并发更新环境下的视图维护异常研究*

舒忠梅[1]，李师贤[1]，左亚尧[2]
（1. 中山大学计算机科学系，广东 广州 510275；
2. 广东工业大学计算机学院，广东 广州 510090）

摘 要：通过对视图维护过程中的更新异常进行分析，发现并发更新是导致视图维护异常的主要原因。在并发环境下，数据更新和模式更新极有可能同时存在，这对视图维护工作带来极大的困难。在实例分析基础上，深入而细致地刻画了模式与数据全面并发的典型情形，为视图维护工作奠定基础。

关键词：并发更新；视图维护；维护异常
中图分类号：TP311 **文献标识码**：A **文章编号**：0529-6579（2005）05-0021-04

在一个典型的集成分布式数据源（Data Source，DS）数据的数据仓库（Data Warehouse，DW）系统中，不仅数据源的数据会产生更新，模式也可能发生改变。而且这些数据源可能并发地产生数据更新（Data Updates，DU）和模式更新（Schema Changes，SC），不会考虑任何对基于它们之上定义的实化视图的影响。

1 相关工作

当数据源产生更新之后，如何保持数据仓库与数据源的一致性是保证数据仓库中数据质量的一个关键问题，一致性视图维护方法也成为数据仓库中的重要研究领域[1]。最近几年，许多学者从不同的角度提出了一些有效的方法，如ECA算法（Eager Compensating Algorithm）[2]、Strobe算法和C-Strobe算法[3]、MDVM（Multiple Dimension View Maintenance）算法[4]及PMDVM（Parallel Multiple Dimension View Maintenance）[5]算法等，然而这些方法都忽略了模式更新问题。对于模式更新以及模式与数据都发生更新甚至并发更新的这类更加复杂、但又更接近现实状况的更新问题还没有进行深入地研究[6]。本文分析了在全面并发更新条件下导致维护异常问题的主要原因，深入分析并刻画了模式与数据全面并发更新的典型情形，对相关的视图维护工作奠定了良好的基础。

2 更新异常维护示例

例1 令实化视图定义为 $MV=\prod_{A,D,F}\sigma_{((B=C)\wedge(D=E))}R_1\times R_2\times R_3$。其中，$R_1$，$R_2$，$R_3$是3个分别位于数据源$X$，$Y$，$Z$的关系。初始关系和实化视图如下：

$$X.R_1:\begin{matrix}A&B\\4&2\\3&5\end{matrix}\quad Y.R_2:\begin{matrix}C&D\\2&3\\5&2\end{matrix}\quad Z.R_3:\begin{matrix}E&F\\3&4\\2&6\end{matrix}\quad MV:\begin{matrix}A&D&F\\4&3&4\\3&2&6\end{matrix}$$

将实化视图维护中所产生的维护消息形式化地描述为：op _ pred（op _ ob，msg _ start）→msg _ term：t_1，其中，操作谓词op _ pred∈{insert，update，delete，rename，query，answer，commit}；op _ ob代表更新或维护查询的数据和模式，元组和模式分别用“[…]”和“(…)”来引导操作对象；msg _ start代表消息的发起对象；msg _ term代表消息的被受端，它或者是数据源或者是数据仓库；t_i表示更新产生的时刻且满足$\forall i,j$：$t_i<t_j$（$i<j$），符号$<$表示前者发生的时刻先于后者。考虑顺序产生如图1的数据更新处理消息序列：

① 数据仓库收到从数据源X产生的数据更新：insert（[2，5]，$X.R_1$）→DW：t_1；

② 数据仓库根据增量维护算法，发起维护查询：query(([2，5]$\times Y.R_2$)，DW)→Y：t_2；

③ 数据仓库收到数据源Y产生的更新：delete(([5，2]$\times Y.R_2$)→DW：t_3；

④ 数据源Y响应维护查询：answer(([2，5]$\times Y.R_2$)→DW：t_4；

⑤ 数据仓库继续发起维护查询：query(([2，5]$\times Y.R_2\times Z.R_3$)，DW)→Z：t_5；

⑥ 数据源Z响应维护查询：ΔMV_1 = answer

* 收稿日期：2004-11-24
基金项目：广东省科技计划工业攻关基金资助项目（2003A1030403）
作者简介：舒忠梅（1974年生），女，博士生，讲师；E-mail：issszm@zsu.edu.cn

$(([2,5] \times Y.R_2 \times Z \setminus \mathcal{R}_3) = \phi \rightarrow DW: t_6$；

