

**研究生学位论文开题报告**

**报告题目 面向大数据应用的硬件加速结构研究**

**学生姓名 马丽娜 学号 201328013229036**

**指导教师 范东睿 职称 研究员**

**学位类别 工学硕士**

**学科专业 计算机系统结构**

**研究方向 处理器结构**

**培养单位 中国科学院计算技术研究所**

**填表日期 2015.12.01**

**中国科学院大学制**

**填 表 说 明**

1. 本表内容须真实、完整、准确。
2. “学位类别”名称填写：哲学博士、教育学博士、理学博士、工学博士、农学博士、医学博士、管理学博士，哲学硕士、经济学硕士、法学硕士、教育学硕士、文学硕士、理学硕士、工学硕士、农学硕士、医学硕士、管理学硕士等。
3. “学科专业”名称填写： “二级学科”全称。

**报告提纲**

1. 选题的背景及意义
2. 国内外本学科领域的发展现状与趋势
3. 课题主要研究内容、预期目标
4. 拟采用的研究方法、技术路线、实验方案及其可行性分析
5. 已有科研基础与所需的科研条件
6. 研究工作计划与进度安排
7. 参考文献

1. 目录

[1 本文的研究背景和意义 6](#_Toc458526062)

[2 国内外本学科领域的发展现状与趋势 6](#_Toc458526063)

[2.1 大数据应用领域Benchmark研究现状 6](#_Toc458526064)

[2.2 硬件加速结构研究现状 8](#_Toc458526065)

[2.2.1 GPU加速 8](#_Toc458526066)

[2.2.2 特定功能算法的专用加速器 9](#_Toc458526067)

[2.3 面向大数据应用的硬件加速结构研究现状 10](#_Toc458526068)

[3 3 课题主要研究内容、预期目标 10](#_Toc458526069)

[3.1 研究内容 10](#_Toc458526070)

[3.2 预期目标 10](#_Toc458526071)

[4 拟采用的研究方法和技术路线 11](#_Toc458526072)

[4.1 大数据应用Benchmark研究范围 11](#_Toc458526073)

[4.2 Benchmark特征提取研究方法 14](#_Toc458526074)

[4.2.1 Benchmark分析方法 14](#_Toc458526075)

[4.2.2 Sort算法的具体分析过程 15](#_Toc458526076)

[4.2.3 特征提取 19](#_Toc458526077)

[4.3 面向大数据应用的硬件加速结构设计 20](#_Toc458526078)

[4.3.1 加速结构 20](#_Toc458526079)

[4.3.2 实验验证平台 22](#_Toc458526080)

[5 已有科研基础与所需的科研条件 23](#_Toc458526081)

[5.1 已有科研基础 23](#_Toc458526082)

[5.2 所需的科研条件 23](#_Toc458526083)

[6 6 研究工作计划与进度安排 23](#_Toc458526084)

[7 7 参考文献 23](#_Toc458526085)

# 本文的研究背景和意义

星载计算系统是计算机技术在空间环境下的应用,负责完成空间飞行器的控制和数据处理任务。由于空间环境的恶劣条件，从而对星载计算系统在性能、可靠性和成本上提出了巨大的挑战。在高昂的研究与制造费用、有限的硬件资源下,要确保海量数据处理的高可靠性是一项困难又关键的任务。

设计一个高速、可靠并且在成本上可接受的星载数据处理系统对于宇宙科学探索及完成预定科学任务具有重大意义。

在星载计算机系统中处理器性能的稳定与可靠，在整个系统的稳定与可靠性中占有重要地位，因此选择合适的处理器架构变得相当重要。目前，在世界范围内星载计算机系统中所使用的处理器架构只有两种，一种是由美国使用的POWERPC架构，另一种就是欧洲主导的SPARC架构。

由于中国与美国的战略关系，因此中国的星载计算机不太可能使用POWERPC架构，因此SPARC架构目前正逐渐被国内相关星载处理器研制单位所采用，特别是因为SPARC架构的开源优势，可以替资本并不雄厚的本土IC设计企业省下一大笔开发经费。

# 国内外本学科领域的发展现状与趋势

## 大数据应用领域Benchmark研究现状

Benchmark研究一直是计算机体系结构领域一个热门的研究方向，也是当今被人们普遍接受的一种计算机性能评测方式。Benchmark研究一方面能够指导消费者购买更加合适的电子产品，另一方面能够指导计算机研究者做出有针对性的计算机性能改进设计。

如今，大数据应用已经成为了学术界的热门话题，针对大数据应用领域的benchmark研究成果也有所积累，其中经典的大数据benchmarks总结见表1：

Table 经典大数据Benchmark内容总结

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Benchmarks** | **开发商** | **关注点** | **Application** | |
| HiBench | Intel | Hadoop | Micro Benchmarks（Sort, WordCount, TeraSort） | |
| HDFS | |
| Web Search | |
| Machine Learning | |
| Data Analytics | |
| LinkBench | Faebook | MySQL | DataBase | |
| YCSB | Yahoo | NoSQL | NoSQL Framework | |
| CloudSuite | CloudSuite | Scale-out | Data Analytics | Machine learning |
| Data Caching （Memcached） | |
| Data Serving （Cassandra ，NoSQL） | |
| Graph Analytics | |
| Media Streaming | |
| SW Testing as a Service | |
| Web Search | |
| Web Serving | |
| BigDataBench | 中科院计算所 | Internet Services | Micro Benchmarks | |
| Basic Datastore Operations | |
| Relational Query | |
| Search Engine | |
| Social Networks | |
| E-commerce | |
| DCBench | 中科院计算所 | Websites and Web services | Data Analytics（Basic operation，Classification，Cluster,Feature reduction,Vector calculate） | |
| Social Networks(Recommendation，Association rule mining，Segmentation，Graph mining) | |
| Warehouse operation | |
| Service（Search engine） | |
| Interactive real-time application（Media streaming） | |
| TPC | 事务处理性能委员会 | 测试框架 | — | |

