**并行计算技术及其应用简介**

XX

（XXX， XX，XXX）

**摘 要：** 并行计算是实现高性能计算的主要技术手段。在本文中从并行计算的发展历程开始介绍，总结了并行计算在发展过程中所面临的问题以及其发展历程中出现的重要技术。通过分析在当前比较常用的实现并行计算的框架和技术，来对并行计算的现状进行阐述。常用的并行架构分为SMP（多处理系统）、NUMA（非统一内存存储）、MPP（巨型并行处理）以及集群。涉及并行计算的编程模型有MPI、PVM、OpenMP、TBB及Cilk++等。并结合当前研究比较多的云计算和大数据来探讨并行计算的应用。最后通过MPI编程模型，进行了并行编程的简单实验。

**关键词：** 并行计算；框架；编写模型；应用；实验

# A Succinct Survey about Parallel Computing Technology and It’s Application

**Abstract:** Parallel computing is the main technology to implement high performance computing. This paper starts from the history of the development of Parallel Computing. It summarizes the problems faced in the development of parallel computing and the important technologies in the course of its development. Through the analysis of framework and technology commonly used in parallel computing currently,to explain the current situation of parallel computing.Framework commonly used in parallel are SMP(multi processing system),NUMA(non uniform memory storage),MPP(massively parallel processing) and cluster.The programming models of parallel computing are MPI, PVM, OpenMP, TBB and Cilk++, etc.Explored the application of parallel computing combined with cloud computing and big data which are very popular in current research.Finally ,through the MPI programming model,a simple experiment of parallel programming is carried out.

**Key words:** parallel computing; framework; programming model; application; experiment

**1引言**

近年来多核处理器的快速发展，使得当前软件技术面临巨大的挑战。单纯的提高单机性能，已经不能满足软件发展的需求，特别是在处理一些大的计算问题上，单机性能越发显得不足。在最近AlphaGo与李世石的围棋大战中，AlphaGo就使用了分布式并行计算技术，才能获得强大的搜索计算能力。并行计算正是在这种背景下，应运而生。并行计算或称平行计算时相对于串行计算来说的。它是一种一次可执行多个指令的算法，目的是提高计算速度，及通过扩大问题求解规模，解决大型而复杂的计算问题。可分为时间上的并行和空间上的并行。时间上的并行就是指流水线技术，而空间上的并行则是指用多个处理器并发的执行计算。其中空间上的并行，也是本文主要的关注点。

并行计算（Parallel Computing）是指同时使用多种计算资源解决计算问题的过程，是提高计算机系统计算速度和处理能力的一种有效手段。它的基本思想是用多个处理器来协同求解同一问题，即将被求解的问题分解成若干个部分，各部分均由一个独立的处理机来并行计算。并行计算系统既可以是专门设计的，含有多个处理器的超级计算机，也可以是以某种方式互联的若干台的独立计算机构成的集群。通过并行计算集群完成数据的处理，再将处理的结果返回给用户。

目前常用的并行计算技术中，有调用系统函数启动多线程以及利用多种并行编程语言开发并行程序，常用的并行模型有MPI、PVM、OpenMP、TBB、Cilk++等。利用这些并行技术可以充分利用多核资源适应目前快速发展的社会需求。并行技术不仅要提高并行效率，也要在一定程度上减轻软件开发人员负担，如近年来的TBB、Cilk++并行模型就在一定程度上减少了开发难度，提高了开发效率，使得并行软件开发人员把更多精力专注于如何提高算法本身效率，而非把时间和精力放在如何去并行一个算法。

