运行时间自适应的事务调度算法

张云[1] 李岚[2]

（甘肃联合大学 电子信息工程学院, 兰州市 730000）[1]

（兰州理工大学 计算机与通信学院，兰州市 730000）[2]

摘 要：实时数据库中是基于事务的不同优先级进行事务的调度，所以事务优先级的确定对实时数据库的效率有着至关重要的作用。本文讨论了一个运行时间自适应的事务调度算法，对算法思想和调度策略进行了深入的研究。该算法事务的估计运行时间与 CPU 时钟无关，具有自适应的能力，并且可以得到周期性事务的准确运行时间，对于周期性事务比例比较高的应用系统非常适合。

关键字：实时数据库；事务调度；优先级

中图法分类号 TP311.1 文献标识码 A doi:10.3969/j.issn.1003-6970.2011.01.006

Run-time adaptive scheduling algorithm for transaction

ZHANG Yun[1] LI Lan[12]

(Dept. of Computer, Gansu Lianhe University,Lanzhou Gansu 730000 , China)[1]

(Dept. of Computer and Communication, Lanzhou Ligong University, Lanzhou 730000, China)[2]

【**Abstract** 】 Real-time database is transaction-based services with different priorities for scheduling, so the transaction priority in the efficiency of real-time database has a vital role. This article discusses the affairs of a run-time adaptive scheduling algorithm, the algorithm and scheduling strategy thought out in-depth research. The algorithm estimates of transaction and the CPU clock running time has nothing to do with the adaptive ability, and can get the exact running time periodic transaction, for a relatively high proportion of recurring transaction is suitable for applications.

【**Key words**】Real-time database; transaction scheduling; priority

0 引言

传统数据库中，事务以相同优先级运行，而在实时数据库中，是基于事务不同的优先级进行调度。实时数据库采用线程池——Threads Pool 来实现多个事务的并发处理，在事务处理过程中，根据每个线程处理事务花费的时间和事务本身的优先级从线程池中选择最合适的线程对事务进行处理。

1 事务执行模型

线程在系统初始化时由系统创建，然后将其挂起等待处理事务，每个线程维护其事务缓冲池，当新的事务被发送过来时，根据优先级插入到事务缓冲池中，原则是保证高优先级事务优先得到执行。事务一般由 Begin Transcation 开始至 Commit 或 Roll back 结束。事务开始执行后，在自身的线程空间内对数据对象进行读、写的操作，如果事务正常执行直到全部操作均完成，那么它将在执行提交(Commit)后结束；在事务执行过程中也可能被夭折(abort)而发生回滚(Roll back)或者等待其它事务完成后再继续执行。

事务执行时，分为两个阶段：

（1）预备阶段：系统接收到事务请求以后，首先创建事务结构体并把它加入到新建队列中，分配事务操作所需要的系统资源(比如内存等)。然后进行优先级的分配。并根据就绪队列里事务处理情况，将事务加入到就绪队列中等待调度。

（2）执行阶段：事务调度器根据系统资源的状态，从就绪队列中选取优先级最高的事务，开始事务的执行。当前正在执行的事务均进入活动队列。事务正常执行完成后进行提交，如果执行过程中与其它事务的执行发生冲突，由并发控制算法进

*基金项目：国家自然科学基金项目(61064011)。

***Foundation Items:** The National Natural Science Foundation of China(61064011)

行处理，或者进入等待队列，或者进行事务回滚(即恢复已对数据库的部分操作后回到调度前的状态，重新排队等待调度)。

2 事务的一般属性及事务优先级的分配

事务一般由对数据对象的读、写操作组成，在有的情况下，还存在事务嵌套子事务的模式。一个实时事务具有多项属性。对不同的实时系统，事务所具有的属性会有不同，通常包括事务到达时间、事务截止期、松弛时间、事务执行时间、事务价值、事务优先级等。事务的这些属性信息随事务的创建而创建，并随着事务的提交或夭折而销毁。事务调度算法与并发控制协议根据各事务队列中事务的这些属性进行操作。

