

9-13

并行执行

数据查询

流机制

并行数据库

计算机科学 2000V.11. 27No.1

基于流机制的并行查询执行技术^{*}

The Parallel Query Executor Based on the Stream Mechanism

陈红 文继荣 王珊

(中国人民大学数据与知识工程研究所 北京 100872)

(中国科学院计算技术研究所 北京 100080)

TP311.13

Abstract There exists inaccuracy in estimating the system parameters during the query optimization. Query plan can also be influenced by the environment noises during its execution. These factors dramatically decrease the performance. The parallel query executor of PBASE/2, which is based on the stream mechanism, can perfect the query plan statically and dynamically by the effective strategies of schedule and coordination to reach the overall balance of the workload.

Keywords Parallel query execution, Stream mechanism, Schedule

1 引言

在并行数据库的研究中,查询执行计划的调度与执行因其复杂性而受到人们的关注。查询优化时,优化器必须采用有效策略大幅度裁剪搜索空间,以降低优化开销,但这很可能会丧失掉更优的执行计划。另一方面,当系统吞吐量很高时,一个查询从优化到执行可能有一个较大的时间差,在查询计划执行时一些重要系统参数可能已经发生了较大变化,从而使该执行计划变得不优甚至难于执行。目前解决这一问题的一种方法是将查询优化与查询执行分开^[1],在查询执行阶段通过有效的调度策略来弥补查询执行计划的缺陷,并进一步平衡系统的负载。

目前对并行执行技术的研究强调多种资源、各类负载条件和多用户环境下的综合负载平衡。为操作确定合理的并行度与执行结点是保证负载平衡的一个重要方面。文[2]仅依据各处理结点 CPU 的利用情况来决定连接执行的并行度和执行连接的各个结点。文[3]对此作了改进,综合考虑了 CPU、I/O 和内存三大资源的均衡利用,提出了 Join 并行度的确定同处理结点的分配分开考虑和一起考虑两种策略。文[4]提出在确定操作并行度时,应根据父子操作的处理速率,力争其处理速度相当,以减少中间结果的存储和提高处理机的利用率,避免无谓的空闲等待。在确定处理结点的分配方案时,可以有随机分配、轮转分配、按当前各结点

内存利用情况分配、按当前各处理结点 CPU 利用情况分配、按当前结点磁盘 I/O 利用情况分配和与上一操作分配相同结点等几种方法。文[5]提出将 CPU 密集型任务和 I/O 密集型任务搭配执行,文[6]将任务分为大中小三种类型,并限定了同时存在于系统内的任务类型和数量,以此来限制对系统资源的争用。文[7]把系统资源按其特性不同,分为可以抢占的时间共享资源(CPU、磁盘 I/O、网络等)和不可抢占的空间共享资源(内存等),它们对操作并行度的影响是不一样的。

任务迁移是保证负载平衡的另一个重要方面。所谓任务迁移是指在执行过程中若发现处理结点的负载不均,应将重负载结点上的任务迁移到轻负载结点上。任务迁移包括静态迁移和动态迁移两种,它对提高复杂大查询的性能有较大意义,但目前仍处于理论研究阶段,还有许多问题没有解决。

PBASE/2 是针对 SE、SN 的混合式层次结构的并行硬件平台设计的并行数据库管理系统产品原型^[8],成功地实现了静态和动态负载平衡,实现了任务的静态迁移。PBASE/2 采用多线索结构。每一个单处理器结点或 SE 结点上有且仅有一个局部 DBMS 进程,称作服务进程。每个服务进程内部按功能划分成若干线索,其中负责与用户交互的线索类构成了并行用户管理器 PUM;负责查询优化的线索类构成了并行查询优化器 PQO;所有与执行有关的线索类构成了并行查询执行器 PQE。组成服务进程的所有组件协同工作,共

^{*} 本课题得到国家 863 计划和国家自然科学基金资助。陈红 副教授,在职博士生,研究方向为数据库和知识库系统。文继荣 博士,研究方向为数据库和知识库系统。王珊 教授,博士生导师,研究方向为数据库和知识库系统。

同完成用户的查询请求:PUM 直接与用户交互,接收用户的 SQL 语句;PQO 对该语句进行语法分析,生成语法树,然后对语法树进行优化,生成并行查询执行计划;PQE 根据查询执行计划在相应结点上分配各个操作,并协调和调度这些操作的执行。在查询执行完成后,PUM 汇总执行结果,并返回给用户。

