主存数据库系统与技术*

阳国贵 王 升 张火炬 吴泉源

(国防科技大学计算机系,长沙 410073)

摘要 主存数据库管理系统充分利用当前硬件和先进体系结构所提供的物质条件,如大容量主存,大规模并行计算机体系结构、客户/服务器模型以及网络计算机环境,把所管理的数据全部存于物理内存中以获得非常高的存取速度,这使得MMDB技术和系统可以很好地满足某些在线或实时应用场合的需求.传统数据库中的优化措施是针对磁盘存储特性的,MMDB则采用不同的技术来组织数据和保持数据的可靠性.本文将讨论一些主要的技术以及这些技术在某些已设计或实现的系统中的应用.

关键词 主存数据库,数据库管理系统,存取方法,查询处理,恢复技术.

随着数据库应用领域的扩大,对于数据库技术提出了新的和更高的要求,从数据类型的扩充(对于音、形、图的应用要求)到数据模型的扩充(对规则、数据、对象一体化的要求),从库容量的增大(工程数据库、公共数据库应用的要求),到响应时间的缩减,等等.在一些应用中,对于响应时间有较高的要求,如电话号码自动查询,多媒体数据库中的实时和同步等.近年来,随着VLSI技术的发展,存储器单位容量的价格不断下降,预计2000年时,单机上的数据库管理系统可获得256Gb的工作内存,而MPP(Massive Parallel Processor)机器将有更大的工作主存.硬件的发展,将带来I/O、CPU、内存三大瓶颈的突破.由此,对于那些实时性要求高的应用,人们开始考虑将整个数据库或其中的大部分数据置于主存中,从而产生了MMDB(Main Memory DB)的概念.目前,包括实验原型和设计蓝图在内,已出现了数十种MMDB系统.

在传统数据库管理系统中,其数据驻留在磁盘上,(Disk Resident DB,简写为DRDB),磁盘上的主拷贝可以按需要遵循一定的替换策略调入主存缓冲区,用于存取操作;在主存数据库系统中,数据长期驻留于主存,作为主拷贝,而磁盘上的备份只用于转存和恢复.当然,在以上两种情形下,一个给定的对象可以同时存在于主存和磁盘两处,但两者具有根本性差别:在MMDB中,主存中的数据是作为主拷贝的,这也就意味着在对数据的组织和存取上,两者有着根本的不同.这可从将讨论的诸多技术方面看出.这里,先对MMDB存在的一些

* 本文1993年10月25日收到,1994年1月3日定稿

作者**阳国贵**,30岁,讲师,主要研究领域为数据库与知识处理.**王升**,26岁,硕士,主要研究领域为数据库与专家系统.**张火炬**,女,26岁,助教,主要研究领域为人工智能.**吴泉源**,52岁,教授,博士导师,主要研究领域为智能软件与智能应用.

本文通讯联系人:阳国贵,长沙 410073,国防科技大学计算机系

疑问做简要说明:

1. 整个数据库放入主存是合理的吗?

对于某些应用来讲,这是合理的,也是必要的.在某些情形下,数据库的数据量极为有限或者数据量相对于主存容量增长而言增长速度较慢.例如,公司人事数据库的大小正比于职员的人数,不管该公司如何庞大,将其放入主存应当是合理的.在某些实时应用中,数据必须放入主存中以满足实时处理对时间的要求,当然,也有一些情况,数据库显然不能全放入主存中,如卫星图像数据.MMDB 的批评者认为:主存量上升的速度无法赶上发展的应用需求.但是,即使如此,也可能把数据中那些使用频率较高,通常容量不大,处理时间要求较高的那部分数据分离出来构造一逻辑数据库,由 MMDB 管理.

IMS,是最早的数据库系统之一,已经意识到这些频度差异,并在系统中提供两套子系统,Fast Path 用于主存数据库管理,常规的 IMS 管理其他数据库.目前也有一些文章讨论多层数据库系统和数据转移等内容.

