文章编号:1004 - 4736(2008)02 - 0098 - 04

基于时间变元绑定的双时态索引模型研究

李晓林,刘　莉
(武汉工程大学计算机科学与工程学院,湖北 武汉 430074)

摘　要:在基于点方法的双时态索引算法基础上,引入了时间变元绑定的形式化描述,扩充了其索引的双时态数据,使由于时间变元绑定而导致数据错误的双时态索引问题得以解决,给出了相应的数据变换和查询变换模型,并指出了其实现方式.

关键词:双时态索引;时间变元绑定;数据变换;查询变换
中图分类号:TP 311　　文献标识码:A

0 引　言

双时态数据库是在传统的数据库基础上加上有效时间和事务时间维的时态数据库,它既保存了数据库变迁的历史,又保存了现实世界的真实的数据属性.然而双时态数据库中的数据容量也是海量的,如果没有有效的数据管理方法,双时态数据的时态信息优势将被超大的数据访问代价所抵消.因此如何快速访问到时间数据是时态数据库中的一个重要研究课题.

几种主要的双时态索引技术有 R tree 系列空间索引、GR-tree(Generalize R tree)和 4R-tree[1] 索引技术.R-tree 系列空间索引因为不能有效解决带有效时间变元 *NOW* 和事务时间变元 *UC* 的双时态数据情况,所以并不能直接用于索引双时态数据.而 GR-tree 虽然能有效的处理含时间变元 *NOW* 和 *UC* 的双时态数据,又可以处理一般的双时态数据,但其算法的实现还有待于现有 DBMS 内核的扩展.4R-tree 的索引效果与 GR tree 相当,该方法实现方式可分为:基于支持 R-tree 索引的 DBMS 和基于不支持 R-tree 索引的 DBMS[2](即借助时态中间件),但两种实现方式都需要一个在 4 棵 R-tree 之上的实现层来执行其插入、删除和查询算法.

文献[3]提出了一种基于点的双时态索引算法,这种算法不仅继承了 R*-tree 的优点,又能有效的索引含时间变元的双时态数据,更有实验证明该算法在磁盘 I/O 访问次数和 CPU 使用率的性能上高出最大时间戳 R*-tree(利用可能时间的最大值来代替事务和有效时间的不确定值)的 20 倍以上.但是这种算法不足之处在于:只能索引满足 $Vt^{-}<Vt^{+}$,$Tt^{-}<Tt^{+}$ 条件的双时态数据,但是现实中满足 $Vt^{-}=Vt^{+}$ 或 $Tt^{-}=Tt^{+}$ 条件的双时态数据也是广泛存在的,例如教师工资系统中,李明从 2007 的 7 月份起职称由讲师升为副教授,工资也从原来的 1 300 涨到 1 700,这时在教师工资数据库中插入元组(李明,副教授,1 700,2007-07,*Now*,2007-09-18,*UC*)表示这条信息,但随后马上发现李明的工资是从 6 月份调整的,数据库中会用(李明,副教授,1 700,2007-07,*Now*,2007-09-18,2007-09-18)、(李明,副教授,1 700,2007-06,*Now*,2007-09-18,*UC*)两个元组来表示这次的修改,其中前一条记录表示原来信息的删除,后一条表示目前的状况,如果在原算法中发现这种数据,势必导致索引错误.为了使此算法更具一般性,本文在原有算法的基础上引入了时态变元绑定形式化描述,扩充了索引的双时态数据类型,并改进了相应的索引模型,最后说明了其实现方式并给出了下一步研究的方向与任务.

1 时间变元绑定

1.1 双时态数据类型

双时态数据有两个时间维组成:有效时间(Valid Time)和事务时间(Transaction Time)[4],有效时间指的是指一个对象(事件)在现实世界中发生并保持的那段时间,或者该对象在现实世界中为真的时间.事务时间是指一个数据库对象进行操作的时间,是一个事实储存在数据库中的时间,它记录着对数据库修改和更新的各种操作对应于现有事务或现有数据库状态变迁的历史.目前使用比较广泛是 TQuel 的四个时间戳模式:$(Vt^{-},Vt^{+},Tt^{-},Tt^{+})$[5],它们分别表示有效时间的起始、终止时间和事务时间的起始、终止时间.