文[9]介绍了 PBASE/2 的优化器 PQO,本文将介绍 PBASE/2 的流机制,它是 PBASE/2 查询处理的基本框架,也是整个 PBASE/2 系统的实现基础,并介绍了基于流机制的 PBASE/2 查询执行器 PQE 的基本工作原理,讨论了 PQE 的几个关键技术,最后给出了实验结果。

2 PBASE/2 的查询框架

在关系数据库中,一个用户查询可以分解成通过输入/输出关系联系在一起的多个数据操作。数据操作的输入/输出机制是规范的,尽管操作本身多变,但它相对于输入/输出的接口却是固定的。在并行关系数据库系统中,输入/输出机制更为重要,它是查询并行化的基础,主要的并行化方法均依赖于输入/输出机制的控制和协调。例如对于以流水线方式并行执行的操作,不论它们是由同一结点的不同线索执行,还是在不同结点上执行,输入/输出机制必须负责数据在不同线索间的高效缓冲和有效传输。对于操作内并行的操作,输入/输出机制还必须确定数据的重分布方法以及传送的目标结点。为此,PBASE/2 提出针对操作间数据和消息的输入/输出关系而建立的一套定义和管理机制——“流机制”,它封装了串行、流水线并行性和操作内并行性,构成了数据库查询的基本框架。

形式化地说,流机制 $Stream = \{AttrDef, FlowMan\}$ 由静态的数据流属性和动态的数据传输管理机制两部分构成。

$AttrDef = \{ \text{数据格式, 数据源, 选择谓词, 投影列, 投影列索引, 数据传输模式, 操作的算法, 操作间是否并行, 读操作并行度, 写操作并行度, 读操作结点号, 缓冲区号, 缓冲区大小, 填充因子, 数据分布属性} \}$

$AttrDef$ 是对相关操作间输入/输出数据一系列属性的定义,反映了流的静态特性。这些属性可以分为三类。前 5 个属性是针对流数据本身的描述特性,第 6 和 7 属性描述与该流相关操作的执行特性,其余属性描述该流相关操作的并行处理特性。

$FlowMan = \{Open, Read, Write, Close, Reopen, \dots\}$

$FlowMan$ 反映了流的动态特性,它是基于流属性之上的一套数据传输管理机制,包括流的 $Open$ 、 $Read$ 、 $Write$ 、 $Close$ 、 $Reopen$ 等操作原语,其主要作用是对操

作间的数据流动进行总监控,利用 $AttrDef$ 的信息来正确管理操作间的数据传输,包括缓冲并按正确的传输模式传送操作间的输入/输出数据,同步对同一流的多个读写操作,对写入的数据进行重分片等。

PBASE/2 查询处理的各个阶段实际上都是与流机制打交道(如图 1 所示)。其中,PQO 负责在查询优化的各个阶段中依据优化结果填写和修正流的属性,这些属性表征了执行计划的特征。PQE 依据流的静态属性,调用管理机制调度和执行任务。一个任务在执行过程中,无论是同一结点上操作之间的数据交换,还是不同结点上操作之间的数据交换,都通过流机制属下的数据缓冲区来完成。流机制构成了各个结点之间以及结点内的各个线索之间的联结纽带,它有效地封装了线索间复杂的数据通信问题。

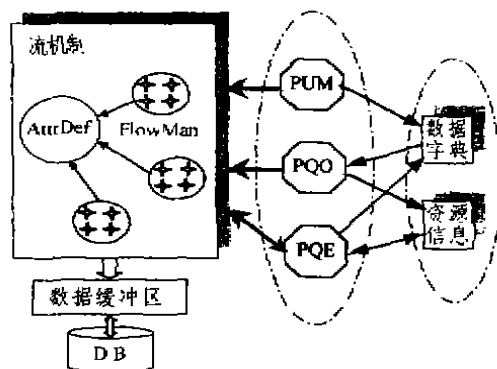


图 1 PBASE/2 查询框

3 PQE 的基本工作原理

PBASE/2 的查询执行器 PQE 负责调度和执行由 PQO 生成的并行查询执行计划。在执行查询计划过程中,通过有效的调度和协调策略,PQE 能够静态和动态地校正执行计划的缺陷,平衡系统的负载。