事务调度的前提是优先级分配，与常规数据库系统优先级分配不同，实时事务的优先级分配需要充分考虑到事务截止期的特性，实时事务优先级与以下因素有关：

（1）紧迫度。事务越紧急，它的优先级越高；

（2）截止时间。截止时间越早，事务的优先级越高；

（3）年龄。到达早的事务优先级高于到达晚的事务，有助于保持数据的外部一致性；

（4）松弛度。一个事务的松弛度越紧，优先级越高。其中，松弛度用于衡量一个事务的执行能被延迟多长时间而同样可能保证事务满足截止期要求。

（5）未完成工作量。未完成工作量越少，事务优先级越高；

（6）已完成工作量。已完成工作量越多，优先级越高；

硬实时事务(HRT)、软实时事务(SRT)和非实时事务（NRT)它们的优级顺序是 Priority(HRT)>Priority(SRT)>Priority(NRT)。Priority(NRT)最低，一般安排在系统空闲时执行。

3 事务调度算法

根据查阅大量参考文献，学习了目前常用的事务调度算法有如下几种：

（1)FCFS (First Come First Serve) 算法，也称为 ERF (Earliest Release First) 算法。该策略将最高优先级指派给具有最早“放行”时间的事务。所谓放行时间就是事务可以投入运行的最早时间，一般就是事务到达时间。该策略主要优点是实现简单，适合截止期容易满足事务调度, 但算法由于没考虑到定时限制，不适合于实时系统。

（2）EDF (EDF，Earliest Deadline First) 算法。它使截止期最早的事务的优先级最高。在实时性要求较低的系统中或某些特殊的情况下才采用这种算法，其主要缺点是已过截止期或几乎要过截止期的事务将获得最高优先级，会导致系统性能下降。

（3）HVF (Highest Value First) 算法。每个事务都有一个价值函数，其值最大者最优先，关键是如何构造合适的价值函数。

（4）GVDF (Greatest Value Density First) 算法。即完成事务的期望价值与实现该价值所需计算量的比值最大的事务优先级最高。对于期望价值一样的函数，该策略偏向于较短者，它每单位消耗时间所获得的价值更大。该策略也涉及到如何构造价值函数的问题。

（5）LSF(Least Slack First)算法。本文在传统 LSF 算法基础上提出一种运行时间自适应的 LSF 算法。

4 优化的事务调度算法——运行时间自适应的 LSF 算法

LSF 算法不但考虑了事务的截止时间，同时还参考了系统运行时间。它以事务的松弛度来安排其优先级。根据优先级赋值的时机，分为静态 LSF 和动态 LSF 两种。静态 LSF 的松弛度计算公式可表示为：

s=d-（t+rte）　　（1）

动态 LSF 的计算公式可表示为：

s=d-（t+rte-p）　　（2）,在（1）（2）式中，

s——Slack,表示事务的松驰度；

d——Deadline,表示事务的截止期；

t——current time,表示系统目前的时间；

rte——run time estimate,表示事务的估计运行时间；

p——service time，表示事务在系统中已经运行的时间。

在该算法中，计算出准确的 rte 非常重要。一般情况下，根据事务处理所需数据的特征和数据库操作类型来得出 rte。将实时数据库常驻内存，一方面是为了保证运行速度，即保证数据访问的实时性；另一个重要的原因是为了数据操作时间的可计算性。在流程工业的实时应用中，有许多事务是周期性事务，如数据采集、实时数据显示等。根据事务周期性的特点，可以充分利用事务执行的历史信息，计算出更准确的 rte，这就是自适应 LSF 算法的指导思想。

在系统初始化时，创建历史执行时间表。历史执行时间表中的元素 his_atom 定义为一个五元组：his_atom=(t，rs，ws，tl，tm),其中，

t 为事务的读、写、更新类型；

rs={tagl，tag2，…，tagn}为读取数据的集合：

ws={tagl，tag2，…，tagn)为写入数据的集合；

tl 为上一次执行时间；

tm 为平均执行时间。

自适应 LSF 算法步骤描述如下：

1)读取事务 Ctran；

2)从历史执行时间表中查找事务；

3)若找到相同的事务，得到对应的 t1 和 tm；转第 5 步;

