

数据库系统概论新技术篇

数据仓库与联机分析处理技术(4)

陈红

中国人民大学信息学院

数据仓库与OLAP技术

- ❖ 从数据库到数据仓库
- ❖ 数据仓库的特征与体系结构
- ❖ OLAP的模型与技术
- ❖ 新的研究方向



新的研究方向

❖ 传统问题

- 实体化视图的增量维护
- 数据集成
-

❖ 新的多维数据分析方法

- SKYLINE
- TOP-K
- KNN
- RANK
- MEDIAN

❖ 新的硬件环境

- 内存OLAP
- 多核OLAP
- 基于协处理器的OLAP
- 实时数据仓库

❖ 新的应用场景

- 大数据OLAP
- 流数据的联机分析
- 物联网中的联机分析



新的研究方向

❖ 传统问题

- 实体化视图的增量维护
- 数据集成
-

❖ 新的多维数据分析方法

- SKYLINE
- TOP-K
- KNN
- RANK
- MEDIAN

❖ 新的硬件环境

- 内存OLAP
- 多核OLAP
- 基于协处理器的OLAP
- 实时数据仓库

❖ 新的应用场景

- 大数据OLAP
- 流数据的联机分析
- 物联网中的联机分析



磁盘OLAP的困境

- ❖ 基于磁盘的**OLAP**采用了各种各样的优化策略（如预计算、预建索引、数据聚簇存放），查询性能在一定程度上得到了提高，但是受磁盘**I/O**性能的限制，仍然无法满足用户的需求，更难以支持要求苛刻的决策应用。
- ❖ 据**OLAP SURVEY**调查数据显示，在所罗列的**OLAP**产品的**15**个问题中，“性能太差”排在第一，而且用户关于性能的抱怨以很大的比例逐年上升。
- ❖ 性能问题已成为制约**OLAP**应用的主要障碍。



磁盘OLAP的困境

- ❖ 一个事实是，过去的**20**年里，无论是**CPU**速度还是内存容量，无论是每单位成本的内存容量还是磁盘的存储密度，无论是网络吞吐量还有一些其他的评价标准，都遵循摩尔定律，即按照每**18**个月翻一番的方式提高。
- ❖ 唯独磁盘转速是一个例外。其速度在过去**50**年仅增长了**12.5**倍，从**1956**年的每分钟**1200**转到现在的**15000RPM**。



What is 内存OLAP

- ❖ 基于内存的**OLAP**是指这样的一种技术：从基于磁盘的永久性数据存储中将数据加载到内存，在内存中对数据进行各种各样的**OLAP**分析
- ❖ 相对于传统的基于磁盘的**OLAP**而言，内存**OLAP**的性能特征、优化策略、性能瓶颈主要受半导体**RAM**（内存，而不是磁盘）的影响。







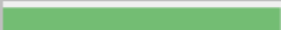



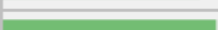







内存OLAP的可行性

❖ 海量数据？

■ BI Survey数据

- 调查数据显示，目前一半以上的用户用于联机分析处理的基础数据通常只有数百兆至数GB；大约70%的用户少于50GB

数据量	用户比例
<200MB	14.9%
<1GB	35.4%
<10GB	56.4%
<50GB	69.9%
<250GB	79.8%
<500GB	85.5%
>500GB	14.5%

Input data volume band	Percentage of those answering	Band
Less than 10 MB	 3.2%	<i>Very small</i> 19.9%
10 to less than 50 MB	 7.4%	
50 to less than 200 MB	 9.3%	
200 to less than 500 MB	 6.5%	<i>Small</i> 15.5%
500 to less than 1000 MB	 8.9%	
1 to less than 5 GB	 14.9%	<i>Medium</i> 21.0%
5 to less than 10 GB	 6.1%	
10 to less than 20 GB	 6.4%	<i>Large</i> 23.2%
20 to less than 50 GB	 6.9%	
50 to less than 100 GB	 5.2%	
100 to less than 250 GB	 4.7%	
250 to less than 500 GB	 4.7%	<i>Very large</i> 20.4%
500 to less than 1000 GB	 5.9%	
1 to less than 2 TB	 4.4%	
2 to less than 5 TB	 3.3%	
5 TB or more	 2%	