PQE 是 DBMS 服务进程中最复杂的一个组件,由诸多类别的线索组成,包括负责查询执行的操作线索和通信线索,负责调度的任务分配线索和任务协调线索,负责全局信息管理的全局死锁检测线索、全局数据字典管理线索和全局资源管理线索,负责局部资源管

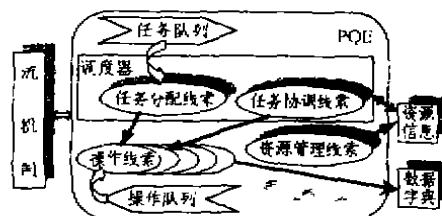


图 2 并行查询执行器

理的数据缓冲区管理线索、日志缓冲区管理线索和局部死锁监控线索等,如图2所示。

具体地讲,PQE的主要功能有两个:一是合理安排查询执行计划中各操作的执行顺序,把它们分布到相应结点上执行,并通过流机制有效地控制各种操作之间的数据流和信号传递,使查询在分布式环境下有序地进行;二是静态地和动态地调整执行计划,使其更适合系统当前的实际负载,减小优化误差。PQE执行一个查询计划的过程可以大致分为4步:

(1) 挑选可执行的任务

PBASE/2以任务为调度、执行和资源分配的基本单位。PQO在生成查询执行计划树后,会将其进一步分割为任务。所谓任务,就是执行计划的一棵子树,它由多个操作组成。划分为任务后,执行计划就变成由多个任务组成的一个有向依赖图。

将计划划分为任务主要出于三方面考虑。一是计划中某些操作间呈阻塞关系,即有的父操作必须在子操作执行完后才能开始执行,无法和子操作并行执行,因此计划中本来就存在自然的断点。二是有些执行计划太大,占用资源太多,无法作为一个整体同时运行,将计划划分为任务后可以减少对资源的占用。三是将计划划分为任务后能够增加任务调度的灵活性,例如可以将一个计划中的CPU-Bound型任务与另一个计划中的I/O-Bound型任务并行执行,从而提高了系统资源的利用率,改进了系统的总体性能。

任务的粒度并没有一个统一的标准,它可以粗到整个计划,也可以细到单个操作。任务的粒度越细就越容易做到负载平衡,但系统开销也越大。这是因为流水线上的父子操作间存在依赖关系,如果它们属于同一任务,会同时分配资源等待执行,这时父操作的执行由子操作产生数据来驱动;而如果它们属于不同任务,为维持父子操作的偏序关系,即保证子操作在父操作之前被调度执行,需要付出额外开销。因此,划分任务时必须依据父子操作的阻塞关系和系统可用资源状况进行权衡,选择合理的粒度。

执行计划被划分为任务后,排入PQE的任务队列中。当任一结点的操作队列长度小于指定值时,会给PQE调度器中的任务分配线索发消息,请求追加新操作。这时任务分配线索会依据每个计划中各任务间的偏序关系,按照一定策略从任务队列中挑选满足执行条件的任务。

(2) 检查和调整执行参数

任务的各种执行参数已由PQO填写在流属性中,当吞吐量极大或所有用户的查询都极其复杂时,从优化到执行会有一定的时间间隔,这时一些重要的系统资源参数可能已经发生较大变化,从而使计划变得

不太优。因此,任务分配线索在挑选出满足执行条件的任务后,还需要依据系统资源的最新状态检查和调整执行参数,然后才能将组成任务的各操作分别排入指定的操作队列,等待操作线索执行。

PBASE/2之所以不在优化后直接把组成任务的各操作分配到各处理结点,排入操作队列,而是先将任务排入任务队列,就是为了减少执行前系统资源参数变化对任务的干扰,即在不影响执行的条件下,尽可能推迟操作的分配,为调整计划提供时机。一旦把任务分解成操作排入操作队列,再调整起来代价会相当大。

任务分配线索需要检查和调整的执行计划参数主要包括并行度、处理结点、缓冲区大小等。决定是否调整执行计划的基本原则是,尽量使系统的整体工作负载达到基本平衡,同时尽量使一个查询在各个结点上的响应时间均衡,避免因其中某一个操作执行过慢而响应整个查询的响应时间。为此必须综合考虑各个处理结点的资源信息,将对某种资源需求量的操作分配给该种资源负载较轻的结点,或者适当增加资源需要量大的操作的并行度,以减少对每个结点的资源压力。具体调整时有如下启发式规则:

规则1 如果任务A是包含执行计划叶结点的子树,则它必然是可以最早执行的任务之一,为它调整参数比较自由,可以仅依据系统当前资源信息进行。

规则2 如果任务A不包含执行计划的叶结点,则必有 m 个($m \geq 1$)在其前面执行的任务 B_i ($1 \leq i \leq m$),任务A需要以其中 n 个($n \leq m$) B_j ($1 \leq j \leq m$)的输出数据作为输入。由于这些 B_i 的输出数据传送到哪些结点已经确定,而且此时可能已经开始执行甚至已经执行完毕,因此调整A的执行参数不应影响这些 B_i ,否则开销太大,换句话说,在这种情况下不能调整A的叶结点的并行度和处理结点。

规则3 执行计划的叶结点通常是Scan操作,只可能在存储相应数据的结点上执行,因此在调整并行度和执行结点两属性时不必考虑它们。

规则4 设任务A由操作 O_i ($1 \leq i \leq n$)组成,并具有偏序关系(用符号 $\ll$ 表示): $O_i \ll O_j$ ($i < j$),则 O_i 操作所在结点的负载应小于等于 O_j 操作所在结点的负载。即应优先把 O_i 分配到小负载处理结点上,尽可能避免因 O_i 比 O_j 先执行或 O_i 执行过快,因而不得不等待。

规则5 设任务A由操作 O_i ($1 \leq i \leq n$)组成,并具有偏序关系: $O_i \ll O_j$ ($i < j$),调整 O_i 的执行地点时,必须同时调整 O_i 输出数据的分布策略和分布结点。

规则6 如果因对系统实际资源的估算有误差,任务A的资源需求目前无法满足,则将A截断为 A_1

和 A_2 两个子任务 ($A_1 \leq A_2$), 并将 A_2 放回任务队列。

检查和调整执行计划是要付出一定代价的。对于普通 OLTP 应用, 这种调整可能得不偿失, 因此 PBASE/2 设置了一个开关, 以简单查询为主的应用可由 DBA 关闭该开关。

(3) 分配任务

执行参数确定下来后, 任务分配线索就可以依据流属性将该任务的各个操作分配到指定结点, 排入操作队列, 等待执行。

(4) 执行

执行开始时操作线索首先依据流属性, 调用流的管理机制初始化运行环境, 执行结束后还要清理环境。任务的整个执行过程由数据驱动, 除开始的 Scan 操作外, 所有操作的处理行为都是因为接收到其它操作的输出数据而触发的。操作线索间的数据传送通过流机制进行管理和控制。流机制提供了一套标准的流操纵接口, 包括流的 Open、Read、Write、Close、Reopen、Lseek 等。流机制使得数据可以在相同结点和不同结点上的线索之间高效地传输。这种底层的异地数据传输对于操作来说是完全透明的。

执行过程中, PQE 的任务协调线索会对任务中各个操作的执行进行监控和协调。一旦发现操作间不和谐, 任务协调线索会动态调整计划, 如增加或减少缓冲区尺寸, 改变操作线索的优先级等。

4 PQE 的关键技术

4.1 PQE 对并行性的支持

PQE 在调度任务执行时, 可以以多种方式运行组成该任务的各操作, 实现了操作间流水线并行、操作间独立并行、操作内并行等多种粒度的并行性。PQE 为不同的执行方式提供了不同的协调策略。

不同操作独立并行执行时, 因操作之间无数据交换关系, 实现起来很简单。不同操作以流水线方式并行执行时, 父子操作通过数据驱动, PQE 采用基于页面的封锁机制实现操作线索之间异步读写缓冲区, 同时采用了有效策略匹配父子操作的速率。操作内的并行执行是通过数据划分实现的, 目前 PBASE/2 支持 Round-Robin、Range、Hash、One-to-one 和 One-to-all 五种数据分布方法。

当操作内并行执行而操作间串行执行时, 父子操作的并行度相同, 为减少数据传输, PQE 通常将父子操作分配在同一结点, 使子操作输出的数据不需要重新划分, 直接存放在本结点的缓冲区中, 供父操作读取。