HiBench[5]是一个由Intel开发、用于Hadoop集群性能测试的程序集，由五类benchmark组成，分别为Micro benchmarks、HDFS benchmarks、web search benchmarks、machine learning benchmarks和data analytics benchmarks，从基本计算能力、HDFS吞吐量、网络服务性能、机器学习性能和数据处理能力五个方面Hadoop集群系统进行全方位测评。

LinkBench[6]是Facebook开发的针对社交网络数据库负载分析的性能测试集合。社交网络中展现数据的一种最重要的方式就是社交图谱（social graph），人、文章、评论和页面都是通过节点间的不同关系类型相互关联的，关系型数据库自然也成为社交网络的基础。MySQL是Facebook基础架构中的重要组件，因此LinkBench的关注点集中在MySQL上。

YCSB（Yahoo! Cloud Serving Benchmark）[7]是2010年Yahoo研究院针对NoSQL系统开发的开源基准测试框架。随着大数据时代的到来，传统的关系数据库越来越不能够满足对数据处理的需求，“Not only SQL”营运而生并迅速发展。YCSB可以对不同NoSQL系统进行统一的基准测试，帮助开发人员选择更合适的数据库系统。目前YCSB测试结果得到较多认同，HBase官方文档中也将其推荐为性能测试的基准。

CloudeSuite[8]是针对云计算应用的标准测试程序集。针对云计算应用的Scale-out特点，CloudSuite依据当前数据中心的流行程度，选取了网络服务、网络搜索、数据分析、数据缓存、数据服务、图分析流媒体和软件测试等8个常用负载作为其集中测试点。

BigDataBench[9]是一个抽取Internet典型服务而构建的大数据基准测试程序集。涵盖了完整的系统软件栈，覆盖了实时分析、离线分析和数据服务应用类型。其特点是保留了真实的应用场景，使用真实数据集，对系统的测试更加真实可靠。

DCBench[10]是有中科院计算所发布的一套针对数据中心负责的测试集合。DCBench以数据中心系统为目标，具有代表性明确、多样化编程模型、分布式和使用新型技术等四个特点。

TPC(Transaction Processing Performance Council)是有数十个计算机软硬件厂家创建的非营利性组织。TPC的主要功能是定制商务应用Benchmark的标准规范、性能和价格度量，管理测试结果的发布。TPC不给出基准程序代码，而只是给出基准程序的标准规范。

## 硬件加速结构研究现状

随着电子计算机的普及和新应用需求的不断产生，传统CPU体系结构越来越明显的面临着存储墙、功耗墙和指令级并行墙等问题。虽然CPU结构也提供了诸如SIMD数据并行等处理单元，但是其性能仍然远远满足不了新应用的需求，因此，体系结构设计的加速研究也一直是计算机领域一个很重要的课题。体系结构领域硬件加速研究主要包括相对通用的众核、异构GPU加速和相对专用的加速器设计如陈等人的DianNao[12]等。

### GPU加速

在过去，为了减轻CPU负担，加速玩家游戏体验，将诸如顶点计算、光照计算等大量的图形处理操作交给GPU（Graphic Process Unit）来处理，随着性能的不断提高，如今，GPU已经演化成为一种高效存储、高浮点计算能力的多线程众核处理器。GPU体系结构结合了共享存储器模型和消息传递模型，采用了类似于分布式共享存储的设计，同时具备可伸缩性和良好的可编程性。

GPU芯片[13]由一个或者多个相互独立工作的计算单元SM（Stream Multiprocessor）流多处理器构成，通过片上网络互联，如图1所示。每个SM包含一定数量的SP流处理器（Stream Processor）。SP有独立的寄存器和指令指针，但没有取指和调度元件，而SM包括完整的取指、译码、发射和执行单元等。因此，SM才能算得上是GPU的完整核心，规模的伸缩也是以SM为基本单位。从结构上看，一个SM相当于一个n路的SIMD（Single Instruction Multiple Data,单指令流多数据流）处理器，同时又是自动向量化的，因此NVDIA将其命名为SIMT（Single Instruction Multiple Thread，单指令流多线程）。

GPU通过大量且简单的流处理器共同执行来提高运算和数据的吞吐率，通过大量由硬件管理的细粒度轻量级线程的切换，隐藏延迟效果。相对于CPU存储系统，GPU有数个存储控制单元，内存带宽明显高于CPU，没有复制的缓存体系和一致性机制，因而没有访村失效所带来的巨大延迟。