2. MMDB 与具有大缓存的 DRDB 的差异是什么?

如果 DRDB 的缓存足够大,数据可常驻主存,即使该系统性能较好,也还没能完全利用主存的长处.例如,它的索引结构还是针对于磁盘存取的(如 B—树),尽管其数据全部在内存,可数据的存取仍然不得不经过缓冲区管理器,因为它仍假定数据存于磁盘之上.例如,当某一应用希望存取一给定元组的时候,它的磁盘地址将每次重新计算,需由缓存管理器来检测相应的块是否在内存,当对应块一旦找到,如元组要被用到,则将其拷贝到这一应用的元组缓存区.显然,假如记录总是处于主存之中,用其在内存的地址来引用它更为有效.Tobin J. Lehman 等人在对其实现的 Starburst 系统[1]进行性能测试时,专门对此做了工作.测试中,一个子系统采用主存数据库管理方式,另一个子系统是采用传统 DRDB 方式.为了准确起见,后者的数据完全置于缓存,且两者都不进行 I/O 操作.实验结果表明,前者速度比后者快 2.3 倍到 7.1 倍(对不同应用例子).

我们这里列出的仅是内存组织优化的一种情况,其余的将在下面讨论.值得注意的是,一些 DRDB 和面向对象的存储系统(OOSS)也注意到具有大容量 cache 的时候,其中一些数据将常驻主存,进而实现 MMDB 中内存组织和管理的一些优化.例如,一些新的系统把元组或对象转换成一种内存表示形式并将为应用程序一直保留这一指针.今后,我们希望 MMDB 和 DRDB 的这种差异将消除,任何好的数据库管理系统都将会意识到有些数据将长驻主存这一特点并进而采用相应的管理策略.

3. 我们能否假定采用特定硬件后主存是非易逝的和可靠的呢?

我们试图做这样一个假设,从而使 MMDB 设计简单、性能进一步改善.对此,恐怕不存在肯定或否定的回答.主存只是一种存储介质,可以用诸如后备电池存储卡,非中断电源系统,错误检测和纠正存储技术等使得它更为可靠,但是,这只是减少介质故障的可能性,而不能使之降低到零.因而我们将还是有必要做数据库的备份,如在磁盘上,这对 DRDB 也是需要的.

当然,随介质故障率的降低做备份的频度可以降低,另外,频度还受如下因素影响.

(1)主存由处理器直接存取,它更容易遭到 OS 和应用软件错误造成的破坏,也就是说,即使硬件很可靠但主存的内容将由于系统的崩溃而不时地丢失.

(2)在传统技术中,一个磁盘的故障不影响其它盘片.恢复之时,只需将数据库在故障磁盘上的一部分从备份中复原,数据库的其余部分仍是可用的.而MMDB一旦出现故障,通常必须重新开机,或重启系统,整个数据库都将丢失,正因为数据的恢复极为耗时,保存一份较新的备份是必要的.(备份越旧,需要从log中取的内容越多).

(3)后备电池内存或不中断电源(uninterrupt power suppplies(UPS))这类设备丢失数据的可能性比磁盘要高,因为UPS可能过热、电池可能泄漏或漏电.

总之,除非对内存可靠性有足够的信任,内存备份将不得不采取相对较频繁的策略.备份设施的性能将具有重要决定作用,同时,由于写盘的代价比内存高得多.这样,与DRDB时相比,备份的相对代价就要高得多.

1 主存数据库管理系统技术

1.1 环境特性

计算机的主存和磁盘显然具有不同的特性[2],正是这些特性影响着数据库系统的设计和性能.

· 主存和磁盘在存取时间上有数量级的差别.

· 主存通常是易逝的,而磁盘却是永久性存储.当然,也可以一定的代价构造非易逝(nonvolatile)主存.

· 磁盘是块存储设备,而主存是以字或字节编码的.

· 数据在磁盘中的存储方式(结构)对性能的影响远比主存敏感,对磁盘而言,顺序存取远快于随机存取.

· 主存通常可由处理器直接存取,而磁盘则不然,因而主存数据比磁盘数据更易受到由于软件错误所导致的数据破坏.

· 多机系统,Client/Server,网络计算机等环境进一步发展.