收稿日期:2007 - 12 - 02

作者简介:李晓林(1962 -),男,湖北安陆人,副教授,硕士.研究方向:计算机网络,数据库系统.

当有效时间的终止值不确定的时候，可以用 *Now* 来表示有效时间的终止，而事务时间的终止值不确定时，就用 *UC* 来表示事务时间的终止，且双时态数据的时态约束条件为：$Vt^{\vdash} \leqslant Vt^{\dashv}$，$Tt^{\vdash} \leqslant Tt^{\dashv}$ 且 $Tt^{\vdash} \leqslant CT$(Current Time)．根据有效时间和事务时间的终止值是否为不确定值，按四个时间戳为坐标围成的双时态域可将双时态数据分为固定矩形、增长矩形、固定楼梯形和增长楼梯形四种类型．

1.2　时间区间绑定

时间区间根据是否含有变元 *Now* 可分为：*Now* 时间区间和 Ground 时间区间[6,7]，前者表示时间区间中含有时间变元 *Now*；后者表示时间区间中不含有时间变元．下面时间区间绑定的定义中要用到一种特殊的时间：参考时间(Reference Time 简称 Rt)，它是指时态数据库中的用户观察数据库的时间，可有多种选择．

定义 1　设时间区间 $I=[t_1,t_2]$，则时间区间的绑定如下：

$$I_{Rt}^{\text{bind}}=[t_1,t_2]_{Rt}^{\text{bind}}=[t_{1Rt}^{\text{bind}},t_{2Rt}^{\text{bind}}]=\begin{cases}[t_1,t_2]\text{当为 Ground 时间凸区间时}\\ [t_1,R_t]\text{当为 } Now \text{ 时间凸区间时}\end{cases}\tag{1}$$

由式(1)可见，对于 Ground 时间区间的绑定是没有改变原来的时间凸区间的，而 *Now* 时间区间的绑定是将 *Now* 用参考时间取代即可．本文时间变元 *Now* 绑定是采用了用户观察数据库的时间即 *Rt* 是等于时间区间的上限的．如下式：

$$I_{Rt}^{\text{bind}}=[t_1,t_2]_{Rt}^{\text{bind}}=[t_{1Rt}^{\text{bind}},t_{2Rt}^{\text{bind}}]=[t_1,t_1]$$

式中 $t_2=Now$．

那么绑定后双时态域对应的变换为固定矩形、平行于 *Vt* 轴的线段、平行于 *Tt* 轴的线段和点．

在固定矩形这种双时态数据类型中，由于 $Vt^{\vdash}=Vt^{\dashv}$ 或 $Tt^{\vdash}=Tt^{\dashv}$，存在一种退化了固定矩形，其双时态域表现为点、平行于 *Vt* 轴的线段或则平行于 *Tt* 轴的线段．并且经过时间区间的绑定，其双时态域不变，那么这些特殊的固定矩形势必跟其他几种双时态数据类型发生混乱，造成索引错误．解决这个问题的最简单有效的方法就是引入一个变量来记录时间变元绑定前的双时态数据类型．

2　双时态数据变换

以上描述了时态变元绑定的思想，下面给出其形式化、规范化定义．首先给出双时态数据的定义，然后给出双时态数据变换的函数．

定义 2　已知时间点域 *T*，元组指针域为 ID，具有变元的双时态数据 D^{CT} 和基于点的双时态域 D^{point} 的定义如下：

$$D^{CT}\subseteq\{\langle Vt^{\vdash},Vt^{\dashv},Tt^{\vdash},Tt^{\dashv},id\rangle\in T\times(T\cup\{\text{Now}\})\times T\times(T\cup\{UC\})\times ID\mid(Vt^{\vdash}=\text{Now}\vee Vt^{\vdash}\leqslant Vt^{\dashv})\wedge(Tt^{\vdash}=UC\vee Tt^{\vdash}\leqslant Tt^{\dashv})\}\tag{2}$$

$$D^{\text{point}}\subseteq\{\langle Vt^{\vdash},Vt^{\dashv},Tt^{\vdash},Tt^{\dashv},id\rangle\in T\times T\times T\times T\times ID\mid(Vt^{\vdash}\leqslant Vt^{\dashv})\wedge(Tt^{\vdash}\leqslant Tt^{\dashv})\}\tag{3}$$