文献[1]回顾了并行计算技术六十多年的发展历史，并分析了其发展的未来趋势和所面临的可拓展性“墙”。文献[2]对大规模MPI计算中，当前常用的容错机制和多模冗余带来的开销进行了分析，提出了一种可拓展三模冗余容错机制。文献[3]则阐述了并行计算和云计算之间的关系，并介绍了并行计算在云计算中运用的一些指导原则。文献[4]分析了在大数据时代，并行计算面临的挑战和基于，综述了面向批处理、面向流处理、面向图数据以及面向内存等几类并行计算模型的相关研究。文献[5]对并行计算模型进行了概述，介绍了理想计算模型的特征，讨论了一些典型的计算模型，综述了目前并行计算模型在异构计算环境的发展情况。文献[6]针对传统单一的并行计算模型变得越来越复杂，对并行计算个阶段针对性不强，提出了一种分层并行计算模型的思想。文献[7]针对在专业集群中进行并行计算的处理，存在费用高、管理困难和操作复杂等问题，利用PC和以太网相连的实验环境，采用集群工作框架和并行计算的关键技术，提出基于Linux集群的并行计算系统环境，并在该环境上对系统性能进行了测试。文献[8]在CPU/GPU协同并行计算这一新兴领域，对当前的研究工作进行了综述。根据所用计算资源的规模将CPU/GPU协同并行计算研究划分为三类，尔后从立项依据、研究内容和研究方法等方面重点介绍了几个混合计算项目，并指出了可进一步研究的方向，以期为领域科学家进行协同并行计算研究提供一定参考。文献[9]中对图形处理器中的相关问题进行研究。提出了统一计算架构（CUDA）为实现GPU通用计算提供了高效、简便的方法．因此，建立了基于GPU通用计算平台的中心差分格式的显式有限元并行计算方法．该方法针对GPU计算的特点，对串行算法的流程进行了优化和调整，通过采用线程与单元或节点的一一映射策略，实现了迭代过程的完全并行化。

**2并行计算发展的三个重要阶段**

并行计算是实现超高计算性能的主要技术手段，回顾其发展的历史进程，可以看出并行计算的发展从整体上跨越了三个阶段。

**2．1 Amdahl公式与适度并行**

早在二十世纪四十年代中后期，现代电子计算机的鼻祖冯·诺依曼就提出了用并行计算提高计算机性能的技术途径。所谓并行计算，就是采用多个运算部件或多台计算机共同开展计算任务。理论上，并行多少台计算机就可以获得多少倍的运算速度，但实际加速能力往往和应用特征密切相关。1967年，Amdahl提出了著名的计算性能加速比公式（也称Amdahl定律）：

（1）

公式（1）中的表示应用的串行版本程序的执行时间，f表示应用中不能被并行化部分的工作量占整个程序工作量的比率，n表示并行系统的处理器数目。表示应用的并行版本程序和n个处理器上的执行时间。这些概念和理论方面的基础研究极大地推动了适度并行计算技术的实用化。

**2．2 Gustafson公式与大规模并行**

80年代后期学术界展开了一次很大的讨论，大家对并行是否有前途众说纷坛，问题的焦点是-------能不能通过并行规模增大来持续提高计算机性能。

1988年，美国科学家Gustafson在大量实际物理问题并行计算过程中总结了一个著名公式，发表在《Communication of ACM》上。这篇论文只有两页，但它揭示了一个规律，即Gustafson通过他的应用实践发现，很多程序随着应用规模的扩大，程序中不能被并行化部分工作量的绝对量基本保持不变。而不能被并行化部分工作量占总工作量的相对比例f会相应减小。基于这个规律，他提出了著名的Gustafson加速比公式：

（2）

从公式可以得出，当f随着计算规模扩大而不断缩小时，处理器数如果增多，一个程序的加速比还是可以随着处理器的增多而提升的。这个公式不是一个精细的性能评估公式，而是一个思想性的公式，它揭示了大规模并行的要旨：并行计算机的实际性能和它所运行的并行程序特征密切相关。在大规模并行计算机上要取得高实用计算性能，在其上运行的并行程序必须具有串行计算负载保持不变，并行计算负载随着计算机规模的扩大而扩大的特点，这样的程序通常成为可扩展并行程序。在提高大规模并行计算使用效率的过程中，能否设计出可扩展并行算法和可扩展并行程序，与大规模并行计算机系统的研制同等重要。