这些差异对于数据库管理的几乎每个方面都有影响,从并发控制到应用界面,进而形成了MMDB的技术特点[3].

1.2 并发控制

由于对主存的存取比磁盘快得多,在MMDB中,事务运行速度将较快,在一些使用基于锁机制的并发控制系统中,锁的占有时间较短,这意味着锁定的内容没有DRDB中那么重要.

采用较小封锁粒度(域或记录)的系统减少锁定的范围,这种策略由于数据常驻内存时锁定范围就已经较小,因而使得细精度锁的基本优点就丧失了,所以MMDB中一般都建议采用大粒度锁策略,如关系.

在极端情况下,锁粒度可选定为整个数据库,这就等于事务的串行执行,事务的串行执行也是很可取的,因为并发控制的代价(置锁、解锁、处理死锁)等几乎完全不存在了.进而,CPU缓存交换的次数大为减少.每当一个事务挂起等待锁时,一个新事务将被启动,CPU缓存中的内容将被切换.在高性能计算机系统中,CPU缓存交换相当于成千条指令,因而,由此而获得的效益将是明显的.当然,当事务较长时,串行执行是不现实的,我们应设法实现

短事务与长事务的并行执行,进一步讲,在多处理器环境下,即使所有事务都较短,我们也还是需要某种形式的并发控制策略.

在 MMDB 环境下,对于锁机制的实现也可以进一步优化,在传统系统中,锁用 hash 表来实现,hash 表中登录当前锁定的对象,这些对象(位于磁盘)本身不包含任何锁信息.若对象位于内存,我们可以让其本身包含某种锁信息.

1.3　提交处理

为了在介质故障时受到保护,有必要对数据库做一备份并用日志登录它的活动.由于内存通常是易逝的,日志应当存于一稳妥的介质上,在一事务提交前,事务的活动情况必须写入日志.对日志操作将会在某种程度上影响 MMDB 的性能,这种日志在成为瓶颈时还影响事务的吞吐量.当然,这些问题在 DRDB 时也存在,但在 MMDB 更为突出,它是事务对磁盘操作的唯一根源.

对这一问题的解决办法可有几种.一种是构造一部分非易逝的主存,用于部分日志的保存.提交时,只需把日志写入这一主存中,一特定的进程或处理器用于把该主存的数据移到日志磁盘.第二种策略是提前提交,在日志记录写入日志时就释放锁,这样,而不等到日志信息全部写入磁盘之后.日志的这种顺序特性保证了一个依赖于其他事务的事务不可能比被依赖事务提前提交,它不减少事务响应时间,但减少阻塞延时,即减少并发事务中其它事务响应时间.

一种称为成组提交的技术可用于解决日志瓶颈问题.在成组提交情况下,一事务的日志记录并不要求在它提交时马上写入磁盘,而是在内存中积累到一定程度时(如一页满),用一磁盘操作把它写入到日志磁盘,这样成组提交减少了磁盘操作次数.

1.4　存取方法与数据组织

在 MMDB 中,数据组织结构需要做些变化,人们已提出了一些新结构,它包括多种形式的 hash 技术和树结构,如 T－Tree 结构[1]就是为 MMDB 设计的.主存树结构与适用于磁盘结构的 B 树不同,它不一定要求路径短,这主要由于内存中的树遍历速度较快,另一方面,在做索引结构时,由于内存随机查找同样高速,可以多使用一些指针结构,使索引项与索引数据本身分离,避免索引数据的变长特性所造成的索引组织的困难.在数据组织时,变长数据存于 heap 堆中,用规范结构中的指针指示.在 DRDB 中,那些一起存取的数据对象(如元组域)通常存储在一起,或称为聚集.例如,对一部门及其部门内所有职员的查询,这时希望所有职员记录能和它所在工作的部门记录位于同一磁盘页中.而在 MMDB 中,就不一定要对对象进行聚集,事实上,对象的成份可以散布于内存中,如元组仅保存指向数据部分的指针,而数据存于其他位置.这里还有一个在 DRDB 中不曾有过的问题,当一个对象迁往磁盘时,它是如何存储以及存储于何处?对此,有不同的解决办法,如可让用户指明在迁移时对象如何聚集,或完全让系统决定存储模式对之进行自动聚集.这里主要想说明的是对于聚集与迁移问题,MMDB 与 DRDB 所遇到的并不相同.