⑦ 对更新进行类似的处理得到$\Delta MV_2 = -((3,2,6),(2,2,6))$。

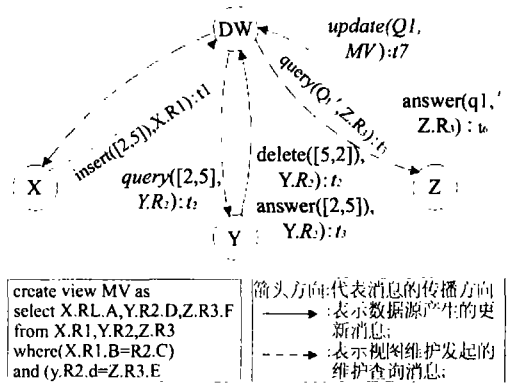

图1 一个数据更新的异常维护示例

Fig.1 An example of maintenance anomaly caused by a data update

讨论：

（1）如果各数据源顺序地产生其更新，如在例1的处理过程中，即先产生 ΔR_1 并在 R_2 处正确地响应了维护查询，而后产生 ΔR_2，也即 t_3 与 t_4 时刻作交换后的顺序。对于这种顺序处理，数据仓库先提交 $\Delta MV_1 = +(2, 2, 6)$，再提交 $\Delta MV_2 = -((3, 2, 6), (2, 2, 6))$，2个更新后的最终视图状态为 $MV = (4, 3, 4)$，称之为正确的视图维护。

（2）如果更新的产生顺序如图1所示，即 ΔR_1 在 R_2 处的维护查询尚未进行时 R_2 也产生了更新 ΔR_2，上面的处理顺序描述了这个处理过程。由于在 R_2 处进行维护查询时，ΔR_2 已经发生，导致 $\Delta MV_1 = \phi$，同样对 ΔR_2 进行维护查询后 $\Delta MV_2 = -((3, 2, 6), (2, 2, 6))$，但是在 MV 中并不存在可删除的元组 $(2, 2, 6)$，这导致在数据仓库端产生一个提交异常，称之为异常维护。

3 异常维护分析

为了探寻导致异常维护的原因，本文从时序的角度对图1的示例进行分析，按照维护消息产生的时序，给出一个时间线描述，如图2所示。图中实线表示更新 $X.\Delta R_1$ 的处理时序，虚线表示 $Y.\Delta R_2$ 的处理时序。从图中可以非常清晰地看到2个维护消息在 $[t_3, t_4]$ 时间区间产生了时序交叠，这是因为在 R_2 响应 ΔR_1 的维护查询时并发地产生了 ΔR_2，从而扰乱了 R_2 对 ΔR_1 的维护查询的正常处理时序。

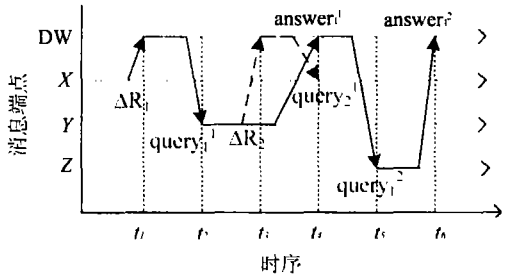

图2 异常维护的时序

Fig.2 The temporal relation of maintenance anomaly

定义1 数据源 DS_j 在响应DW对数据源 DS_i 的更新 DU_m 或 SC_m 的维护查询或视图调整的过程中，DS_j 本身也可能并发地产生更新 DU_n 或 SC_n，其中 $i \neq j$，$m \neq n$，$Q^l_{l,M}$ 表示对 DS_i 的第 m 个更新发起的第 l 个维护子查询。把 DU_n 或 SC_n 称为 DU_m 或 SC_n 的视图维护并发更新（Maintenance Concurrency Updates），简称为并发更新（Concurrency Updates），记为 $XU_m \Uparrow XU_n$，其中 $XU \in \{DU, SC\}$，下同。

4 并发更新类型分析

从上面的分析不难看出，并发是导致维护异常的一个重要原因，要想有效地对并发更新进行正确维护，必须首先分析这些并发更新的类型，然后才能针对不同的更新问题采用相应的纠正策略。

4.1 数据并发更新

例1说明了一个典型的数据并发更新情形，数据更新是指在数据源之上产生的数据内容更新，它只是元组的更新问题，不涉及到任何模式。

定义2 数据并发更新：

情形1 异源数据并发更新：令 ΔR_i 和 ΔR_j 代表2个在 R_i 和 R_j $(i \neq j)$ 产生的更新，更新 ΔR_j 称为 ΔR_i 的维护并发更新，如图3(a)，当且仅当：

