对查询优化重写 QRA算法的一种改进

涂利明[1] 郑 宁[1] 葛瀛龙[2]
[1](杭州电子科技大学计算机学院 浙江 杭州 310018)
[2](杭州电子科技大学软件职业技术学院 浙江 杭州 310012)

摘 要 数据仓库系统中利用物化视图可以提高查询效率,但是,过分使用物化视图重写查询有时不一定能起到提高查询性能这一预期目的。针对 QRA算法的不足并结合左深度处理树技术提出一种改进算法,通过实验结果表明由改进算法优化后查询的性能更优。

关键词 物化视图 查询重写 查询优化 QRA算法

AN IMPROVED QUERY REFORMULATION ALGORITHM FOR QUERY OPTIMIZATION

Tu Liming Zheng Ning Ge Yinglong
[1] (College of Computer Science, Hangzhou Dianzi University, Hangzhou 310018, Zhejiang, China)
[2] (College of Software Technology, Hangzhou Dianzi University, Hangzhou 310012, Zhejiang, China)

Abstract Materialized views can improve the performance of queries in data warehouse system. However, sometimes the performance of queries may not be raised if the materialized views are overused to rewrite queries. Considering the shortage of the QRA algorithm, and with the combination of the technology of left-deep processing tree, an advanced algorithm is proposed. Experiments show that the queries optimized by the advanced algorithm perform better than before.

Keywords Materialized views Query reformulation Query optimization QRA algorithm

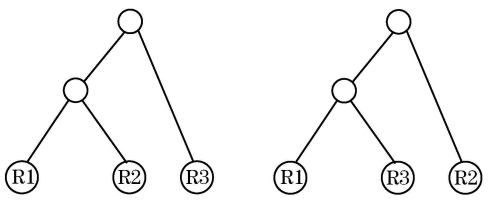
0 引 言

随着数据仓库技术应用越来越广泛,数据仓库中查询使用频率也越来越高,查询性能已成为数据仓库系统一个重要的性能指标。查询优化已经成为数据仓库领域中研究热点之一。文献[1]侧重研究了树结构数据仓库系统查询优化问题。文献[4]给出判断一个查询能否利用物化视图来提高其查询效率的实用算法。文献[3]中 QRA算法的主要思想是尽可能地利用数据仓库中物化视图来优化重写查询,该算法优点是查询时能充分发挥物化视图的作用,但查询利用的物化视图越多查询越复杂,结果是查询代价也相应地增加。因此,仅仅强调利用物化视图优化查询有时并不能真正达到提高查询性能的目的。本文研究了查询优化问题现状,重点分析了 QRA算法,并针对 QRA算法的不足并结合左深度处理树技术提出一种改进的新算法,通过实验结果表明经改进算法优化后查询的性能表现更优。

1 左深度处理树

已知{ R_1, R_2, …, R_m}是关系数据库中的一组关系。只考虑查询中的连接操作,可以用一种二叉树来表示查询过程的连接操作,树中叶子结点代表基本关系,内部结点代表连接操作,这种二叉树称为左深度处理树[2]。整个左深度处理树内部结点的连接操作代价之和,称为连接操作总代价。如果查询子表达式有多个连接操作,则理论上有 P_n^n个不同的连接形式。

假设某查询的子表达式中包含连接操作 $R_1 \infty R_2 \infty R_3$,其中 $|R_1|$, $|R_2|$, $|R_3|$表示关系集的大小,值分别为 20, 30, 40。查询的理论左深度处理树有 6种不同的形式,为了问题的简单,忽略同一内部结点左右子树因相互交换位置而产生的差异。图 1给出了其中同一结构的两种不同连接形式的左深度处理树。图中(a)处理树 $R_1 \infty R_2$的代价 $20 * 30 = 600$,已知 $R_1 \infty R_2$产生 20个结果,那么 $R_1 \infty R_2$与 R_3的连接代价为 $20 * 40 = 800$,($R_1 \infty R_2$)∞R_3的总代价为 $600 + 800 = 1400$。假设(b)处理树 $R_1 \infty R_3$产生 10个结果,那么用同样方法,可以计算得到右边处理树($R_1 \infty R_3$)∞R_2的总代价为 $20 * 40 + 10 * 30 = 1100$。可见虽然这些左深度处理树的结构形态相同,但它们连接形式不同则连接操作总代价也不同。显然,对查询的连接操作进行优化是减少查询代价,提高查询效率的一种有效手段。