式(2)中双时态域 D^{CT} 含有时间变元 *Now* 和 *UC*，而 D^{point} 是不含时间变元的．下面介绍从双时态域 D^{CT} 到不含时间变元 D^{point} 的数据变换函数．

定义 3　令 $r\in D^{CT}$，类型 Type$=\{1,2,3,4\}$，那么从双时态域 D^{CT} 到不含时间变元 D^{point} 的数据变换函数为：$f: D^{CT}\rightarrow D^{\text{piont}}\times\text{Type}$

$$f(r)=f(\langle Vt^{\vdash},Vt^{\dashv},Tt^{\vdash},Tt^{\dashv},id\rangle)=\begin{cases}\langle Vt^{\vdash},Vt^{\dashv},Tt^{\vdash},Tt^{\dashv},id,1\rangle \text{ if } Vt^{\vdash}=\text{Now}\wedge Tt^{\dashv}=UC\\ \langle Vt^{\vdash},Vt^{\dashv},Tt^{\vdash},Tt^{\dashv},id,2\rangle \text{ if } Vt^{\vdash}=\text{Now}\wedge Tt^{\dashv}\neq UC\\ \langle Vt^{\vdash},Vt^{\dashv},Tt^{\vdash},Tt^{\dashv},id,3\rangle \text{ if } Vt^{\vdash}\neq\text{Now}\wedge Tt^{\dashv}=UC\\ \langle Vt^{\vdash},Vt^{\dashv},Tt^{\vdash},Tt^{\dashv},id,4\rangle \text{ if } Vt^{\vdash}\neq\text{Now}\wedge Tt^{\dashv}\neq UC\end{cases}$$

由数据变换函数的定义可知，*f* 是双射函数，其中 Type 中的元素是用于记录数据变换函数中原像的数据类型．

3　查询变换及其实现方式

数据变换后，在变换后的数据域上的查询也要经过一定的变换才能操作．时态查询语言主要包括时间点的查询和时间区间的查询，所以查询变换分为基于时间点的查询变换和基于时间区间的查询变换．

3.1　基于时间点的查询变换

时间点的查询可以分为三种[8,9]，其一是查找当前概念中世界的当前状态，记为 Q1；第二种是查找当前概念中世界的过去状态，记为 Q2；第三种是查找以往概念中关于世界的过去状态，记为 Q3．如图 1 所示．

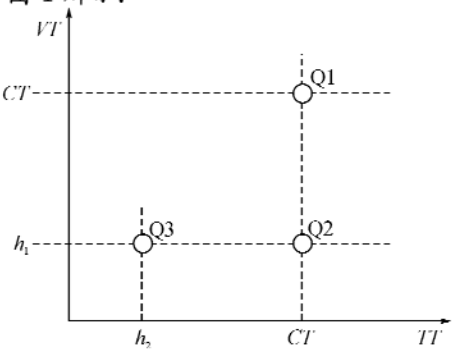

图 1　点查询的分类：Q1、Q2 和 Q3

Fig. 1　Timeslice queries Q_1、Q_2 and Q_3

由图1可知，Q1选择查询有效时间区间与Now相交且事务时间到Now的元组. 因此在D^{CT}中Q1选择元组所要满足查询条件为：

$(Vt^{\rightarrow}\leqslant Now\wedge Vt^{\leftarrow}\geqslant Now)\wedge(Tt^{\rightarrow}\leqslant Tt^{\leftarrow}\wedge Tt^{\leftarrow}=UC)$