**2．3 效能评价模型与高效能并行**

高性能计算机应该向什么方向发展，2002年美国国防部（DARPA）提出的“高效能计算系统（HPCS）”计划，着眼于高性能向高效能的转变，认为高效能计算机系统不仅要关注很高的计算性能，还要关注系统的实用性能、可编程性、可移植性、系统鲁棒性和经济可行性。于是基于经济学中产生率的概念和效用理论逐渐取代了以往单一的计算性能评价模型。效能模型框架如下：

（3）

其中，T为问题P解决的时间（Time-to-solution），M是并行计算机系统，U为效用函数，C为在系统M上T时间内解决问题P所投入的最小成本函数。该效能度量模型综合考虑了并行系统开发（Development）和运行（Execution）两种状态，将开发的时间和系统运行的时间统一到问题解决时间T中。虽然，围绕这一公式标志着高性能计算机学术界、产业界对大规模并行计算机发展趋势产生了新的认识。

**3 并行编程模型**

并行编程可以调用系统函数启动多线程，也可以利用并行编程模型，如常用的并行模型有MPI、PVM、OpenMP、TBB及Cilk++等。

**3.1 MPI简介**

MPI（Message Passing Interface）是基于消息传递的并行编程模型。消息传递指的是并行执行的各个进程具有自己独立的堆栈和代码段，作为互补相关的多个程序独立执行，进程之间的通信通过显式地调用通信函数来完成。

（1）MPI程序基本函数

①MPI\_Init用来初始化MPI执行环境，建立多个MPI进程之间的联系，为后续通信做准备。MPI\_Finalize则是结束MPI执行环境。这两个函数之间定义MPI程序的并行区域。

②MPI\_Comm\_rank 来表示各个MPI进程。

③MPI\_Comn\_size 用来表示进程组中有多少个进程，返回整型的错误值，同时有两个函数参数，MPI\_Comn 类型的通信域，标识参与计算的MPI进程组，如MPI\_COMM\_WORLD;整数指标，返回相应进程组中的进程数。

1. MPI性能分析与优化举例

MPI消息传递开销由两部分组成：一部分是通信固有的延迟；另一部分是消息传递开销，与消息的数据量和网络带宽有关。时间消耗可以用下面的式子表示：

（4）

其中，、、分别表示程序总用时、通信用时和串行部分用时。而通信用时一般可作如下分解：

（5）

其中，n为点对点通信次数，m为集群通信次数，为通信固定延迟，为第i次通信的容量，w为网络带环，为负载不平衡造成的时间消耗，集群通信耗时。

1. 解决负载均衡问题

在并行计算中，若各个处理器上的执行时间不同，因同步会使先完成的处理器等待未完成的处理器，此时应该考虑负载均衡问题，是的资源得到合理利用。负载均衡分为：静态负载均衡及动态负载均衡。静态负载均衡使用与计算前可以准确估算总的负载，且这些负载容易平均划分给各个进程的情况。对于实现不知道负载总数，或者总负载不易平均划分的情况，需要动态负载均衡策略来解决。

**3.2 PVM简介**

PVM是一个在网络上的虚拟并行机系统的软件包，它允许将网络上基于UNIX操作系统的并行机和单处理机的集合当成一台单一的并行虚拟机来使用。PVM支持用户采用消息传递方式编写并行程序。计算以任务为单位，一个任务就是一个UNIX进程，每个任务都有一个taskid来标识（不同于进程号）。

PVM支持在虚拟机中自动加载任务运行，任务间可以相互通讯以及同步，在PVM系统中，一个任务呗加载到哪个节点上去运行，PVM就允许用户指定任务被加载的节点，这样就方便了用户编写并行程序。

**3.3 OpenMP简介**

OpenMp是一种面向共享内存及分布式共享内存的多处理器多线程并行编程语言，它也是一种能够用于显示指导多线程、共享内存并行的应用程序编程接口。OpenMP 的编程模型以线程为基础，通过编译指导语句来显示地指导并行化，为编程人员提供了并行化的完整控制。OpenMP 的执行模型采用Fork － Join 形式，Fork － Join 执行模式在开始执行的时候，只有一个主线程存在。主线程在运行过程，当遇到需要进行并行计算的时候，派生出子线程来执行并行任务。在并行执行的时候，主线程和派生线程共同工作。在并行代码结束执行后，派生线程退出或者挂起，不再工作，控制流程回到单独的主线程中。