1.5　查询处理

由于主存中顺序存取与随机存取在时间上的差别与磁盘大不一样,DRDB 中原有的一些查询优化措施主要是基于这一差异的.在多数传统系统中则试图减少磁盘存取,如排序归并连接技术.而内存驻留数据的查询处理必须着眼于处理代价.当然,在一复杂的数据管理

系统中,对处理代价的估算是较困难的.必须先确定费时的操作(建立索引或拷贝数据),然后设计一些策略以减少其出现的频率,操作代码在不同系统中的不同,可能造成在某一系统中很合适的优化技术在另一系统中则可能表现不佳.

1.6 恢复

恢复由几个不同部分所构成,一个过程用于数据库通常操作的保留,以便后备文件保持最新版本,第二个是用于故障发生后使数据库得到复原的过程.

我们已谈到过提交处理,它使所有提交事务成为可靠.当然,多数使用日志的系统也使用备份或检测点技术以减少故障恢复时日志的处理工作量.在MMDB数据库中,检测点和故障恢复是存取磁盘数据库的唯一理由,应用事务从不需要存取磁盘数据.因而,MMDB中磁盘存取可为检测点的工作而特别设计,如磁盘I/O块应使用超大块因子,因为大块对于读写更为有效,大块占用较长时间,但不是应用程序而是系统去等待这些操作的完成[4].

检测点与事务处理间的干扰应尽量小,一些检测点保存的算法如事务一致(Transaction-consistent)或动作一致(action-consistent)要求它与事务同步,另一不需要同步的可能做法称为增量转存,然而,一致性检测可以通过记录逻辑日志而简化日志内容.

故障后MMDB必须利用磁盘后备和日志恢复数据库,当库较大时,磁盘到内存的数据传输将占用较长的时间,对此问题的一种解决方法可根据数据库的需要逐步将其装入主存,但是,在高性能的数据库系统中,对于数以千计的事务不知怎样逐步装入数据库才算合适,另一种途径是用盘阵列,各盘并行地传递数据,但这要求各盘到主存的路径互不依赖.

1.7 性能

对于主存数据库而言,除了事务提交处理以外,性能主要取决于处理时间而不是磁盘I/O.即使在恢复处理中,它包括对于I/O的磁盘操作,但影响性能处理主要还是处理器,因为磁盘操作通常在事务关键处理以外,这是与DRDB系统中以I/O磁盘操作作为影响算法的主要因素时的情况是不同的.除此之外,MMDB中对于性能影响所必须考虑的部分可能也不一样,如在传统DRDB中,做备份(如检测点)在正常系统操作中不影响它的性能,对此,在性能分析中不对此做深入的研究.但在MMDB中,备份将更为频繁,且包含I/O,所以备份的性能及检测点算法将更为关键,有必要做深入研究.

1.8 应用程序接口和安全保护

在传统DRDB中,应用与数据库系统通过私有缓存交换数据.例如,读一对象,应用程序调用数据库,给出对象的标志id及程序空间内的缓冲区地址,数据库系统从磁盘把对象读入它的缓冲池,然后把对象拷贝到应用程序私有缓冲区.写入时,应用程序修改私有缓存,然后调用数据库系统,系统把修改后的对象读入它的缓冲池,做好相应的日志登录,并进行I/O操作.