在基于时间变元绑定的双时态数据存储中，由于$[t_1,t_2]_R^{bind}=[t_1,t_1]$当$t_2=\mathrm{Now}\vee UC$，替换过来元组所要满足的查询条件为：

$$[(Vt^{\vdash}\leqslant CT\wedge Vt^{\dashv}>CT)\wedge Tt^{\dashv}=Tt^{\rightarrow}\wedge \mathrm{GetType}()=3]\vee[(Vt^{\rightarrow}\leqslant CT\wedge Vt^{\leftarrow}=Vt^{\rightarrow})\wedge Tt^{\leftarrow}=Tt^{\rightarrow}\wedge \mathrm{GetType}()=1] \quad (4)$$

由式(4)中的$Tt^{\dashv}=Tt^{\vdash}$可知，搜索元组在D^{point}中表现为点和线，且该点是位于区域$(0,Now,0,Now)$内的，线是平行于Vt轴并与$Vt=CT$相交的. 其中GetType()函数是用来确认点和线是由含变元双时态域中的增长楼梯形和增长矩形变换而来.

而Q2选择查询的是有效时间包括过去的某个时间点h且事务时间到Now的元组. 因此在D^{CT}中Q2选择元组所要满足的查询条件为：

$(Vt^{\rightarrow}\leqslant h\wedge Vt^{\leftarrow}\geqslant h)\wedge(Tt^{\rightarrow}\leqslant Tt^{\leftarrow}\wedge Tt^{\leftarrow}=UC)$

在基于时间变元绑定的双时态数据存储中，由于$[t_1,t_2]_R^{\mathrm{bind}}t=[t_1,t_1]$当$t_2=Now\vee Uc$，替换过来元组所要满足的查询条件为：

$$[(Vt^{\vdash}\leqslant h\wedge Vt^{\dashv}\geqslant h)\wedge Tt^{\dashv}=Tt^{\vdash}\wedge \mathrm{GetType}()=3]\vee[(Vt^{\vdash}\leqslant h\wedge Vt^{\leftarrow}=Vt^{\rightarrow})\wedge Tt^{\leftarrow}=Tt^{\vdash}\wedge \mathrm{GetType}()=1] \quad (5)$$

由式(5)中的$Tt^{\dashv}=Tt^{\vdash}$可知，搜索元组在不含变元的双时态域中表现为点和线，且该点是位于区域$(0,Now,0,h)$内的，线是平行于Vt轴并与$Vt=h$相交的. 其中GetType()函数是用来确认点和线是由含变元双时态域中的增长楼梯形和增长矩形变换而来.

而Q3选择查询有效时间区间包括过去的某个时间点$h\ 1$且事务时间区间也包括过去的某个时间点$h\ 2$的元组. 因此在D^{CT}中Q3选择元组所要满足的查询条件为：

$(Vt^{\vdash}\leqslant h\ 1\wedge Vt^{\dashv}\geqslant h\ 2)\wedge(Tt^{\rightarrow}\leqslant h\ 2\wedge Tt^{\leftarrow}\geqslant h\ 2)$

在基于时间变元绑定的双时态数据存储中，由于$[\iota_1,\iota_2]_{Ie}^{\mathrm{bind}}=[\iota_1,\iota_1]$当$\iota_2=Now\vee UC$，替换过来元组所要满足的查询条件为：

$$[(Vt^{\rightarrow}\leqslant h\ 1\wedge Vt^{\leftarrow}\geqslant h\ 1)\wedge(Tt^{\rightarrow}\leqslant h\ 2\wedge Tt^{\leftarrow}\geqslant h\ 2)\wedge \mathrm{GetType}()=4]$$
$$\vee[(Vt^{\vdash}\leqslant h\ 1\wedge Vt^{\dashv}\geqslant h\ 1)\wedge(Tt^{\vdash}\leqslant h\ 2\wedge Tt^{\dashv}=Tt^{\rightarrow})\wedge \mathrm{GetType}()=3]$$
$$\vee[(Vt^{\rightarrow}\leqslant h\ 1\wedge Vt^{\leftarrow}=Vt^{\rightarrow})\wedge(Tt\leqslant h\ 2\wedge Tt^{\leftarrow}\geqslant h\ 2)\wedge \mathrm{GetType}()=2]$$
$$\vee[(Vt^{\vdash}\leqslant h\ 1\wedge Vt^{\dashv}=Vt^{\vdash})\wedge(Tt^{\vdash}\leqslant h\ 2\wedge Tt^{\dashv}=Tt^{\rightarrow})\wedge \mathrm{GetType}()=1] \quad (6)$$

