的并行程序调度，我们依据 OpenMP 的特点，依次设计了非并行调度(OS)、OMP 默认调度(OMP)、静态调度(Static)、动态调度(Dynamic)和指导调度(Guided)，如表 1 所示。第一列为非并行调度(OS)，第二列为 OMP 默认调度(OMP)，第三列为静态调度(Static)，第四列为动态调度(Dynamic)，第五列为指导调度(Guided)。前五行为分别进行五次实验得到的运行时间，第六行为前五次实验得到的运行时间平均值，单位均为毫秒。

从表 1 可以得出，对于我们的实验标的，非并行调度运行时间均值达到 14,898 毫秒，而对于并行设计中最快的指导调度运行时间均值为 2854 毫秒，非并行设计运行时间均值为指导调度运行时间均值的 5.22 倍，运行速度提升非常明显。

对于并行多核设计的四种调度方式，指导调度(Guided)效果最好，根本原因在于这四种调度方式对于数据并行粒度的划分方式的不同。如表 2 所示，对于两千万(20,000,000)数据，不同的调度策略对数据并行粒度的划分有很大的不同。第一列为 OMP 默认调度(OMP)，第二列为静态调度(Static)，这两种方式实际是采用的一种简单的平均粒度划分方式；第三列为动态调度(Dynamic)，采用共享分配方式，某个处理器处理完当前的数据后申请下一块数据处理；第四列为指导调度(Guided)，采用优化的调度策略，让每一

个处理器充分发挥发挥各自的处理能力，各尽所能。

4.2. 不同多核程序并行设计

为了进一步研究同一程序在不同多核机器上运行的特点，我们分别在双核、四核、八核和十六核机器上进行了实验，程序运行时间如图 4 所示。

为了保证实验的准确性，每一个实验我们都进行了多次实验，然后采用其平均值。从图 4 中，我们可以得到几个显著的特征：

Table 1. Multicore parallel running time
表 1. 多核并行运行时间

OS	OMP	Static	Dynamic	Guided
14,890	3590	2990	2900	2850
14,890	3440	2880	2870	2870
14,910	3460	2870	2860	2860
14,910	3470	2880	2860	2850
14,890	3440	2990	2850	2840
14,898	3480	2922	2868	2854

Table 2. Multicore parallel granularity division
表 2. 多核并行粒度划分

OMP	Static	Dynamic	Guided
2500000	2500000	2763288	2924818
2500000	2500000	2594680	3639266
2500000	2500000	2397616	2407550
2500000	2500000	2531784	2532023
2500000	2500000	2509888	2092543
2500000	2500000	2388136	2071571
2500000	2500000	2296928	2332491
2500000	2500000	2517680	1999738

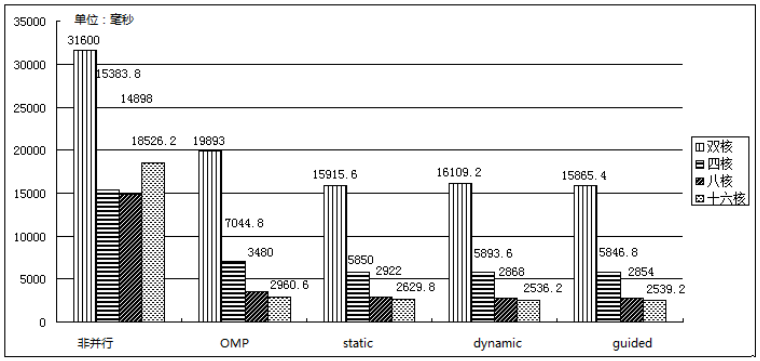

Figure 4. The chart of different multicore program running time
图 4. 不同多核程序运行时间图

1) 对于多核并行设计的三种方式，指导(Guided)方式效果最好，动态(Dynamic)方式次之，静态(Static)方式最差。这与他们采用的并行分配机制密切相关：指导方式采用启发引导分配方式，减少分配次数，充分发挥多核的作用；动态方式只有一个分配队列，在分配时间上耗费较多；静态方式采用平均分配，没有区别不同 CPU 负载不一样的现状。

2) 多核程序要求多核必须具有一定的数量，才能体现出较好的效果。双核的并行时间提升与其他多核比起来效果最差；双核的四种并行方式在运行时间上效果差异不明显。十六核的并行运行时间提升最为显著，从非并行的 18526.2 毫秒到并行的 2539.2 毫秒，最好的并行时间仅为非并行时间的 0.14。但是，实验结果也表明，多核的数量超过了一定的数量后，再增加更多的多核，运行时间并没有更加显著的提升。如图 4 所示，在运行时间上，双核到四核、四核到八核，提升非常明显；但八核到十六核，提升并不明显。考虑到成本因素，我们实验的运行最佳环境应该是八核。

5. 未来工作

我们采用 OpenMP 对大数据进行了数据粒度上的并行运算处理，深入研究了 OpenMP 的四种并行调度方式，并通过实验得到了这四种调度方式的特点。我们可以充分利用普通计算机上的多核进行计算，大大的提升运算速度。

对于复杂的大数据，其内部逻辑关系复杂，不可能采用单一的并行数据粒度划分，只有依据其内部逻辑关系，进行分层次的并行数据粒度划分，才能充分发挥多核并行处理能力。未来的工作就是在层次并行数据粒度划分上深入研究，进一步优化提升多核并行数据处理的性能。

基金项目

感谢西华大学电气信息学院董秀成教授工作室的大力支持，感谢西华大学四川省信号与信息处理重点实验室的大力支持。同时感谢四川省教育厅重大培育项目(编号：13ZC0003)和教育部“春晖计划”科研项目(编号：Z2012028)的大力支持。感谢各位评审提供的宝贵修改意见，使文章增色不少，得以发表。

参考文献 (References)

[1] 于慧莉, 李勤新, 宫春明 (2014) 一种基于 OpenMP 和 MPI 的非序贯蒙塔卡罗暂态稳定评估的动态混合并行化方法. *现代电力*, **2**, 31-36.

[2] 潘亮, 郭改枝, 宋鑫梦 (2014) 基于 OpenMP 矩阵相乘并行算法的设计. *宝鸡文理学院学报*(*自然科学版*), **1**, 21-23.

[3] 熊雨前, 冯丙春, 薛军超 (2014) OpenMP 在 PDF 技术中的应用. *数字与缩微影像*, **1**, 43-45.

[4] 万波, 魏帆, 金钟 (2014) 自洽场方法在 Xeon Phi 上的并行实现. *科研信息化技术与应用*, **2**, 34-42.

[5] 白洪涛, 李昂, 欧阳丹彤, 邢书豪, 刘雪飞 (2014) 基于多核的粗粒度 2.5 维电磁场正演并行算法. *吉林大学学报*(*理学版*), **3**, 509-514.

[6] 蒋沁谷 (2014) GRAPES 全球模式 MPI + OpenMP 混合并行方法. 中国气象科学研究院, 北京.