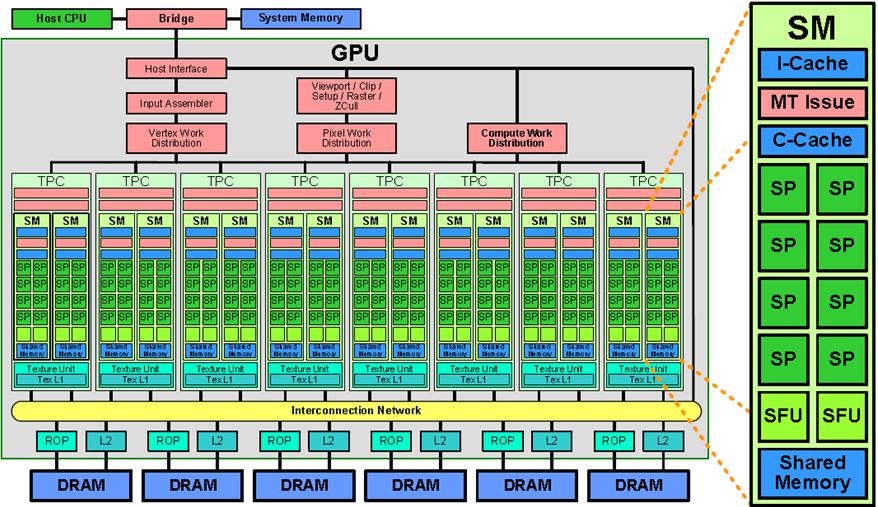


Figure GPU结构示意图

### 特定功能算法的专用加速器

伴随着半导体工艺的发展，处理器微结构设计越来越复杂，时钟频率不断提升，处理器的功耗也明显增加。在纳米工艺条件下，线延迟、功耗墙、存储墙等问题逐渐凸显，成为微结构设计瓶颈。面向特定的应用领域或特定算法设计硬件加速器正是解决这些问题的方法之一。最近的研究表明，专用加速器能够得到主流通用处理器1000~10000倍的效率[14]。

CryptoManiac[15]是由吴等人设计的一款快速灵活的针对Cryptographic Services的协处理器，主要解决加密和解密问题。CryptoManiac在0.25um工艺的物理设计下能够得到比600MHz Alpha 21264处理器2.25倍的加速，同时，在相同工艺下面积却只有其1/100。

针对机器学习领域的CNN和DNN算法，陈云霁等人设计的DianNao[16]和DaDianNao[17]加速器取得了很高的性能收益。DianNao中强调了存储对加速器设计、性能和功耗的重要影响，实现了在3.02mm2 小面积、485mW超低功耗下得到452GOP/s的带宽。与128位2GHz的SIMD处理器相比得到117倍的加速，同时功耗仅是其1/21。

## 面向大数据应用的硬件加速结构研究现状

面向大数据应用的处理器不同于传统的高性能处理器，需要运行许多大数据应用作业，数据特点离散性强，耦合性弱。根据前面的分析，对于一些新兴的应用领域，学术界已经有了很多的相关研究，如云计算、大数据、数据中心等方向，已经有了比较权威的Benchmark集。

然而，从上面的介绍可以看出，这些Benchmark主要是用于对系统级进行性能测试和评价的，而不是对芯片级的测试和评价。例如，HiBench用于对Hadoop集群性能的测试，YCBS用于对NoSQL系统的测试，LinkBench用于对社交图谱数据库的测试等等。系统级的评价主要关注系统运行的整体性能，如集群内部协同工作的效率，板间互连和通信的效率，系统软件栈的性能等。而芯片级的测试和评价则主要关注芯片内部的工作情况，如众核处理器中多个核的并行处理能力，线程在处理器中的调度效率，共享存储的利用效率，cache的命中率，数据通路的使用效率等。因此，这些Benchmark虽然具有大数据应用的特点，但是不适合于高通量处理器的性能测试和评价。

本文工作之一就是基于现有benchmark研究基础之上，抽取定义芯片级测评的大数据应用workload。

针对大数据应用，目前常用的加速方式是GPU加速和通过优化分布式框架在集群上进行软件优化层面的加速，并没有针对性的对单节点处理器本身进行加速的相关研究。因此本文的第二个工作就是基于抽取的workload进行分析聚类，归纳出大数据应用的单节点特性，借鉴已有的硬件加速技术，提出面向大数据应用的硬件加速结构。

# 3 课题主要研究内容、预期目标

## 研究内容

* 大数据Benchmark分析及特征提取；
* 基于特征提取从指令级、核内、核间和片间四个层面上归纳设计，提出适合面向大数据应用的硬件加速方案；
* 对硬件加速方案进行试验验证。

## 3.2 预期目标

* 得到大数据应用映射到芯片层面的特征和设计需求
* 设计面向大数据应用的硬件加速器
* 实验分析，得到理想的加速结果和分析

# 拟采用的研究方法和技术路线

## 大数据应用Benchmark研究范围

针对大数据应用领域，Big Data Analytics、Database、Machine Learning、SocialNetworks、E-Commerce、Web search和Media Streaming等7个当前火热的应用方向占据了当前已有的大数据测试集合的大部分。通过简单统计分析可以看到，以上7个应用在表1中经典的大数据Benchmark 占比总和超过96%（表2），能够有效的覆盖大数据应用。