(a) (b)

图 1 左深度处理树

收稿日期:2006-05-16。涂利明,硕士生,主研领域:信息处理,数据仓库。

2　QRA算法分析

QRA算法由 FindRecoverViews和 Reformulation两部分组成[3]，其中 FindRecoverViews是整个 QRA算法的关键。FindRecoverViews功能是判断视图缺失属性集 m是否可恢复，如果可恢复则返回用于恢复 m的物化视图集的接连。算法对视图 V能够从物化视图 V_i得到恢复的缺失属性 m_i产生一个属性集 $m_i \cap m$。由此可知，缺失属性集 m中的每一属性 m_i在视图集中至少有一个视图 $V_j(1 \in \in k)$含有该属性，其中 V_j在$\infty[\{V V_1, \cdots, V_k\} \leqslant T]$接连无关。在 FindRecoverViews中求解 viewset是一个难点，因为缺失属性的恢复代价不仅依赖于视图还依赖于数据库本身的属性如索引，数据分布等。如果用 viewset中得到的视图的数量最少来替代视图的代价最小，能减少 QRA算法的搜索空间，此时 QRA算法和贪婪算法的时间复杂度相同，都为 $O(|MV|^2)$，其中 |MV|是物化视图的个数。

Reformulation第一步判断输入的查询 Q和视图 V是否满足 QRA算法要求满足的条件 1和条件 2。如果满足就能保证找到一个选择谓词集 $S_{1V}(Q')$。第二步判断是否满足条件 3。再由 FindRecoverViews来测试 m是否能通过其它物化视图集来得到恢复，如果可恢复，则最后通过 $P(Q')$，$S(Q')$，$J(Q')$来获得重写后的查询 Q'，否则返回 False。

从以上分析可知，如果能保证算法最后得到的结果查询代价最小则能提高查询效率。而 QRA算法没有考虑优化后查询的代价是否为最小。针对此不足，结合左深度处理树中连接操作优化技术，本文提出一种改进的 QRA算法用于优化查询，进一步改善数据仓库系统查询性能。

3　MQRA算法

3.1　算法描述

假定查询和视图都不带集合操作。MQRA(Mend Query Reformulation)是一种基于 QRA的改进算法，MQRA算法框架如下：

```
Mend Reformulation(Q, M)
 input: Q and V are a query and a view defined in (1)
 output: minimum cost of Q''
 BEGIN
    For each V∈MV
    if Reformulation(Q, V_i) then
        比较查询 Q和 Q'两者的代价;
        Q_temp=查询总代价小者;
      else Q_temp=Q
    if Q_temp存在多个连接操作 then
      对于 Q_temp语句子表达式中的连接操作进行优化,
      保留查询结果与 Q_temp相等的查询,得到一个扩
      大的查询候选集。
      计算候选集中的各个查询的总代价;
        Q''=查询总代价小者;
    else Q''=Q_temp
    return Q'';
 END
```

第一步，对于指定的查询 Q，物化视图集 MV中的每个视图 V_i通过 QRA算法来寻找利用物化视图重写后的查询 Q'。如果找到 Q'，比较 Q和 Q'两者的总代价，代价小者给 Q_{temp}，如果没有找到 Q'，则查询 Q赋给 Q_{temp}。第二步，如果上一步得到的 Q_{temp}子表达式存在多个连接操作，则利用左深度处理树的连接操作优化方法对连接操作进行优化，取与原查询 Q等价且总代价最小的候选查询赋给 Q'。如果 Q_{temp}不存在多个连接操作，则直接把 Q_{temp}赋给 Q''。最后返回优化后的查询 Q''。