**3.4 Intel TBB（Threading Building Block）简介**

Intel TBB 是一个用来支撑可扩展编程的库，TBB 的工作层次要高于裸线程，它为C + + 程序提供了丰富而完备的方法来表达并行语义。TBB 指定的是任务而不是线程，一般线程软件包都要求创建、汇合及管理线程。TBB 运用自己的调度算法把任务分配到各个线程，高效合理地利用处理器资源。TBB 的目标是通过线程来提升性能，将密集的计算任务并行化，提供更高层次和更简单的解决方案，TBB 依赖泛型编程。TBB 中还支持算法模板，如基于递归区间的面向循环算法，根据需求来递归地划分问题，从而获得并行任务的正确层次。相比静态划分问题，递归划分问题的方式更占优势。线程构建模块的函数都定义在空间tbb 中，TBB 提供parallel\_for 与parallel\_reduce 泛型并行算法，可以负载均衡地对固定数目的独立循环迭代体进行并行执行。

**3.5 Cilk++简介**

Cilk ++ 语言扩展了C ++ 语言来编写并行应用程序，有效地利用多处理器。Cilk + + 语言特别适合但不限于分而治之算法，把任务分解成子问题( 任务) ，可以独立地加以解决。递归函数通常用于分而治之算法。这些任务会在不同循环迭代。Cilk ++ 关键字识别函数调用和循环，可以并行运行。Cilk ++ 语言比较适合“二分法”算法，将问题分解成若干独立子问题，再将这些执行结果合并起来。任务既可以在不同的函数里实现，也可以在一个迭代的循环中完成。Cilk ++ 的关键词能有效地标识可并行执行的函数调用和循环，Cilk ++ 的运行环境能有效地将这些任务调度到空闲的处理器上运行。

**4几种应用背景下的并行计算**

**4.1 云计算与并行计算**

云计算是分布式处理、并行处理和网格计算的进一步发展，是一种新兴的共享基础架构的技术，它可以将巨大的系统池连接在一起以提供各种IT 服务。连接设备、实时数据流、SOA 的采用以及搜索、开放协作、社会网络和移动商务等都是推动云计算技术发展的推动因素。

云计算与并行计算相比，在应用广泛性方面，并行计算是为满足某科学领域的专业需求，要求开发者具备较强的技术水平，不具备较强的广泛性，但云计算可以运用在较多领域，而且具有好的效果。在性能方面，并行计算主要追求较高的加速比，而云计算则单机的计算能力要求较低，此系统通过将分散的结点统一，根据相应调度算法合理调度，从而获得巨大的计算能力，而且还能根据客户的需求变化进行调整。

并行计算和虚拟化一起构成了云计算技术的骨干。对于大型应用系统实施云计算的主要内容就是实施并行计算。并行计算技术除了目前炙手可热的MapReduce 以外，值得推荐的还有OpenMP、MPI 和Erlang。这3 个技术主要应用于并行程序的开发， 和其他并行策略（如SOA）一起，用于从根本上实现应用的并行化。中国电信在支撑云的研究、建设中，计划在多个层面实施并行计算，以提高应用运行效率和可靠性，降低采购和运营成本。对于条件成熟的应用，拟从业务并行开始，在开发层面即实现并行计算。目前，经过研究分析和测试，针对不同的应用，初步制定出不同的并行化解决方案，采用合适的并行计算技术实现不同层次的并行计算。