① 中间层收到 ΔR_i 的时间要早于 ΔR_j，（即 $i < j$）；②数据仓库在收到 R_j 响应 ΔR_i 而发起的子查询 Q_i 结果之前收到 ΔR_j。

情形2 同源数据并发更新，令 $\Delta_i R$ 和 $\Delta_j R$ 代表在某一 R 处产生的2个连续更新，更新 $\Delta_j R$ 称为 $\Delta_i R$ 的维护并发更新，如图3(b)，当且仅当：

① 中间层收到 $\Delta_i R$ 的时间要早于 $\Delta_j R$，（即 $i < j$）；②数据仓库在收到其它数据源响应 $\Delta_i R$ 而发起的子查询 Q_i 结果之前收到 $\Delta_j R$。

4.2 模式并发更新

不仅数据元组会产生更新，模式也可能发生改

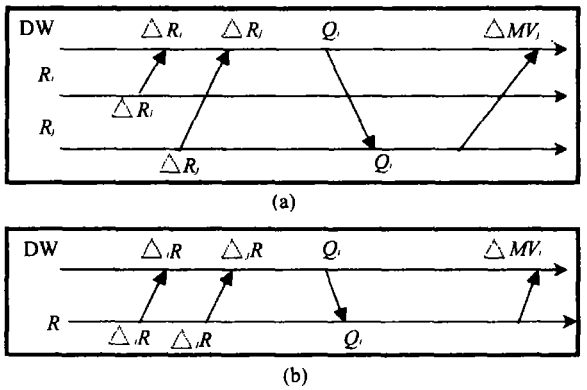

图 3　数据并发更新情形

Fig.3　The cases of concurrent data updates

变，模式的改变既可能影响实化视图中的元组，也可能会影响视图的定义，最终导致模式更新维护异常问题，使得基于原定义的视图不可维护。通常，模式更新后要对元数据进行同步，以对视图定义进行置换、重演算和重定义等视图调整（View Adaption，VA）工作；然后对实化视图中的元组数据进行加工，对模式更新引起的噪声数据、空缺数据和不一致数据进行重新集成数据归约处理。模式更新的主要操作包括增加、删除、重命名等，操作的对象可以是属性也可以是关系模式本身。下面的例 2 对例 1 发生模式改变后的视图维护情况进行分类分析。

例 2　考察关系模式 $X.R_1$ 更新为 $X.R_1'$ 的视图维护处理，初始关系模式如下：

$X.R_1:(A,B),Y.R_2(C,D),Z.R_3:(E,F)$，实化视图 MV 定义为 $\prod_{A,D,F}\sigma_{((B=C)\wedge(D=E))}R_1\times R_2\times R_3$。用形如 $DS_i.R_j:(F_1,F_2,\cdots F_n)\Rightarrow DS_i.R_j:(F'_1,F'_2,\cdots F'_m)\mid DS_i.R'_j:(F_1,F_2,\cdots F_n)\mid\phi$ 的式子来表示模式的具体改变，其中 DS_i 代表数据源，R_j 代表 DS_i 的一个关系模式，$F_l(0\leqslant l\leqslant n)$，$F'_k(0\leqslant k\leqslant m)$ 表示关系模式的属性，模式改变后的结果可以为 ϕ。

①插入更新：$X.R_1:(A,B)\Rightarrow X.R_1:(A,B,G)$，由于视图的定义不受形如 insert$((G),X.R_1)\rightarrow DW:t_1$ 的维护消息的影响，因此对于模式的插入更新操作，维护处理过程直接将它过滤掉。

②删除更新：包括模式删除和属性删除，考察以下的示例 $X.R_1:(A,B)\Rightarrow X.R_1:(A)\mid\phi$，删除使视图定义 $\prod_{A,D,F}\sigma_{((B=C)\wedge(D=E))}R_1\times R_2\times R_3$ 语义溢出，即相关模式更新使 MV 定义与数据源产生不一致性问题。要对 MV 进行重定义演算，对于模式删除，可以通过寻找 R_1 模式的真子集来进行替换，即如果 $\exists R'_1$ 且 $R_1\neq R'_1$，使 $R'_1\subset R_1$，则在 MV 定义中可用 R'_1 来置换 R_1；或者直接将该视图定义修改为不再联机维护。