当操作内并行执行同时操作间也并行执行时, 父子操作均分别在多个不同的结点上执行, 父子操作的

并行度可以不同, 这时各子操作输出的数据需要重新划分到父操作所在结点。

4.2 PQE 的数据通信机制

当任务以操作内并行且操作间并行方式执行时, 子操作的输出往往需要重新分布, 发送到异地缓冲区。PQE 中设有专门负责结点间通信的后台线索, 它监控一个网络接口, 负责本接口数据的发送和接收。当需要进行数据异地传输时, 后台通信线索会在本地为异地缓冲区生成一个影子缓冲区, 负责缓存本地发往异地的记录, 将其打包后发送给目标结点。目标结点的后台通信线索收到数据包后, 根据包头信息将其定位到相应缓冲区, 然后将包中数据写入该缓冲区。可以看出, 在这种机制下, 网络数据传输对操作是完全透明的。

4.3 基于流的缓冲区管理策略

在 PBASE/2 中, 缓冲区负责主存与辅存之间的数据缓冲, 同时为流机制中传输的数据提供存储空间。缓冲区中的数据主要有三种来源: 一是直接从基表中读取数据; 二是从临时表中读取数据, 临时表可能是子表达式或视图的处理结果; 三是来自其它操作的输出, 这是最常见也最重要的情况。无论是哪种情况, 其接口对于从流中读取数据的操作而言都是一样的。

缓冲区管理策略和页面淘汰算法是影响数据库系统性能的重要因素之一。目前商用数据库系统通常都是象操作系统一样采用随机模型来刻画数据存取行为, 因此一般采用 LRU、MRU 等简单的策略来管理缓冲区。但实际上, 与操作系统缓冲区不同, 数据库缓冲区的数据存取行为特性是可以预测的。为此, PQE 根据页面存取模式的不同采用了不同的缓冲策略。

PQE 采用一种页面级封锁来保证流水线上父子操作读写缓冲区的同步和互斥, 即当读、写数据位于不同的页面时, 读写可以真正并行地执行, 这样做可以减少额外开销。另外, 缓冲区页面可以依据父子操作的速率动态追加或减少。

当父子操作分属不同任务时, 它们之间呈阻塞关系, 即子操作全部执行完毕后父操作才能开始执行, 这时必须保存子操作的全部数据。有时中间结果的大小是很难精确估计的, 因此到底为其分配多大的缓冲区就成为一个难题。有些系统为简单起见, 只为所有的阻塞型中间结果分配一块很小的缓冲区, 超过缓冲区的数数据都被写入磁盘临时表中。这种方法的效率显然很低, 因为很多不是太大的中间结果本可以直接放在内存中。PQE 的缓冲策略是: 假设优化时估计的缓冲区大小是 P 页, 如果实际运行时缓冲池空闲页面数 $< P$, 则为其初始分配一个最小页面数 $\text{Min}P$ 页; 如果 $P < =$ 系统预设的临界值 μ , 则为其分配 μ 页。当写入的数据超过分配的页面数时, PQE 查看缓冲池中是否有空

闲页面,如果有,则为该虚表追加新的缓冲页面;如果没有空闲页面或者该中间结果总的缓冲页面数>系统预设的最大页面数 MaxP,则为其在磁盘上生成一个临时表,后续的数据将被写入到临时表中。这种缓冲策略能够在中间结果估计不精确的情况下,尽量减少磁盘 I/O 的数量,同时也防止了为一个中间结果分配太多缓冲页面而影响其它操作的读写效率,从而兼顾了缓冲池的整体使用效率。

4.4 流水线父子操作的协调

流水线上父子操作的执行是靠缓冲区中的数据来协调;当缓冲区已写满时,子操作挂起等待,当缓冲区中没有所需数据时,父操作挂起等待,但如果父子操作的执行速度相差悬殊时,会影响整个任务的总体执行,同时也会浪费系统资源。PQO 在生成执行计划时已经充分考虑了这一点,但由于估算的误差以及实际执行时的各种环境噪音,仍可能出现速率严重不匹配。PQE 采用了执行前预防和执行中补救等多种手段来解决或缓解这一问题。