3.2　算法分析

MQRA算法具有以下优点：(1)由于 QRA算法只考虑利用物化视图来重写查询，而没有考虑利用的物化视图越多查询变得越复杂，查询所花费的代价就越大，优化后查询代价可能不是最小。针对此种情况，MQRA算法增加了对优化后查询 Q'与原有查询 Q的比较，取其优者，从而很好地弥补了原 QRA算法的不足；(2)对查询的连接操作进行优化，从而更进一步地提高了查询的性能；(3)保证了优化后查询的查询总代价最小。

当然 MQRA算法与原 QRA算法相比较，如果查询子表达式有多个连接操作时，MQRA算法对连接操作优化需要额外花费一定的时间，连接操作数越多所花费的时间越长。

4　实验结果

按照这种方法，我们在 CPU Intel(R) 2.66GHz、1G RAM、Windows2000 professional的平台上取某公交的数据仓库系统中 20个查询进行模拟实验。实验 1对查询分三种情况进行比较分析：(Ⅰ)对原查询不进行任何的优化处理；(Ⅱ)利用 QRA算法对原查询进行优化；(Ⅲ)采用 MQRA算法对原查询进行优化。按查询数的不同对查询时间进行比较，实验结果如表 1所示。从表 1可以看出，利用算法 QRA和 MQRA优化后的查询明显比没有优化的查询所花费的时间少，且 MQRA查询花费时间比 QRA减少了 15%左右。

表 1　实验 1查询时间(单位：s)

查询情况 / 查询个数	Ⅰ	Ⅱ	Ⅲ
5	68.48	47.66	40.23
10	170.82	134.24	114.59
15	396.56	302.82	252.68
20	634.78	511.30	415.07

实验 2对 QRA和 MQRA算法中不同连接操作数对运行时间的影响进行对比测试，实验结果如图 2所示。

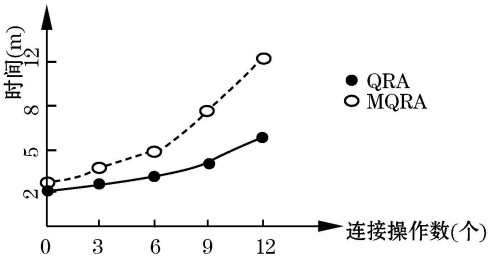


图 2　连接操作个数对算法运行时间的影响

该图表明当查询子表达式中少于 3个连接操作时，MQRA算法运行时间与原来 QRA算法运行时间相差无几。当查询连

(下转第 254页)

分发成功后策略服务器将设备信息写入活动设备库。

管理员采用基于口令的认证方式，认证之后也将获取他和策略服务器之间的数据加密密钥，以对通信数据进行加密。

（3）创建策略

将管理员从安全策略控制台输入的策略在后台转换成Ponder策略并存入域服务器。

（4）分发和执行策略

当有事件触发自管理策略时，则根据相应的自管理行为，对该自管理策略的关联策略执行分发、启用（即执行）、禁用、卸载或发布等自管理措施。若执行分发操作，则将关联策略分发到相应的策略执行代理；若执行启用操作，则进入该关联策略的执行阶段。

（5）自适应调整策略

启用关联策略的同时启用该关联策略的自适应策略，自适应策略在事件监控器注册关心的事件，事件触发后自动调整关联策略。

3 系统的状态转换

为确保管理员远程管理该系统，保证策略服务器和安全设备间策略的安全传输，系统维护了合法管理员集、策略集、活动设备集和策略通信集四个数据集。系统状态转换如图7所示。

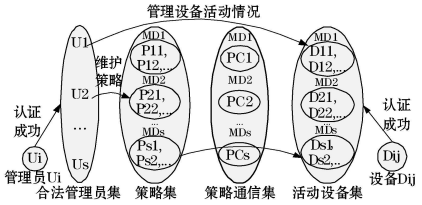


图7 系统的状态转换

假设有 s个管理员维护该系统，Ui（i=1，2…，s）表示第 i个管理员；MDi（i=1，2…，s）表示管理员 Ui对应的管理域，MD i是一个逻辑概念，它可以是多个管理域的并集；MDi内被管的安全设备有 Mi（i=1，2…s）台，策略有 Ni条（i=1，2…s）；Dij（j=1，2…，Mi）表示管理域 MDi内的第 j台安全设备；Pik（k=1，2…，Ni）表示 MDi内的第 k条策略；PC i是三元组（Pik，Dij，Status）的集合，它表示 MDi中策略和设备的映射关系，Status表示策略 Pik当前的状态。