Figure 主要应用在Benchmark中占比

依据UC Berkeley针对大数据和高性能应用的一份报告[18]，其中，HPC和Genomics属于高性能计算领域，Web Search, Social Network, Data Base, Big Data Analytics, Big Data Multimedia, Advertising, Finance, Visualization等为大数据相关的应用内容。而根据对应应用的Benchmark组成进行分析，我们将可穿戴与可视化合并为Visualization，将Finance和Advertising合并为E-commerce。

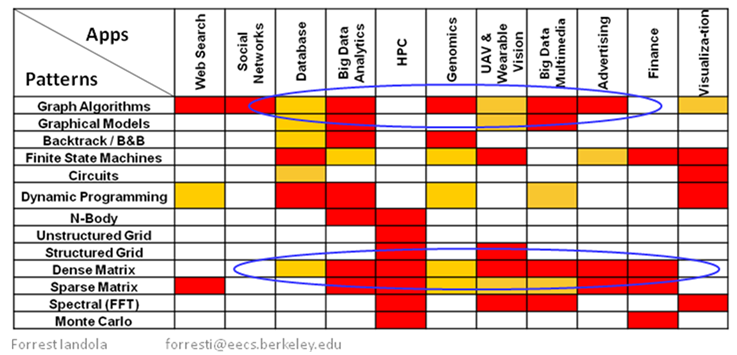


Figure UC Berkeley提出大数据及高性能应用与计算Patterns对应表

综合前面的分析，同时考虑到新兴的热门应用如“数据挖掘”等，本文最终选取了9个应用作为最终的研究目标，并且根据每种应用分类下算法的重要性进行打分，汇总如表2：

Table 大数据应用Benchmark分析汇总

|  |  |  |  |
| --- | --- | --- | --- |
| **应用分类** | **解决方案** | **具体算法** | **打分** |
| **（一）**  **Big Data Analytics**  **大数据分析** | Basic operation | Sort | 5 |
| grep | 5 |
| wordcount | 5 |
| Classification | Naïve Bayes | 5 |
| SVM | 5 |
| Cluster | K-means | 5 |
| Fuzzy C-means | 3 |
| Segmentation | HMM | 4 |
| regression analysis | regression analysis | 4 |
| **（二）**  **Data Mining**  **数据挖掘** | 分类和回归 | 回归分析 | 4 |
| C4.5决策树 | 5 |
| CART决策树 | 5 |
| 朴素贝叶斯网络 | 5 |
| 支持向量机（SVM） | 5 |
| K最近邻分类（KNN） | 5 |
| AdaBoost迭代 | 5 |
| 关联分析 | Aprior算法 | 5 |
| FP-TREE算法 | 4 |
| 灰色关联分析 | 4 |
| 聚类分析 | K-mean | 5 |
| 最大期望（EM） | 5 |
| 偏差检测 | 基于模型的技术 | 4 |
| 基于邻近度的技术 | 4 |
| 基于密度的技术 | 4 |
| 基于聚类的技术 | 4 |
| **WEB**挖掘 | PageRank | 5 |
| 链接分析算法（HITS） | 4 |
| **（三）**  **Machine Learning**  **机器学习** | 半监督学习 | 最大似然 | 4 |
| EM算法 | 5 |
| 基于图 | 基于高斯场(Gaussian Fields)和调和函数(Harmonic Functions)的算法 | 5 |
| 概率图模型 | 贝叶斯信念传播算法 | 5 |
| **（四）**  **Web Search**  **网页搜索** | 网络爬虫 | BFS遍历算法 | 5 |
| 网页分析算法 | 5 |
| DFS遍历算法 | 5 |
| 网页过滤技术 | 4 |
| Hash Table（记录已被下载的网页） | 4 |
| 建立网页索引 | Bool | 5 |
| Index | 4 |
| 搜索结果的排名 | PageRank算法 | 5 |
| TF-IDF算法 | 5 |
| **（五）**  **Social Networks**  **社交网络** | 基础部分 | 内存缓存系统 | 5 |
| 高性能图片存储系统 | 5 |
| 分布式存储系统 | 5 |
| 数据库 | 5 |
| 必要部分 | 分发方式 | 4 |
| 动态网页处理系统 | 4 |
| 日志系统 | 4 |
| 完善部分 | 社交网络影响力最大化问题 | 5 |
| 推荐算法 | 5 |
| 隐私保护 | 5 |
| 数据分析 | 5 |
| **（六）**  **Database**  **数据库** | B+树索引文件 | B+树查询 | 5 |
| B+树插入 | 5 |
| B+树删除 | 5 |
| 外部排序归并算法 | 归并算法 | 5 |
| 连接运算 | 嵌套循环连接 | 4 |
| 排序-合并连接 | 5 |
| 并行嵌套循环连接算法 | 5 |
| 哈希连接 | 4 |
| 磁盘臂调度算法 | 电梯算法 | 3 |
| **（七）**  **E-commerce**  **电子商务** | 搜索系统 | 分词技术 | 4 |
| 根据商品不同的关键字建立索引(Index) | 4 |
| 响应用户查询(Query) | 4 |
| 数据库操作 | Select Query | 3 |
| Aggregate Query | 3 |
| Join Query | 3 |
| 推荐系统 | 协同过滤算法 | 5 |
| 基于内容的推荐算法 | 5 |
| 基于聚类算法的推荐 | 4 |
| 基于产品到产品的推荐 | 3 |
| 系统框架 | RUBiS server | 5 |
| **（八）**  **Big Data Multimedia**  **大数据多媒体** | 编解码技术 | MPEG | 2 |
| H264 | 2 |
| 网络传输技术 | 资源预留协议RSVP | 2 |
| 实时传输协议RTP | 2 |
| 实时传输控制协议RTCP | 2 |
| 实时流协议RTSP | 2 |
| 流媒体Benchmark框架 | Streaming Media | 3 |
| **（九）**  **Visualization**  **可视化** | 多维数据降维 | 主成分分析算法（PCA） | 5 |
| 多尺度分析算法（MDA） | 5 |
| 自组织映射算法（SOM） | 5 |
| 多维数据可视化 | 基于降维映射的技术 | 3 |
| 基于层次的多维可视化技术 | 3 |
| 基于几何的多维可视化技术 | 3 |
| 网络可视化 | 力导引布局算法（FDA） | 4 |
| Frunchterman-Reingold算法（FR） | 4 |
| 基于地图布局算法 | 2 |