执行前,任务分配线索在依据当前系统资源状况检查和调整执行参数时,如果发现父子操作速率之差超过阈值,则重新调整父子操作的并行度和结点分配。例如降低(增加)执行速度快(慢)的操作的并行度,或者将执行快(慢)的操作分配到资源相对较少(多)的结点上等。

执行过程中,一旦发生了父子操作的速率不匹配问题,PQE 提供了多种补救措施。

为减少父子操作速率不匹配而导致父或子操作线索频繁挂起从而增加线索的切换开销,流属性中设了一个填充因子属性,它规定一个阈值(α, β),只有当缓冲区中数据的充满度大于 α 时才唤醒父操作线索,当缓冲区中数据的充满度小于 β 时才唤醒子操作线索。

同时任务协调线索还会调用流的管理机制,增加(减少)父子操作间缓冲区的尺寸。如果子操作执行过快,增大缓冲区可以使子操作不必挂起等待。如果父操作执行过快,减小缓冲区可以节省宝贵的内存资源。如果这种速率不匹配不是十分严重,这时就能基本解决问题。

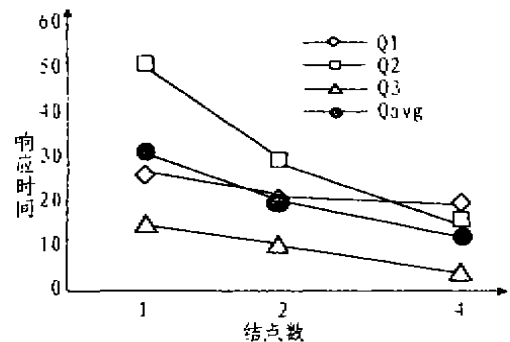
当父子进程速率严重不匹配时,任务协调线索将提高(降低)一个线索的优先级,强制其执行速度变快(慢)。

5 实验结果

测试 PBASE/2 查询执行技术采用的硬件环境分别是:(1)由带宽为 100MB/S 的局域网连接四台 PC 工作站组成 SN 集群系统。(2)由带宽为 100MB/S 的局域网连接两台 PC 工作站组成 SN 集群系统。(3)一

台 PC 工作站。

图 3 给出了 3 个典型查询的系统加速比。其中 Q1 是一个单表查询,Q2 和 Q3 都是多表连接查询。tb1 表 10 万条元组,tb2 表 800 条元组,tb3 表 5000 条元组。从图 3 可以看到,当处理结点增加时,系统加速比接近线性。



Q1:select * from tb1 where coll like 'phq%';

Q2:select * from tb2,tb3 where sint4=sint5

Q3:select distinct * from tb2,tb3

where sint4>sint5 and sint4<3300 and sint5>100
order by sint4;

Qavg:平均响应时间

图 3 PBASE/2 性能

参考文献

- 1 Nippl C, Mitschang B. TOPAZ: a Cost-Based, Rule-Driven, Multi-Phase Parallelizer. In: Proc. of the 24th Intl. Conf. on VLDB, 1998
- 2 Rahm E, Marek R. Analysis of Dynamic Load Balancing Strategies for Parallel Shared Nothing Database Systems. In: Proc. of the 19th Intl. Conf. on VLDB, 1993
- 3 Rahm E, Marek R. Dynamic Multi-Resource Load Balancing in Parallel Database System. In: Proc. of the 21st Intl. Conf. on VLDB, 1995
- 4 Methta M, DeWitt D J. Managing Intra-operator Parallelism in Parallel Database System. In: Proc. of the 21st Intl. Conf. on VLDB, 1995
- 5 Hong W. Parallel Query Processing Using Shared Memory Multiprocessors and Disk Arrays, 1992
- 6 Methta M, DeWitt D J. Dynamic Memory Allocation for Multiple-Query Workloads. In: Proc. of the 19th Intl. Conf. on VLDB, 1993
- 7 Garofalakis M N, Ioannidis Y E. Parallel Query Scheduling and Optimization with Time and Space-Shared Resources. In: Proc. of the 23rd Intl. Conf. on VLDB, 1997
- 8 并行数据库系统 PBASE/2 技术研究报告. 中国人民大学内部资料, 1998. 9
- 9 文继荣, 陈红, 王珊. 基于流机制的并行数据库查询优化技术. 计算机学报, 2000(1)