设备认证成功后以管理域为单位记入活动设备库；管理员认证成功后记入合法管理员集，合法的管理员可以维护其管理域内活动设备的情况和设备上的策略信息。

策略集以管理域为单位定义了管理员管辖范围内的所有策略。

策略通信集定义了策略和活动设备之间的映射关系，这种映射关系决定了策略的状态，包括休眠态、待激活态、激活态和挂起态四种。策略刚生成时处于休眠态；策略从策略服务器或策略发布服务器分发到被管设备，但尚未执行时处于待激活态；策略翻译后装入设备内核的状态为激活态；从内核卸载策略后处于挂起态。通过提取策略通信集的数据，可以监控策略的当前状态。

4 结 论

本文设计的自适应的策略管理系统具有以下特点：

- 支持基于管理域的远程移动管理；
- 解决了管理员和安全设备进入系统前的身份认证问题；
- 利用加密技术解决了策略安全传输问题；
- 通过灵活的事件触发，执行相应的自管理操作，并能够根据监控到的事件自动调整已经部署的策略，具有良好的动态自管理和自适应能力。

参 考 文 献

[1] John Strassner Walter Weiss etc. draft ietf-policy-framework-00[EB/OL]. http://www.ietf.org/html.charters, 2003.

[2] Nicodemos Damianou, Naranker Dulay, Emil Lupu, et al. Ponder: A Language for Specifying Security and Management Policies for Distributed Systems-Version 2.3[R]. Research Report DoC 2000/1.

[3] Dulay N, Lupu E, Sloman M. A Policy Deployment Model for the Ponder Language[C]. In Proceedings of the 7th IFIP/IEEE International Symposium on Integrated Network Management (IM'2001), Seattle, Washington, 2001: 529-543.

[4] Francois BARRERE, Abdelmalek BENZEKRI. A multi-domain security policy distribution architecture for dynamic IP based VPN management. In Proceedings of the Third International Workshop on Policies for Distributed Systems and Networks, 2002: 224-227.

[5] 吴蓓，陈性元，赵亮. 策略管理模型研究与改进[J]. 计算机工程.

（上接第245页）

接操作增加到6个以上时，MQRA算法运行时间明显增加。

5 结 论

数据仓库系统中利用物化视图可以提高查询效率。但是，仅仅强调利用物化视图来优化查询，有时并不一定能达到提高查询性能的目的。文中对此问题进行研究，提出一种新算法（MQRA算法）用于解决此问题。从前面对 QRA和 MQRA算法的分析可知，改进后的算法保证了得到的结果查询代价最小，更具一般性。通过实验验证了前面分析的正确性，在实际应用中经 MQRA算法优化后的查询总体性能表现良好。当然 MQRA算法还可以再进一步地完善，比如对算法中多个连接操作优化效率的进一步提高以及除连接操作外其它操作的优化等等，这些都是今后研究工作的方向和重点。

参 考 文 献

[1] Dalamagas T, Koufopoulos A, Theodoratos D, Oria V. Evaluation of queries on tree-structured data using dimension graphs[J]. Database Engineering and Application Symposium[C]. 9th International, 25-27, 2005(7): 65-74.

[2] Chuan Zhang, Xin Yao, Jian Yang. An evolutionary approach to materialized views selection in a data warehouse environment[J]. Systems, Man and Cybernetics, Part C, IEEE Transactions, 2001, 31(3): 282-294.

[3] Chang Jae Young, Lee Sang Goo. An extended query reformulation technique using materialized views[C]. Database and Expert Systems Applications, 1998. Proceedings. Ninth International Workshop on, 26-28, 1998(8): 931-936.

[4] Jonathan Goldstein, Per-Åke Larson. Optimizing queries using materialized views: a practical, scalable solution[C]. Proceedings of the 2001 ACM SIGMOD international conference on Management of data, SIGMOD'01, 2001, 30(2).