对表2中的所有Benchmark从计算特征、访村特征、数据并行、线程依赖、I/O使用等不同方面进行特征分析，归纳大数据应用领域对结构设计的需求。最后通过统计和聚类的方式，得到最突出的几个特征。

## Benchmark特征提取研究方法

上述表2汇总了针对大数据应用领域所需要分析的Benchmark以及其具体对应的算法，并且从不同的方面论证了其作为大数据应用领域的调研目标的全面性。针对以上9个大数据领域应用，进行具体分析并提取出显著特征是本文的重要工作之一。

### Benchmark分析方法

根据每一个应用分类，根据其需要解决的问题和真实应用，进行具体的应用分类，并且针对每一个应用分类抽取workload，根据每个workload在所对应的应用分类中使用量的多少进行打分排序，最后得到需要分析的具体算法。作为举例，在Big Data Analytics领域，最后抽取并研究的workloads如表3所示：

Table Big Data Analytic算法分类

|  |  |  |
| --- | --- | --- |
| **领域分类** | **Workload/算法** | |
| **Basic operation** | Sort | Terasort |
| Shuffle |
| Grep | KMP |
| Boyer-Moore |
| Wordcount | |
| **Classification** | Naïve Bayes | |
| SVM | |
| **Cluster** | K-means | |
| Fuzzy c-means | |
| **Segmentation** | HMM | |
| **regression analysis** | regression analysis | |

接下来，我们对表中所抽取的每一应用分类对应的所有算法进行详细分析：

* 分析算法，得到算法流程图，并估计出重要操作步骤所占比重；
* 分解算法特征；
* 将算法关键步骤分解对应到原子函数和指令操作；
* 得到算法分解tree图；
* 分别从计算特征、访存特征、并行性、线程间关系和I/O操作等五个方面进行特征分析，并得到算法对体系结构设计从指令级、协处理器级、内部高速总线级和外部高速总线级四个不同层次的设计需求。

通过上述方法，我们能够得到所有9个领域Benchmark所对应workload的特征分析和对设计的具体需求，最后通过聚类和原子操作统计两种方式，得到大数据应用的最突出重要的几个特征，指导后续体系结构设计思路。

### Sort算法的具体分析过程

大数据分析是一个比较广泛的概念，凡是具有大数据特点的应用，都会或多或少用到大数据分析的相关工作，例如，多数互联网服务会用到词频统计来提高服务质量，互联网服务中的推荐系统会用到聚类相关的算法等等。当然，也有应用领域主要的工作就是分析数据，比如数据挖掘，数据挖掘的主要目的就是通过对大数据的分析，得出有效的结论。

Basic operation是指在大数据领域经常会用到的几个算法，几乎所有应用的后台工作都要涉及到，是非常重要的，在大数据应用分析和Benchmark设计过程中是必须要有的，这里选取了最常用的Sort、Grep、WordCount几个算法。

#### Sort算法分析

无论是在传统应用领域还是大数据分析领域，排序都是基本的算法之一。大数据领域有其特有的排序算法。Terasort是一个典型的大数据排序算法，是Hadoop中提供的一个排序工具。流程步骤如图4。

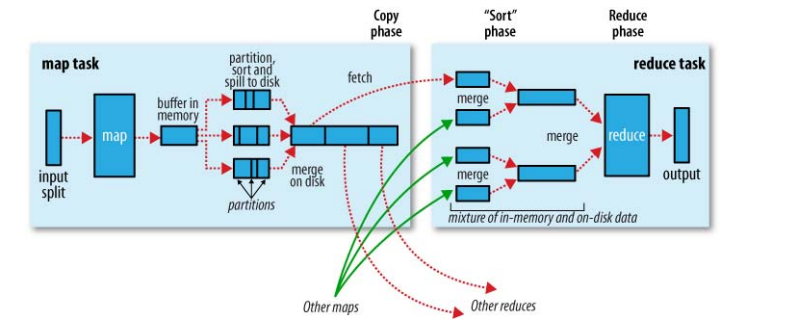


Figure 基于MapReduce框架的terasort算法

基于MapReduce框架的Terasort的过程如下所示：

1)一般情况下，作业会需要指定input目录和output目录；

2)作业的Mapper根据设置的InputFormat来从input目录读取输入数据，分成多个splits; 每一个split交给一个mapper处理；

3) Mapper的输出会按照partions分组，每一个partion对应着一个reducer的输入； 在每个partion内，会有一个按key排序的过程，也就是说，每一个partion内的数据是有序的；