**4.2 大图数据背景下的并行计算模型及其优化技术**

大数据时代背景下，数十亿顶点级别大规模图的不断涌现以及云计算基础设施的持续完善，推动着图数据处理的研究重心由单机图算法的高度优化逐渐转向分布式并行大图处理的优化。目前，大图数据处理存在两种典型的模式：一是采用通用的海量数据分布式并行计算框架MapReduce进行处理；二是采用完全面向图结构设计的专用大图计算框架。由于Mapreduce在应用定位与设计目标上主要是针对模式自由（schema-free）的数据对象实现高吞吐的批量处理，因此其缺乏有效处理大图的内部机制。相比之下，专用大图并行计算模型有针对性地考虑了图计算的基本特征，因此从框架内部就已经提供了对大图处理的支持，能获得较好的性能。其显著的特征是提供了对图计算中最常见的迭代操作的支持和实现了直观易用的以顶点为中心的编程抽象。

从存储架构上可以将大图并行计算模型分为：面向分布内存架构以及面向单机多核共享内存架构两类。在分布内存架构下，目前具有代表性的大图并行计算模型有Pregel、HAMA、Giraph、Distributed GraphLab以及Trinity等。由于图结构的高耦合性，分布式环境下图计算的网络通信代价很高，图划分是优化分布式大图计算通信开销的有效手段。部分大图处理系统采用经典的划分方法，如ParMetis，也有其他一些系统探索了新的划分方法，如GraphX和PowerGraph均采用vertex-cut的划分方法缓解自然图中高度数（high-degree）顶点通信集中的问题；Trinity采用多层标签传递的划分方法，借助图的语义有效支持了十亿顶点级别大图的划分；而GPS和Mizan都利用动态的划分方法缓解了大图计算过程中负载失衡时的顶点重分配的问题。除了最棘手的图划分问题外，分布式内存架构仍然面临很多困难，如需要开发者具有专门的分布式编程知识在集群环境中进行复杂的调试与优化工作。鉴于此，一些研究工作面向单机环境提出了有针对性的优化技术，如序列化随机访问、利用多核以及新型存储的高并发能力、引入异步机制等，并设计了相应的大图计算模型。上述研究利用外存的一些访问特性，通过有针对性的优化提升处理效率，缓解大图对内存的压力。此外，还有一些基于单机的图计算系统，仍然假设以全内存的方式进行图数据处理。虽然这些研究的侧重点不在存储模式，但是在大图环境下这些研究成果的实施效果可能会受到一定的影响。

**4.3 集群背景下的并行计算**

集群(Cluster)是一组相互独立的服务器在网络中表现为单一的系统，并以单一系统的模式加以管理，以此单一系统为客户提供高可靠性的服务。

在大多数模式下，集群中所有的计算机拥有一个共同的名称，集群内任一系统上运行的服务可被所有的网络客户所使用。Cluster必须可以协调管理各分离的组件的错误和失败，并可透明地向Cluster 中加入组件。一个Cluster 包含多台(至少2 台)拥有共享数据存储空间的服务器。任何一台服务器运行一个应用时，应用数据被存储在共享的数据空间内。每台服务器的操作系统和应用程序文件存储在其各自的本地储存空间上。Cluster 内各节点服务器通过一个内部局域网相互通信。当一台节点服务器发生故障时，这台服务器上所运行的应用程序将在另一台节点服务器上被自动接管，Cluster 结构如图1 所示。

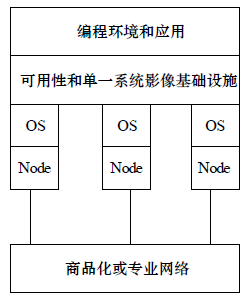


图1 Cluster体系结构

采用 Cluster 体系结构的系统，具有可自由伸缩、高度可管理、高可用、高性能价格比等诸多优点，解决了跨平台管理、跨操作系统管理、系统软硬件运行状态监控等混合平台应用等技术难题，是担当大规模科学工程计算(如并行计算)的坚实且理想的系统平台。

集群可以分为 3 类：高可用性集群，负载均衡集群，超级计算集群。高可用性集群的主要功能是提供24 h 不间断服务；通常是由多个节点构成，当单个节点出现故障时，系统其他节点接替服务，整个系统仍能继续正常对外提供服务。负载均衡集群将系统的整体负载合理地分配到各个节点上，使得每个节点都不会因为超负荷工作而崩溃；并且系统整体负载需求大于系统整体负载能力时，可以通过增加系统节点，平滑地拓展系统负荷能力。超级计算集群是通过将多台机器连接起来同时处理复杂的计算问题的计算机群。