4)若未找到相同事务，估算 rte；创建一个 his_atom 加入到历史执行时间表；

5)计算松弛度 S 和事务优先级；

6)事务执行后将执行时间更新到历史执行时间表中。

历史执行时间表的尺寸根据并发用户数目和数据源的数目确定。

5 调度策略

实时事务调度采用基于优先级驱动可抢占调度策略。即优先级高的事务可抢占优先级低的事务所拥有的资源，使优先级低的事务处于等待或者回滚进入就绪状态，待优先级高的事务执行完毕再进行调度执行。事务调度策略解决事务状态之间的转换，事务状态包含：就绪(Ready)、执行(Active)、完成(Commit)、阻塞(Wait)、回滚(Roll back)。事务调度状态如图 1 所示。

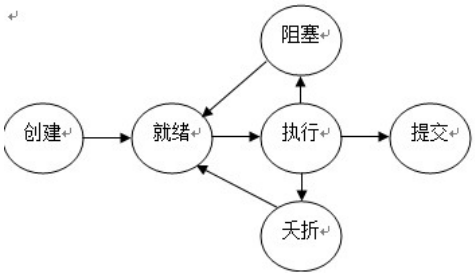

图 1 事务调度状态图

事务状态转换情况如表 1 所示：

表 1 事务状态转换表

事务状态转换	**转换条件或原因**
创建→就绪	当一个新的事务到达时，由事务管理器(TM)创建该事务，并分配内存、分配优先级后进入就绪状态
就绪→执行	处于就绪状态的事务，事务管理器从中选择高优先级的事务，安排相应的线程执行
执行→提交	处于活动状态的事务，在自身线程空间中对数据对象进行操作，完成以后更新到数据库中。然后事务管理器将事务销毁
执行→夭折	当有更高优先级的事务要执行时，当前事务被夭折，恢复对数据更改的操作，并回滚到就绪状态，重新安排优先级后进入就绪队列；对于超过截止期的硬实时事务，不进行回滚
执行→阻塞	当事务请求的数据被其它事务锁住时，进入等待状态。如果不等待，就进入就绪状态，重新调度

6 结束语

本算法具有两大优点：（1)rte 的计算与 CPU 时钟无关，具有自适应的能力；（2)对于周期性事务，采用本算法可得到准确的事务运行时间。唯一缺点是，当前事务与历史事务的匹配增加了时间，但可以考虑通过 HASH 等办法提高性能，避免一一比对。该算法对于周期性事务比例(>30％)比较高的应用系统非常适合。

参考文献

1 张晨艳．实时数据库系统特征及事务处理．论述与研究【J】，2006

2 桑楠．嵌入式应用原理及应用开发技术[M]．北京：北京航空航天大学出版社．2003

3 刘云生．实时数据库系统[J]，计算机科学，1994，21(3)：42-46．

4 刘云生，夏家莉．基于功能替代的实时事务调度[J]，计算机学报，2003，26(2)：250．256

5J.Stankovie，Misconceptions about real—time computing，IEEE Computer 21(1O)(1988)lO—19．

6 刘铭．大型工控软件发展中的若干关键技术研究[D]．．西安交通大学，2003．

7Kao B，Garcia-Molina H．An Overview of Real-Time Database Systems[J]，Upper Saddle River,NJ，USA：Prentice—Hall，Ine．，1995．

8 薛竹飙．实时内存数据库关键技术的研究与实现[D]．东南大学，2006．

9 周东球，杜殿林，左信．先进控制软件系统实时数据库的设计[J]．微计算机信息，2003，(10)：23 一-24．

10 何熠，吴爱国．监控组态软件实时数据库系统体系结构的研究[J]．制造业自动化,2007，(01)：84～86．

11 王常力，罗安．分布式控制系统(DCS)设计与应用实例[M]，北京：电子工业出版社，2004．

作者简介：张云(1981－),女,硕士,讲师,主要研究领域为实时数据库,工业控制