4)当处理完combiner和压缩后（如果有设置），map的输出会写到硬盘上。map结束后，所在的TT会在下一个心跳通知到JT；

5)每一个reducer查询JT了解到属于自己对应partition的mapoutput数据的对应的TT位置，然后去那copy到本地(HTTP协议)；

6) Copy并保存到本地磁盘的过程同mapper端的输出保存过程非常相似。等到reducer获取到属于它的所有mapoutput数据后，它会保持之前mapper端的sort顺序，把这些mapoutput合并成较集中的中间文件（个数取决于数据大小和设置）。为了节省io的开销，merge会保证最后一轮是满负荷合并；并且，merge的最后一轮输出会直接在内存输入给reducer；

7)Reducer的输出按照OutputFormat来保存到output目录。

根据上述对Terasort算法的分析，可以总结出Terasort程序在执行过程中的主要操作，流程图如图5所示。

**错误! 不能通过编辑域代码创建对象。**

Figure terasort算法流程图

Terasort算法处理输入文本的排序问题。其主要是利用MapReduce框架实现，由MapReduce框架负责对数据的分类与合并。算法首先需要对所有选取的数据（M-1）进行数据采样，并且对这(M-1)个数据排序。以这个数据为基准建立Trie Tree，对未分类的数据进行比较分类。分类之后每组Partition[]进行merge合并，并且对合并之后总数值进行分组排序，最终得出排序结果。

#### Terasort算法特征分解

* **步骤1：**在算法的输入初期，需要将文件的输入流切割为固定的文件输入流，便于系统统一处理。处理的动作主要为判断输入字符个数是否达到上限。处理器上的操作可看做ADD累加操作。
* **步骤2：**在输入的所有数据中，选取Key值，并且对Key值进行排序，建立Trie Tree。
* **a2.1**:将数据分为M组，则随机或是按照一定规则选取（M-1）个数据，主要包括LOAD和比较指令。
* **a2.2:**对选取的(M-1)个Key值进行排序，包括比较指令、交换指令。
* **步骤3：**对于分一个分类为split的数据组，进行比较Trie Tree树插入操作。
* **a3.1:**对于每一个split中的值，与Key值数组中的值进行比较，对split中的值归类，主要包括LOAD指令、比较指令。
* **步骤4：**将步骤3中的数据进行merge合并，并且判断其是否满负荷。主要包括：LOAD指令、ADD累加操作。
* **步骤5：**对于合并得到的M个数组，对每一个数组的数据进行排序。主要包括LOAD指令、比较指令、交换指令。

#### Terasort算法关键阶段对应操作分解

本节统计出Terasort算法几个关键步骤及操作分析。如表4所示：

Table Terasort算法关键阶段对应操作

|  |  |  |
| --- | --- | --- |
| 算法关键步骤 | 代表函数 | 对应指令操作 |
| 随机选取Key值，并排序。 | 不同排序算法。 | 比较操作和swap操作。 |
| 比较并插入Trie Tree操作。 | Key[i-1]<=Samper[n]<Key[i]  Partition[i][m] = n ; | 比较操作 |
| Merge合并后对数组排序 | Merge(); | 比较操作 |

#### Terasort算法操作tree示意图

**错误! 不能通过编辑域代码创建对象。**

Figure Terasort算法操作tree示意图

#### terasort算法特征分析

Table sort算法特征分析

|  |  |
| --- | --- |
| **编号项** | **特征** |
| **计算特征** | Terasort算法的主要操作在于对数据的比较和交换，包括简单的加减（ADD/SUB）操作、大量比较(CMP)操作及交换（SWAP）操作。假设数据量为N，其算法的时间复杂度为O(NlogN)。 |
| **访存特征** | Terasort算法需要对大量的样本数据进行排序，访存量比较大，属存储密集型应用。Terasort分几个阶段执行，map阶段是对数据进行分割和映射，map到reduce之间为每块数据内部排序，reduce阶段进行合并和按序输出，所以访存具有较好的局部性。 |
| **并行性** | Terasort算法具有较好的并行性。因需要排序的数据量巨大，所以算法前两个步骤（Trie Tree的建立和插入）可以多线程/进程并行执行。 |
| **线程间关系** | 每个线程负责自己部分数据的排序比较，最后只需结果合并，因此在线程执行过程中无数据依赖。 |
| **I/O** | 因排序数据量通常为TB级，且属存储密集型应用，所以需频繁换入换出，IO使用较多。 |