**5关于并行计算的两个简单实验**

**5.1基于MPI的hello word程序测试**

本实验旨在测试在单机环境下的基于MPI并行程序的实验环境。采用的是编程环境是：VC6.0和MPICH2。程序采用C++语言编写。程序源代码，见附录源代码一。

运行过程：在命令行窗口中输入：mpiexec -n 200 mpitest.exe 。 mpiexec是MPI运行库的命令,-n 200表示启动200个mpitest.exe ；运行截图如下。通过观察任务管理器，发现有很多个mpitest.exe的进程。因为没有联网的其他节点存在，所以200个都在本机上运行了。

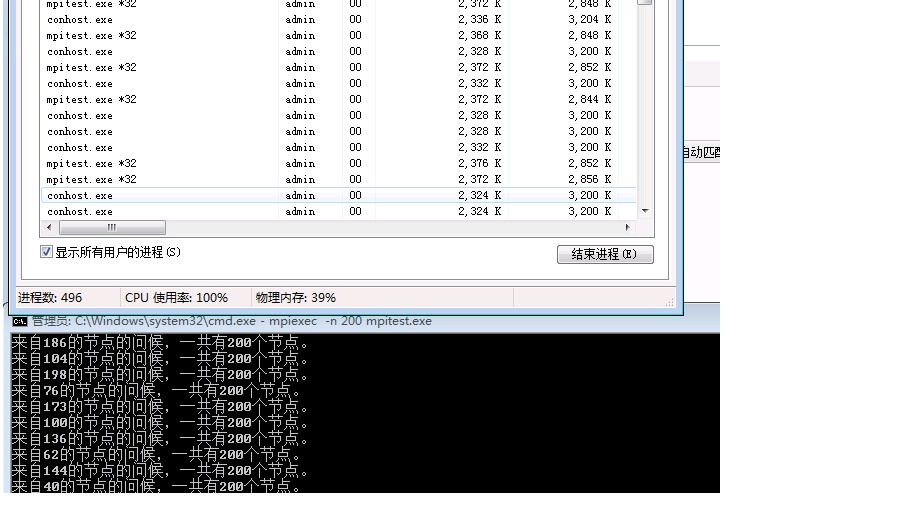


图2 hello world程序运行结果图



图3 hello world程序结果局部放大图

键入命令“mpiexec -n 200 mpitest.exe ”后 ， MPI运行库接收到要并行运行200个mpitest.exe，查找临近节点，调度临近节点上的mpitest.exe，（在这里的测试上因为只有一个节点，所以全部一起执行了）。

**5.2 数字字符串处理问题**

在这一节进行的是一个通过MPI和C语言编写并行程序的实验，首先对问题进行描述如下：对于任意一个给定的合法数字字符串，将其各位累加求和，然后对求和得到的数值进行判断，如果为一位数，则停止，如果为多位数，迭代执行上述过程，直到得到一位数为止。

实验环境：通过在VC6.0平台上配置MPICH2，然后通过C语言编写并行程序。源程序见附录源程序二。运行过程中的输入输出结果如下图所示，输入数据采用txt文本输入，通过程序读入到内存。