③重命名更新：包括关系模式改名和属性改名，如 $X.R_1:(A,B)\Rightarrow X.R'_1:(A,B,G)$。显然，可以通过握手信号来追踪属性前后的修改，并通知元数据重新更新视图的定义 $\prod_{A,D,F}\sigma_{((B=C)\wedge(D=E))}R_1\times R_2\times R_3$ 或者 $\prod_{A,D,F}\sigma_{((B=C)\wedge(D=E))}R'_1\times R_2\times R_3$，使视图定义与数据源模式定义一致。

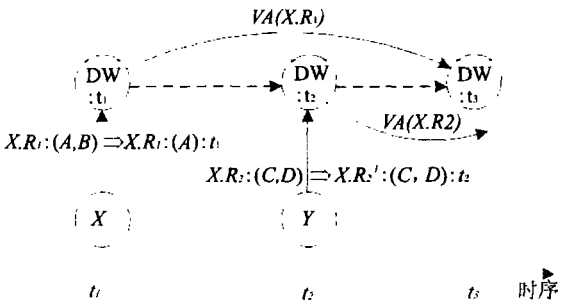

图 4　模式并发更新的时序示意图

Fig.4　The temporal relation of concurrent schema changes

在并发环境下，同样也会产生模式并发更新问题，如图 4 所示。模式的并发更新可能造成视图乱序调整，从而导致视图不可维护。

定义 3　异源模式并发更新：在 DW 响应模式更新 $SC_{iu}=DS_i.R_u:(F_{u1}\ F_{u2},\cdots,F_{um})\Rightarrow DS_i.R_u:(F'_{u1}\ F'_{u2},\cdots,F'_{um})\mid DS_i.R'_u:(F_{u1},F_{u2},\cdots,F_{um})\mid\phi$：$t_{iu}$ 的处理中，如果 $\exists SC_{jv}=DS_j.R_v:(F_{v1},F_{v2},\cdots F_{vn})\Rightarrow DS_j.R_v:(F'_{v1},F'_{v2},\cdots F'_{vn})\mid DS_j.R'_v:(F_{v1},F_{v2},\cdots F_{vn})\mid\phi$：$t_{jv}$，且：

①模式更新产生并送交 DW 的时序满足 $t_{SC_{iu}}<t_{SC_{jv}}$；

②视图调整 VA 时态满足 $t_{SC_{jv}}<t_{VA_{iu}}$，$t_{VA_{iu}}$ 表示 DW 完成因模式更新 SC_{iu} 而引起的视图调整的时刻。则称 SC_{iu} 和 SC_{jv} 构成异源模式并发更新，记为 $SC_{iu}\Uparrow SC_{jv}$。

定义 4　同源模式并发更新：在 DW 响应模式更新 $SC^l_{iu}=DS_i.R_u:(F_{u1},F_{u2},\cdots,F_{um})\Rightarrow DS_i.R_u:(F'_{u1},F'_{u2},\cdots,F'_{um})\mid DS_i.R^l_u:(F_{u1},F_{u2},\cdots,F_{um})\mid\phi$：$t^l_{iu}$ 的处理中，如果 $\exists SC^k_{iu}=DS_i.R_u:(F_{v1},F_{v2},\cdots F_{vn})\Rightarrow DS_i.R_u:(F'_{u1},F'_{u2},\cdots F'_{um})\mid DS_i.R^k_u:(F_{u1},F_{u2},\cdots F_{un})\mid\phi$：$t^k_{iu}$，且：

①模式更新产生并送交DW的时序满足 $t^l_{SC_{iu}} < t^k_{SC_{iu}}$；

②视图调整VA时态满足 $t^k_{SC_{iu}} < t_{VA_l}$，t_{VA_l} 表示DW完成因模式更新 SC^k_{iu} 而引起的视图调整的时刻，下同。则称 SC^l_{iu} 和 SC^k_{iu} 构成同源模式并发更新，记为 $SC^l_{iu} \Uparrow SC^k_{iu}$。