由式(6)可知，搜索元组在D^{point}中囊括了所有类型，其中点是位于区域$(0,\ h\ 2,0,h\ 1)$内的，线是平行于Tt轴且与$Tt=h\ 2$相交的或平行于Vt轴并与$Vt=h\ 1$相交，而搜索元组对应的矩形是与点$(h\ 2,h\ 1)$相交的.

3.2 基于时间区间的查询变换

区间查询是指对有效时间和事务时间范围的查询，通常也称为矩形交叉查询. 元组$(Vt_q^{\vdash},Vt_q^{\dashv},Tt_q^{\vdash},Tt_q^{\dashv})\in T\times T\times T\times T$表示相交查询域，其中$T$表示时间域，查询条件为$Vt_q^{\vdash}\leqslant Vt_q^{\dashv}$，$Tt_q^{\vdash}\leqslant Tt_q^{\dashv}$且$Tt_q^{\dashv}\leqslant CT$. 在$D^{CT}$中区间查询的条件是：

$$[(Vt^{\vdash}\leqslant Vt_q^{\dashv}\wedge Vt^{\dashv}\geqslant Vt_q^{\vdash})\wedge(Tt^{\vdash}\leqslant Tt_q^{\dashv}\wedge Tt^{\dashv}\geqslant Tt_q^{\vdash})]$$

由文献[10]的双时态数据上的相交查询结果的定义，可以知道区间查询变换到D^{point}上应满足的查询条件为：

$$[(Vt^{\rightarrow}\leqslant Vt_q^{\leftarrow}\wedge Vt^{\leftarrow}\geqslant Vt_q^{\rightarrow})\wedge(Tt^{\rightarrow}\leqslant Tt_q^{\leftarrow}\wedge Tt^{\leftarrow}\geqslant Tt_q^{\leftarrow})\wedge \mathrm{GetType}()=4]$$
$$\vee[(Vt^{\rightarrow}\leqslant Vt_q^{\leftarrow}\wedge Vt^{\leftarrow}\geqslant Vt_q^{\rightarrow})\wedge(Tt^{\rightarrow}\leqslant Tt_q^{\leftarrow}\wedge Tt^{\leftarrow}=Tt^{\rightarrow})\wedge \mathrm{GetType}()=3]$$
$$\vee[(Vt^{\vdash}\leqslant Vt_q^{\dashv}\wedge Vt^{\dashv}=Vt^{\vdash})\wedge(Tt^{\vdash}\leqslant Tt_q^{\dashv}\wedge Tt^{\dashv}\geqslant Tt_q^{\dashv})\wedge \mathrm{GetType}()=2]$$
$$\vee[(Vt^{\vdash}\leqslant Vt_q^{\dashv}\wedge Vt^{\dashv}=Vt^{\vdash})\wedge(Tt^{\vdash}\leqslant Tt_q^{\dashv}\wedge Tt^{\dashv}=Tt^{\vdash})\wedge \mathrm{GetType}()=1] \quad (7)$$

式(7)中搜索元组在D^{point}中表现为位于$(0,Tt_q^{\dashv},0,Vt_q^{\dashv})$区域内的点、平行于$Tt$轴并与$(Tt_q^{\vdash},Tt_q^{\dashv},0,Vt_q^{\dashv})$区域相交的线段、平行于$Vt$轴并与$(0,Tt_q^{\dashv},Vt_q^{\vdash},Vt_q^{\dashv})$区域相交的线段和与$(Tt_q^{\vdash},Tt_q^{\dashv},Vt_q^{\vdash},Vt_q^{\dashv})$相交矩形.