内存OLAP的可行性

- ❖ 内存容量的增长速度远快于用户数据量的增长
 - 32位处理器最多可用**4GB**的内存容量
 - 64位处理器的最高内存容量理论上为**17,179,869,184GB**，目前大多数64位系统实际可以支持**64GB**的内存。
 - 2个、4个、8个、16个甚至是128个处理器的并行计算机成本也越来越低。
 - 因此内存容量完全可以充分容纳得下当前绝大多数**OLAP**应用的基础数据。



内存OLAP的优势

- ❖ **Materialized view VS. compute on demand**
- ❖ 采用适当的内存结构，内存**OLAP**可以很好地解决了磁盘**OLAP**中**cube**存储的稀疏数据问题。
- ❖ 基于磁盘的**OLAP**服务器受制于性能瓶颈，难于支持实时数据的分析、难于实现**What-if**分析，难于支持维成员更新，而内存**OLAP**可以对此提供支持



内存OLAP的优势

❖ 内存OLAP所带来的好处是

- 可以以更低的成本完成与以前相同的工作;
- 可以更快更好的完成与以前相同的工作;
- 可以完成那些以前在技术上或经济上不可行的工作。



内存OLAP VS.传统磁盘OLAP运行于大内存环境

❖ 存储结构

- 磁盘OLAP的缓冲区所采用的数据存储结构是面向磁盘的数据结构
- 内存OLAP采用的是完全不同的数据存储结构。

❖ 对分析请求的优化

- 面向磁盘
- 面向内存



内存OLAP的关键技术

- ❖ 内存OLAP的体系结构研究
- ❖ 多维数据的内存组织与压缩
- ❖ 基于内存的多维查询处理与优化技术
- ❖ 内存OLAP系统的研制



典型系统

❖ 内存OLAP服务器

- ORACLE的Exalytics/ExaData
- SAP的HANA
- Applix TM1 (被IBM收购, 现为IBM Cognos TM1)
- QlikTech的QlikView BI
- UC Berkeley的SPARK SQL