4.3 混合并发更新

混合更新是指既有数据更新又有模式更新的情形。当2个先后发生的混合更新在时序上满足一定的约束条件时，它们就会构成混合并发更新。

例3 考察例1中发生以下更新消息及维护处理的序列，并据此画出全部处理过程的维护处理时序图，如图5所示：

① insert($[2,5], X.R_1$)→DW: t_1；

② query(($[2,5] \times Y.R_2$),DW)→Y: t_2；

③ delete(($[5,2] \times Y.R_2$)→DW: t'_2。

④ answer(($[2,5] \times Y.R_2$)→DW: t_3；

⑤ query(($[2,5] \times Y.R_2 \times Z.R_3$),DW)→$Z$: t_4；

⑥ commit(($[2,5] \times Y.R_2 \times Z.R_3$),DW)→$Z$: t_5。

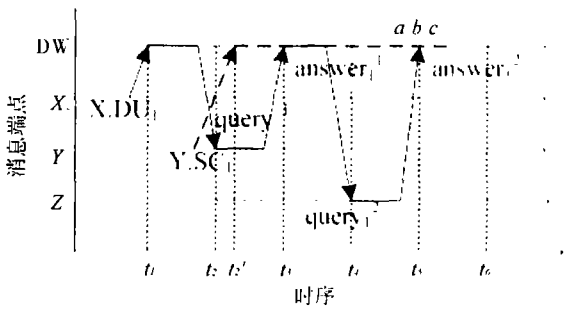

图5 混合并发更新的时序图

Fig.5 The temporal relation of concurrent mixed updates

a，c：代表VA完成的时候；

b：代表视图维护结果的提交时刻

如果视图调整VA在 a 点完成，即 $t_a < t_b$，则视图已重定义，数据更新无法完成提交，t_b 表示 answer$^2_1$ 提交的时刻；如果视图调整VA在 c 点完成，即 $t_b < t_c$，即按顺序维护，则可以正确提交。

定义5 异源混合并发更新：在DW响应更新 $DS_i.DU_U: t_{iu}$ 的过程中，如果 $\exists DS_j.SC_v: t_{jv}, i \neq j$，且：

①更新的时序满足 $t_{DU_{iu}} < t_{SC_{jv}}$；

②视图调整VA满足 $t_{SC_{jv}} < t_{VA_{iu}}$ 或者维护查询的时序满足 $t_{SC_{jv}} < Q^l_{i,m}$。则称 $DS_i.DU_u$ 和 $DS_j.SC_v$ 构成异源混合并发更新，记为 $DS_i.DU_u \Uparrow DS_j.SC_v$。

当定义5中 $i \neq j$ 时，称之为同源混合并发更新。

通过对视图维护过程中的更新异常进行分析，发现并发更新是导致视图维护异常的主要原因。同时，在并发环境下，数据更新和模式更新极有可能同时存在，这对视图维护工作带来极大的困难。为此，在对实例进行分析的前提下，深入而细致地刻画了模式与数据全面并发的典型情形，为以后的视图维护工作奠定了坚实的基础。

参考文献：

[1] AGRAWAL D, ABBADI A, SINGH A, et al. Efficient view maintenance at data warehouses [C]. In: ACM SIGMOD Conference, Teuson, Arizona, USA, 1997:417 - 427.

[2] ZHUGE Y, GARICA-MOLINA H, HAMMER J, et al. View maintenance in a warehousing environment [C]. In: Proceedings of the ACM SIGMOD Internationl Conference on Management of Data, 1995:316 - 327.

[3] ZHUGE Y, GARICA-MOLINA H, WIENER J L. The Strobe algorithms for multi-source warehouse consistency [C]. In: Proceedings of the Conference on Parallel and Distributed Information Systems, Miami Beach, Florida, 1996:3 - 11.

[4] 左亚尧，舒忠梅，潘久辉. 一种高效的视图维护算法[J]. 计算机研究与发展，2003,40(4)：627 - 633.

[5] 舒忠梅,李师贤,左亚尧.并行多维视图维护方法[J]. 计算机研究与发展,2004,41(增刊):1 - 8.

[6] BAMHA M, BENTAYEB F, HAINS G. An efficient scalable parallel view maintenance algorithm for shared nothing multiprocessor machines[C]. In: 10th International Conference on Database and Expert Systems Applications, Springer-Verlag, 1999:616 - 625.

View Maintaining Anomaly Analysis under Concurrent Updating Environment

SHU Zhong-mei[1], *LI Shi-xian*[1], *ZUO Ya-yao*[2]

(1. Department of Computer Science, Sun Yat-sen University Guangzhou 510275, China;

2. Faculty of Computer, Guangdong University of Industry, Guangzhou 5100090, China)

Abstract: Based on the analysis of view maintenance anomaly, the concurrent updates are found to be one of the main reasons that cause view maintenance anomaly. Meanwhile, data update and schema change are possible to coexist under the concurrent environment which brings about many problems to view maintenance. Thus, the typical situations that scheme change mixes with data updates concurrently are analyzed and explored by using examples.

Key words: concurrent update; view maintain; maintenance anomaly