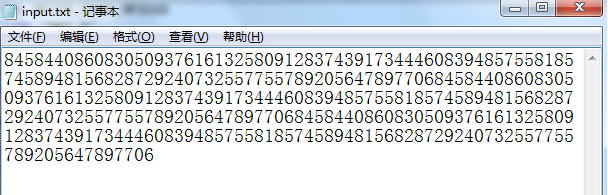


图4 字符串处理问题输入数据文件截图

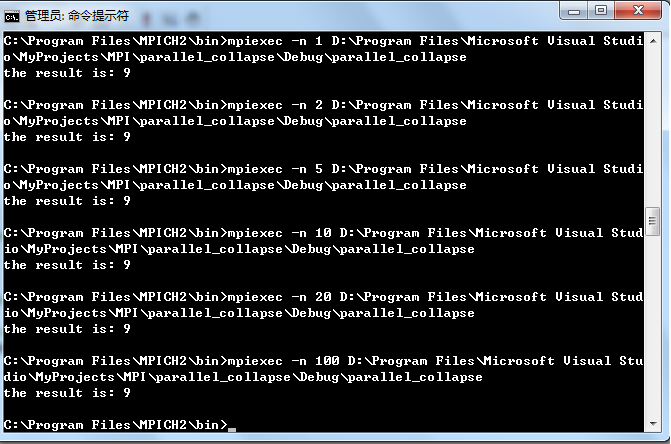


图5 字符串处理问题输出结果图

图5显示了以不同的模拟处理器个数运行程序结果。在写这个并行程序时，我总结的最大问题就是master processor和worker processor的通信，master processor怎么将任务分配给各个worker processor，不仅要注意MPI\_Send和MPI\_Recv数目的严格匹配，还要注意MPI\_Send和MPI\_Recv里面的参数也要严格匹配。

**4总结**

在当前的计算机行业中，并行计算技术具有很重要的地位，特别是近年来，硬件发展的速度逐渐不再符合摩尔定律，其发展速度跟不上软件的需求速度。因此并行计算技术无论是运用在单机还是多机，以及集群上，都具有很重要的意义。近年比较热门的方向，云计算和大数据，都依赖于并行计算技术的实施。此外分布式计算通常与并行计算是具有紧密关联的。在该课程上，我了解并学习了分布式并行计算领域的相关知识，虽然自己的研究方向不是课程的方向，但是自己爱好编程和算法，能够在实验中，进行并行计算方面简单的实验，我觉得自己收获很大。

**致谢** 感谢刘老师的辛勤教学，以及同学们给我在学习和报告中的参考意见.

**参考文献**

1. 杨学军. 并行计算六十年[J]. 计算机工程与科学, 2012, 34(8):1-10.
2. 王之元, 杨学军, 周云, 等. 大规模MPI并行计算的可拓展三模冗余容错机制[J]软件学报, 2012, 4,23(4):1023-1035.
3. 郑文武, 李光绪, 黄执勤. 云计算中的并行计算技术分析[J]. 电信科学. 2011,12:31-37.
4. 潘巍, 李战怀. 大数据环境下并行计算模型的研究进展[J]. 华东师范大学学报（自然科学版）, 201. 9,5:44-49.
5. 王庆先, 孙世新, 尚明生, 刘彦兵. 并行计算模型研究[J]. 计算机科学,2004. 31（9）：128-130.
6. 陈国良, 苗乾坤, 孙广中, 徐云, 郑启龙. 分层并行计算模型[J]. 中国科学技术大学学报, 2008, 38（7）:841-846.
7. 王海涛. 刘淑芬. 基于Linux集群的并行计算[J]. 计算机工程. 2010,36(1):64-66.
8. 卢风顺, 宋君强, 银福康, 张理论 , CPU/GPU协同并行计算研究综述, 2011,3,38(3): 5-9.
9. 蔡勇, 李光耀, 王琥, GPU通用计算平台上中心差分格式显式有限元并行计算[J]. 计算机研究与发展, 2013,50(2): 413-418.
10. zhaolong Ning, Qingyang Song, Yejun Liu, Fanzhao Wang, Xinyu Wu.. Markov-based vertical handoff decision algorithms in heterogeneous wireless networks[J]. Computers and Electrical Engineering, 2014,40:456-472.
11. Jin Zhu,Limin Xu, Lin Yang, Wangqing Xie. An Optimal Vertical Handoff Decision Algorithm for Multiple Services with Different Priorities in Heterogeneous Wireless Networks[J]. Wireless Pers Commun, 2015,83:530-534.
12. Rong Chai, Huili Zhang, Xiaoyu Dong, Qianbin Chen, Tommy Svensson. Optimal joint utility based load balancing algorithm for heterogeneous wireless networks[J]. Wireless Netw, 2014.1.31.