❖ 中间件

- SAP的BI Accelerator



新的研究方向

❖ 传统问题

- 实体化视图的增量维护
- 数据集成
-

❖ 新的多维数据分析方法

- SKYLINE
- TOP-K
- KNN

❖ 新的硬件环境

- 内存OLAP
- 多核OLAP
- 基于协处理器的OLAP
- 实时数据仓库

❖ 新的应用场景

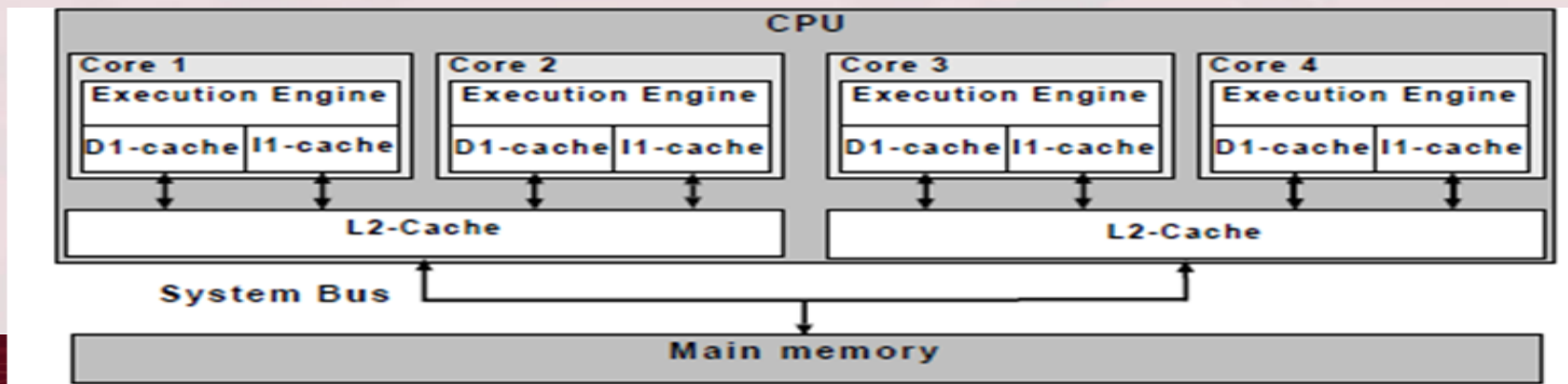
- 大数据OLAP
- 流数据的联机分析
- 物联网中的联机分析



多核处理器设备

❖ 单芯片多处理器 (Chip Multi-Processor, 简称CMP)

- 在一个处理器上集成多个计算引擎核心，能够提供强大的共享内存的并行资源。
- 不仅共享片外资源，而且共享片上资源 (如最后一级cache, LLC)。
- 每个核心都是一个简单的单线程处理器或者多线程处理器，各个核心可以并行执行程序代码，实现程序的多线程并行。
- 所有核心通过高速总线紧密相连。



多核OLAP技术

- ❖ 数据级并行(Data Level Parallel, DLP)
- ❖ 线程级并行(Thread Level Parallel, TLP)
- ❖ 面向内存访问的优化



数据级并行

- ❖ **CMP**的每个核心都有若干个单指令多数据流 (**Single Instruction Multiple Data, SIMD**) 单元，它可以将多个数据打包到**SIMD lane**上，对这些数据同时执行同一个操作，因此**SIMD**又称为数据级并行。
- ❖ 基于多核结构，设计并行的数据库操作算法和数据结构，从而提高原有操作的效率
 - 要高效地利用**SIMD**指令，需要将并行执行的数据连续存储。
 - 利用**SIMD**指令集提供的运算指令以及优化内存中连续数据块传输指令，以**SIMD**方式实现某些操作，达成数据级并行。



线程级并行

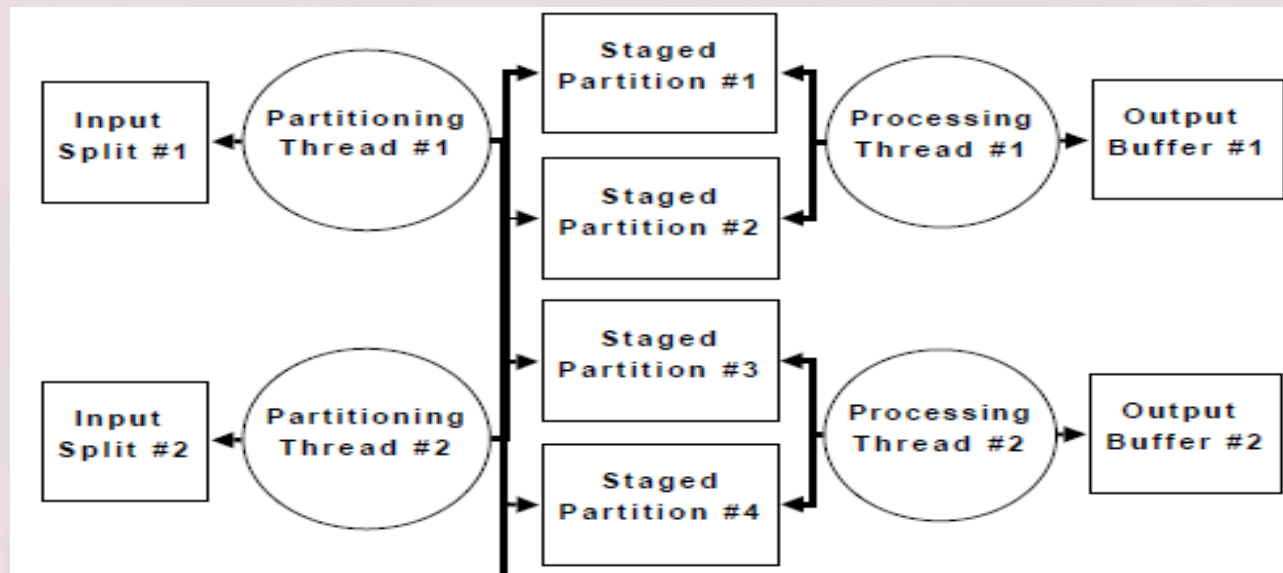
- ❖ 线程级并行：多个线程可以同时运行在不同的核心上，独立完成划分后的运行任务。
- ❖ 线程级别的并行是多核处理器上最为重要的并行，也是算法性能提升的关键。
- ❖ 线程并行有任务划分和数据划分两种方式。
 - 任务划分是指每个核心处理一个子任务
 - 数据划分是指每个线程负责处理一个数据分块。
 - 查询内并行和多查询并行都可以使用上述两种方式实现。