3.3 实现方式

进行查询变换后，双时态数据域不存在增长楼梯形和增长矩形这些不规则形状，因此可以直接支持基于R tree索引的DBMS. 譬如本模型可以直接利用Oracle Spatial来完成对含时态变元的双时态数据的管理. 这样不仅可以继承了Oracle Spatial提供的R*-tree索引的优点，有效的解决了索引带有效时间变元NOW和事务时间变元UC的双时态数据情况，并且与GR-tree和4R-tree相比，本算法只需要在现有的空间索引技

术上进行逻辑的查询变化即可，而不需要对现在的 DBMS 内核进行扩展.

4　结　语

对文献[3]模型的一种改进方法，主要解决了当录入数据库中的数据由于某种原因发生错误后及时更正的双时态数据索引问题. 此外还需要对模型做进一步研究，下一步的研究工作有两点：(1)将有效时间变元 *Now* 和事务时间变元 *UC* 分别进行绑定，研究处理时间变元的最合理算法. (2)将时态变元绑定从数据变换函数中独立出来，方便修改 *Now* 的绑定参考时间以适应不同的需求.

参考文献：

[1] Bliujute. R, Jensen C S, Saltenis S, et al. R-Tree Based Indexing of Now-Relative Bitemporal Data [A]. Proceedings of 24rd International Conference on Very Large DataBases (VLDB'98) [C]. New York: Morgan Kaufmann Publishers, 1998, 345-356.

[2] 康向锋，汤　庸，叶小平，等. 一种基于时态中间件的高效双时态索引模型[J]. 计算机科学，2005，32(9)：91-95.

[3] Stantic. B, Khanna. S, &Thornton. J. An Efficient Method for Indexing Now-Relative Bitemporal Data [A]. Klaus-Dieter Schewe and Hugh E. Williams, Proceedings of the 15th Australasian database conference[C]. Dunedin New Zealand. : Australian Computer Society, 2004, 113-122. .

[4] 汤　庸，汤　娜，叶小平. 时态信息处理技术研究综述[J]. 中山大学学报(自然科学版)，2003，42(4)：5-9.

[5] Snodgrass R T. Temporal Database [J]. IEEE Computer, 1986, 19(9): 35-42.

[6] 王路帮，叶小平，邢鸿雁，等. 时间变元 now 的绑定算法[J]. 中山大学学报(自然科学版)，2004，43(增刊)：152-154.

[7] 王路帮. 时态数据库中的时间变元及其绑定[J]. 浙江大学学报(理学版)，2006，33(6)：662-666.

[8] Torp K, Jensen C S, Snodgrass R T. Stratum Approaches to Temporal DBMS Implementation[A]. Proceedings of the 1998 International Symposium on Database Engineering & Applications [C]. Cardiff Wales: IEEE Computer Society, 1998, 4-13.

[9] Stantic B, Thornton J, Sattar A. A Novel Approach to Model Now in Temporal DataBase [A]. 10th International Symposium on Temporal Representation and Reasoning and Fourth International Conference on Temporal Logic (TIME-ICTL2003) [C]. cairns Australia: IEEE Computer Society, 2003, 174-180.

[10] 汤　庸. 时态数据库导论[M]. 北京：北京大学出版社，2004，139-144.

Research of bitemporal index model based on the temporal variables binding

LI Xiao-lin, LIU Li

(School of Computer Science and Technology, Wuhan Institute of Technology, Wuhan 430074, China)

Abstract: This paper introduces the formal description of temporal variables binding based on the point approach, extends bitemporal data to deal with the bitemporal index problem of erroneous data caused by temporal variables binding, provides corresponding model of data transformation and query transformation, then it demonstrates the implemented quomodo.

Key words: Bitemporal index, temporal variables binding, data transformation, query transformation

本文编辑：陈晓苹