Table sort算法对体系结构的设计需求

|  |  |
| --- | --- |
| **结构设计** | **具体需求** |
| **指令级** | 排序算法涉及的指令主要为内存的读写操作（LOAD/STORE）、数据对比操作（CMP）、简单的加减运算操作（ADD/SUB）及数据交换操作（SWAP）。**运算模式单一：主要为“取数—对比—交换写回”。**因此，在指令集上的设计可以考虑添加此运算模式的“复合指令”，即“取数—对比—交换写回”三步可以在一条复合指令内完成。 |
| **协处理器** | 按照运算模式，可以对“取数—对比—交换写回”这种操作进行硬件加速。设计符合此种计算模式的硬件处理单元阵列，大批量并行处理非相关数据块，达到协处理器加速的目的。 |
| **内部高速总线** | 内部处理单元应设置**共享存储**（共享Cache或SPM等），当一个处理单元处理完一部分数据时可以在片内与其它处理单元的结果再进行**合并排序**。内存总线需要保证处理单元之间的通信和数据搬运性能。内部数据搬运可以设计**数据块迁移加速部件**，保证内部数据的迁移能够高效执行。另一方面，处理单元与片外存储也有频繁的数据交换操作，可以考虑在片内互联网络之外**使用专用的存储访问数据通路。** |
| **外部高速总线** | TeraSort排序的数据量巨大，需要频繁的与外存进行数据交换。无论使用加速卡、多处理器或多机进行并行加速时，都需要外部能提供类似PCIe的连接通道，以及类似DMA这样的块传输机制，达到快速高效的数据迁移。 |

### 特征提取

按照前面sort算法的分析方法，对表2中多有涉及到的算法进行详细分析，分别从计算特征、访存特征、并行性、线程间关系和I/O操作等五个方面进行特征分析，并得到算法对体系结构设计从指令级、协处理器级、内部高速总线级和外部高速总线级四个不同层次的设计需求。最后一步我们要做的是归纳总结出大数据应用领域的现状重要的特征。

使用方法：占比统计法。根据对每个workload算的详细调研及各部分原子操作在算法中的占比，合并所有几十个算法的原子操作，根据统计结果，得出重要的特征点。统计将分别从指令级、协处理器级、内部总线和外部总线四个层次进行，表7是简单的指令级统计示例：

Table 原子操作统计表



以上为本部分工作的实施思路，由于工作目前正在进行中，所以尚未得出最终结论。

通过不完全统计可以发现，sort算法不仅仅在传统应用领域是一类经典算法，在大数据应用领域，其使用频率和所占比重尤其突出。根据目前不完全分析发现，sort算法在大数据分析、数据库、社交网络、大数据多媒体等4个应用方向均属于重要算法，且面对TB数量级的数据排序，排序算法的执行效率直接影响到相关应用的效率和服务质量。因此，sort运算占比大是大数据应用领域的一个重要特征，后面将以此特征为例，设计硬件加速。

## 面向大数据应用的硬件加速结构设计

根据上述全面的大数据应用领域Benchmark分析，对每个workload从不同粒度上进行特征分析和体系结构设计需求分析，通过从四个不同层面的占比统计，归纳总结所有的workload，最终得出最重要的几个特征需求，并以此为指导，从指令级、协处理器级、内部高速总线和外部高速总线四个层面分析设计研究加速器。

由于本部分工作正在进行中，仅以sort运算在大数据应用领域占比大这一特征为例，简单叙述加速结构设计思路。

### 加速结构

Sort算法分为内存排序和外存排序两种，内存排序最显著的特征就是“取数-比较-交换写回”的固定操作模式，外存排序最重要的操作就是对两个有序数列进行归并，归并的基本操作也是“取数-比较-交换写回”模式。因此，针对这两个点进行硬件加速结构设计，sort算法的执行速度也就得到了解决。

方案一：增加简单指令CMPX（比较并交换）

从简单指令添加的角度对结构进行设计相对简单：（1）需要配合相应的指令集进行合理的编码；（2）需要修改优化编译器，让编译器能够在条件恰当的情况下优先选用所增加的指令。

“取数-比较-交换写回”这一固定操作模式，在常规指令集模式下，需要取数后对两个数据进行比较，然后根据比较的结果分支进行交换或不交换操作，而此处的比较并没有可预测性，因而预测错误也常常会带来不必要的代价。将这一模式复合封装成一条指令CMPX（比较并交换），则可以在性能上有很大的提高。

方案二：封装MERGE功能指令部件

FISC（Function Instruction Set Computer）结构是由陈等人[19]提出的一种将专用加速部件与通用加速部件有效整合在一起的解决方案。由于sort算法中的二路归并是并行性相对较差的一个操作步骤，对大规模有序数列的归并是一个很费时的操作，因此可以将二路归并操作设计成一个专用功能部件，并且以FISC方式整合到通用核中。



Figure FISC架构图

MERGE部件内部对应的逻辑相对简单，简单的二级流水线能够保证二路归并每拍都得到一个顺序的数据。或者设计成为必要的阵列，以增加合并速度。

### 实验验证平台

本研究题目提出的所有硬件加速设计方案，都将会在支持千核的DPU模拟实验平台上进行验证，得到对应设计的优化效果，以证明结构设计的有效性。DPU模拟平台是由AMS实验室基于SimICT框架自行研制的一款支持模拟器，支持ARMv6指令集并支持SMT、可配置片上互联组件（例如：Mesh，双环，多级总线）、Cache组件、MCU组件、Memory组件，支持各种性能数据统计和功耗数据统计及结构化输出。其模块关系图例如下：



Figure DPU模拟器模块关系图例

用户可以使用Core、片上网络、Cache、Memory、SPM等组件任意搭建目标系统。其中Core组件支持ARMv6指令集，支持ARM1176流水线，SMT，Pthread，性能统计输出等内容。Cache组件支持多级配置，支持大小、类型、替换算法等参数的配置，支持性能统计输出等功能。片上网络包括Mesh组件和双环组件，支持虚通道、数据包分片等技术，支持性能统计输出及自动化分析等功能。Memory组件支持虚实地址转换，支持万级任务加载和管理等功能。