线程级并行

❖ 1. 利用数据划分实现线程级并行

- 为了避免共享资源的竞争和冲突，进行数据划分时应该尽可能使多线程各自处理独立而不相交的数据集。



线程级并行

❖ 2. cache-sharing-aware综合优化

- 设计精巧的数据结构
减少**cache**冲突和**cache**竞争。
- 确定可以并行执行的查询
减少**cache**污染问题。
- 对硬件**cache**划分方法的选择
cache块和线程相映射。
- 利用原语和锁机制实现共享资源的冲突访问
在性能和正确性间取得平衡。



线程级并行

❖ 3. 优化任务调度策略，优化访存效率和吞吐量

■ 提高程序并行程度

- 多请求执行
- 流水线划分

■ 确保线程之间负载均衡

- 多粒度自适应任务调度
- 线程协作调度
- 共生线程调度



面向内存访问的优化

❖ 造成内存访问延迟的主要因素

■ cache命中率

cache缓存CPU从主存请求的数据

■ TLB命中率

TLB缓存虚拟地址到物理地址的转换页表

■ 带宽

带宽代表了CPU与内存之间的数据传输能力

❖ 解决手段

■ 通过提高OLAP操作的局部性，来提高缓存命中率

■ 考虑多级访存的容量、带宽存在差异，针对访存进行设计以提高带宽利用率。



面向内存访问的优化

❖ 1. 局部性优化

■ 时间局部性

- 本次访问的数据在不久的将来可能再次被访问
- 好的时间局部性可以提高**cache**和**TLB**命中率

■ 空间局部性

- 本次访问的数据的附近的数据也将被访问
- 好的空间局部性也可以大量减少主存访问延迟



面向内存访问的优化

❖ 1. 局部性优化

■ 适应硬件特性

形成SIMD宽度、cache线、主存页(磁盘块)多种层次的分块机制，在各个层面上都达到较优性能，提高局部性。

■ 数据预取

- 数据预取将主存的数据读入cache, 而在这期间CPU不需等待，可以继续下一条指令的执行。
- 数据预取包括软件和硬件预取两种方法。



面向内存访问的优化

❖ 2. 带宽优化

- ❖ 设计更小的数据结构和压缩技术
压缩问题空间，降低空间需求。
- ❖ 即时计算代替数据存储
避免大规模的数据传输。
- ❖ Cache的缓冲区机制
降低可用的带宽和对带宽需求的差距。
- ❖ 多线程间共享扫描
减少cache和内存之间的数据交换。



小结

❖ 内存OLAP

- 内存OLAP的概念
- 内存OLAP的可行性
- 内存OLAP的关键技术

❖ 多核OLAP

- 数据级并行
- 线程级并行
- 面向内存访问的优化