在MMDB中,对于对象的存取可更为有效.首先,应用程序可获得对象在主存中的地址位置,而可不用更为一般的对象id,例如,当应用首次引用一元组时,它可使用其关系名和主码,在读操作以后,系统返回元组的内存地址,它就用于以后对于对象的存取,它可避免费时的各种转换,据此,一种优化办法是去掉私有缓存,使事务直接存取对象.在此情况下,性能将获得较大改善,若事务是一简单事务,在原有情况下,各缓存间的数据拷贝将花费大量时间.但这种直接访问可能带来两个问题,当事务可以直接存取数据库时,它们可能读入或

修改一些未经授权的数据部分.系统将没有办法知道哪些部分被修改,因而也不可能对修改部分做日志.一种可行的解决方法可以是:这些事务须由一特定的数据库系统编译器所编译,对于每一数据库对象的存取,该编译器将产生一些代码,以检测正当的授权和对每一对象的修改做日志.

2 现有系统

MMDB 系统有不少,从一些只有蓝图设计的系统,如 MM—DBMS,MARS,HALO,到原型或实验性实现系统,如 PRISMA/DB[5],GKD—MMDB(国防科技大学研制的主存数据库系统),OBE,TDK,SystemM 甚至到商业性系统,如 Fast Path.这里仅给出一些有代表性的系统,以及它们对上面所提到的一些主要技术在这些系统中是如何处理的.

主存数据库系统概览

	并发控制	提交处理	数据表示	存取方法	查询处理	恢　复
MM—DBMS	two-phase locking of relations	stable log tail by segment	self-contained segments, heap per segment, extensive pointer use	hashing, T-trees, pointers to values	merge, nested-loop, joins	segments recovered on demand, recovery processor
MARS	two-phase locking of relations	stable shadow memory, log tail				recovery processor, fuzzy checkpoints
HALO		in hardware, nearly transparent				physical, word-level log
OBE			extensive use of pointers	inverted indexes	nested loop-join, on-the-fly -index creation, optimization focuses on processor costs	
TPK	serial transaction execution	group commit, precommit	arrays			two memory resident databases, fuzzy checkpoints
System M	two-phase locking, minimize concurrency	several alternatives	self—contained segments, heap per segment			various checkpointing, logging options
Fast Path	VERIFY /CHANGE for hot spots	group commit				

3 结论与展望

综上所述,MMDB数据库管理系统的研究已全面展开.随着VLSI技术的发展,主存容量的增大和硬件的进一步廉价,硬件环境将为MMDB的研究提供更有力的支持,对于多处理机并行环境,Client/Server,网络计算机等环境上主存数据库的研究和实现,将更好地满足一些应用对于实时和事务吞吐率的要求.MMDB系统和技术将得到更为普遍的应用,本文所讨论的一些优化技术和措施将逐步为世人所熟悉.

参考文献

1 Tobin J Lehman *et al*. An evaluation of starburst's memory resident storage component. IEEE Trans. on Knowledge and Data Engineerig, December 1992;4(6):555—566.

2 Hector Garcia—Molina, Kenneth Salem. Main memory database systems: an overview. IEEE Trans. on Knowledge and Data Engineering, December 1992;4(6):509—516.

3 Dewitt D J *et al*. Implementation techniques for main memory database systems. In: Proc. ACM SIGMOD Conf., June 1984.

4 Eich M H. A classification and comparison of main memory database recovery techniques. In: Proc. Int. Conf. on Data Engineering, Feb. 1987:332—339.

5 Peter M G Apers *et al*. PRISMA/DB:a parallel, main memory relational DBMS. IEEE Trans. on Knowledge and Data Engineering, December 1992;4(6):541—554.

MAIN MEMORY DATA BASE SYSTEMS AND TECHNIQUES

Yang Guogui, Wang Sheng, Zhang Huoju and Wu Quanyuan

(*Department of Computer Science, National University of Defence Technology, Changsha* 410073)

Abstract Main Memory Data Base Management System makes the best use of current hardware and advanced architecture benifits, eg., the emerging large volume memory, massive parallel computer architecture, Client/Server model and network computer. Storing its data in main physical memory to get very high speed access, MMDB validates its use for some on—line or real time applications. Coventional database systems are optimized for the particular characteristics of disk storage mechanisms. MMDB, on the other hand, use different optimizations to structure and organize data, as well as to make it reliable. The major techniques and their usage in some designed or implemented MMDB systems will be surveyed in this paper.

Key words MMDB, DBMS, access method, query processing, recovery processing.