目前，DPU模拟器已经稳定且组件灵活可配，为本题目的后续研究评估提供了有力保障。

# 已有科研基础与所需的科研条件

## 已有科研基础

* 已完成大数据领域Benchmarks表3的调研分析；
* 已有千核模拟器组件和对SimICT框架使用的熟悉，能够搭建模拟系统。

## 所需的科研条件

* 多核处理器服务器平台；
* DPU模拟器。

# 6 研究工作计划与进度安排

* 2015/10~2015/12 完成对大数据应用的分析工作，得到显著的大数据应用特点和结构需求，完成开题报告及开题答辩
* 2015/12~2016/3 完成硬件加速结构设计和所需实验，撰写毕业论文，并完成中期答辩
* 2016/3~2016/5 完成毕业论文和答辩工作

# 7 参考文献

[1] 张引,陈敏,廖小飞.大数据应用的现状与展望[J].计算机研究与发展,2013,50:216-233.

[2] High Volume Throughput Computing: Identifying and Characterizing

Throughput Oriented Workloads in Data Centers[C] //Proc of the IEEE 26th International Parallel and Distributed Processing Symposium Workshops & PhD

Forum, 2012: 1712-1721.

[3] Characterization of Real Workloads of Web Search Engines. In Proc. IISWC 2011.

[4] Data centers only operating at 4% utilization. 2014. <http://www.environmentalleader.com/2010/07/08/data-centers-only-operating-at-4-utilization/>

[5] S. Huang, J. Huang, J. Dai, T. Xie, and B. Huang. The hibenchbenchmark suite: Characterization of the mapreducebaseddata analysis. In Data Engineering Workshops(ICDEW), 2010 IEEE 26th International Conference on,pages 41–51. IEEE, 2010.

[6] T. G. Armstrong, V. Ponnekanti, D. Borthakur, and M. Callaghan.Linkbench: a database benchmark based onthe facebook social graph. 2013.

[7]B. F. Cooper, A. Silberstein, E. Tam, R. Ramakrishnan, andR. Sears. Benchmarking cloud serving systems with ycsb.In Proceedings of the 1st ACM symposium on Cloud computing, SoCC ’10, pages 143–154, 2010.

[8] M. Ferdman, A. Adileh, O. Kocberber, S. Volos, M. Alisafaee,D. Jevdjic, C. Kaynak, A. D. Popescu, A. Ailamaki,and B. Falsafi. Clearing the clouds: A study of emergingworkloads on modern hardware.Architectural Support for Programming Languages and Operating Systems, 2012.

[9] W. Gao, Y. Zhu, Z. Jia, C. Luo, L. Wang, J. Zhan, Y. He, S. Gong, X. Li, S. Zhang, and B. Qiu.Bigdatabench: a bigdata benchmark suite from web search engines. The Third Workshop on Architectures and Systems for Big Data (ASBD2013), in conjunction with ISCA 2013.

[10]<http://prof.ict.ac.cn/DCBench/>

[11]<http://www.tpc.org/>

[12]Yunji Chen, Tao Luo, Shaoli Liu, Shijin Zhang, Liqiang He, Jia Wang, Ling Li, Tianshi Chen, Zhiwei Xu, Ninghui Sun, and Olivier Temam, "DaDianNao: A Machine-Learning Supercomputer", in *Proceedings of the 47th IEEE/ACM International Symposium on Microarchitecture (MICRO'14),* 2014.

[13]D.Kanter. NVIDIA’s GT200:Inside a parallel processor[T].Real world technologies,2008.

[14]Guha A, Zhang Y, ur Rasool R et al. Systematic evaluation of workload clustering for extremely energy-efcient architectures. ACM SIGARCH Computer Architecture News, 2013,41(2): 22-29.

[15] Wu L, Weaver C, Austin T. CryptoManiac: A fast flexible architecture for secure communication. In Proc. Int. Symp.Computer Architecture, June 30-July 4, 2001, pp.110-119.

[16] Yunji Chen, Tao Luo, Shaoli Liu, Shijin Zhang, Liqiang He, Jia Wang, Ling Li, Tianshi Chen, Zhiwei Xu, Ninghui Sun, and Olivier Temam, "DaDianNao: A Machine-Learning Supercomputer", in *Proceedings of the 47th IEEE/ACM International Symposium on Microarchitecture (MICRO'14),* 2014.

[17]Tianshi Chen, Zidong Du, Ninghui Sun, Jia Wang, Chengyong Wu, Yunji Chen, and Olivier Temam, "DianNao: A Small-Footprint High-Throughput Accelerator for Ubiquitous Machine-Learning", in *Proceedings of the 19th ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS'14),* 2014.

[18] Micheal Anderson,Khalid Ashraf,Gerald Friedland,Forrest landola .Application-Driven Research in the ASPIRE lab.[R]<http://librozilla.com/doc/447428/ucb-ace-lab>

[19] P Chen，L Zhang，YH Han，YJ Chen. A General-Purpose Many-Accelerator Architecture Based on Dataflow Graph Clustering of Applications. 计算机科学技术学报(英文版), 2014, 29(2):239-246

[20]