用户兴趣驱动的冰山数据立方体构建及更新方法研究

高雅卓 倪志伟 郭峻峰 胡汤磊
(合肥工业大学管理学院 合肥 230009)
(合肥工业大学过程优化与智能决策教育部重点实验室 合肥 230009)

摘 要 为了解决数据立方体完全物化占用过多存储空间的问题,以用户兴趣度为依据,从用户查询的实际情况出发,首次提出在矩阵基础之上进行冰山立方体构建的方法MICC,并在此基础上提出冰山立方体的增量式更新方法ICIU,以解决当用户兴趣发生改变时,需要物化的方体发生改变的问题。实验表明,MICC能够大大节省存储空间,有效支持用户查询,且利用增量方法ICIU能够使构建冰山立方体的效率大大提高。

关键词 兴趣度,冰山立方体,频繁方体,增量更新
中图法分类号 TP274.2 **文献标识码** A

Research on Interest-driven Iceberg Cube Construction and Incremental Update Method

GAO Ya-zhuo NI Zhi-wei GUO Jun-feng HU Tang-lei
(School of Management, Hefei University of Technology, Hefei 230009, China)
(Key Laboratory of Process Optimization and Intelligent Decision-making of Ministry of Education, Hefei University of Technology, Heifei 230009, China)

Abstract This paper proposed a matrix-based iceberg construction method called MICC, in order to reduce the space Consumption during the full materialization for a data cube. Users' interests are considered as the standard to construct an iceberg cube. On the basis of the result of MICC, the paper proposed an incremental update method to dynamically update the iceberg cube while the users' interests are changing. An experiment proves the efficiency and accuracy of these two methods MICC and ICIU.

Keywords Interest, Iceberg cube, Frequent cuboid, Incremental update

1 引言

数据立方体是进行联机分析处理(OLAP)和联机分析挖掘(OLAM)的数据基础。为了确保联机分析操作的快速响应,通常需要对数据立方体进行物化。然而OLAP和OLAM操作面对的是海量、多维的数据,使得数据立方体物化需要耗费大量的计算时间和存储空间。因此,如何在保证OLAP和OLAM操作执行效率的基础上选择有效的数据立方体物化方法来减少系统开支,就成为许多学者关注的问题。

针对这个问题,M. Fang, N. Shivakumar 等(1998)[1]提出了冰山立方体这个概念,其成为数据立方体部分物化的主要方法之一。K. Beyer, R. Ramakrishnan(1999)[2]随后提出了自底向上构建冰山立方体的BUC方法,在构建的过程中采用Apriori性质进行剪枝。D. Xin, J. Han 等(2003)[3]提出了自底向上与自顶向下相结合构建冰山立方体的Star-Cubing方法。

数据立方体的物化视图选择也是普遍采用的一种对数据立方体进行优化的方法。V. Harinarayan, A. Rajaraman 等(1996)[4]首先提出用优化算法解决一定空间条件限制下物化方体的选择问题;J. Yang, K. Karlapalem 等(1997)[5]与C. Zhang, X. Yao 等(2001)[6]将遗传算法获取最优解的能力用于最优物化集的选择;P. Kalnis, N. Mamoulis 等(2002)[7]采用随机算法的思想设计了选择物化视图的算法。

本文正是基于冰山立方体部分物化以及数据立方体优化的思想,研究数据立方体的构建方法,受到牛小飞、石冰等(2004)[8]基于矩阵的关联规则挖掘方法的启发,提出以用户兴趣为驱动,在矩阵的基础之上,采用简单的数组和位运算,记录被查询的方体频数,选择部分方体进行物化,被选择的条件为查询频率超过设定的用户兴趣度。

由于对数据立方体的OLAP查询是不断增多的,且用户的兴趣是可能发生变化的,从而满足兴趣度条件的方体也是随之不断变化的。在这种情况下,若要对查询记录进行重新遍历,将耗费大量的系统资源,并且当历史数据发生丢失时将大大影响新的数据立方体构建结果的准确性。针对这个问

到稿日期:2009-01-15 返修日期:2009-04-03 本文受国家863高技术研究发展计划基金项目(2007AA04Z116),国家自然科学基金项目(70871033)资助。

高雅卓(1984—),女,博士生,主要研究方向为数据挖掘、OLAP及商务智能,E-mail:yazhuogao@163.com;倪志伟(1963—),男,教授,博士生导师,主要研究方向为人工智能、数据挖掘、商务智能;郭峻峰(1983—),男,硕士生,主要研究方向为OLAP及商务智能;胡汤磊(1984—),男,博士生,主要研究方向为数据挖掘、联系发现。

题，本文在基于矩阵的立方体构建方法的基础之上，提出了一种冰山立方体的增量更新方法，该方法充分利用更新之前的方体，不需要重复扫描查询记录集。

实验结果表明，以用户兴趣驱动的基于矩阵的冰山数据立方体构建方法是可行且高效的；在其基础上进行的增量式更新方法是新颖的，并且该方法大大提高了方体物化的效率，能够很好地支持数据立方体的动态更新。

2 相关概念

定义 1(数据立方体 Cube) 将数据立方体 Cube 表示为 $C=(D,M)$，其中 $D=\{D_1,D_2,\cdots,D_n\}$ 为维的集合，$M=\{M_1,M_2,\cdots,M_m\}$ 为度量属性的集合，其中 n,m 分别为维和度量属性的个数。

定义 2(方体 Cubiod) 不同维的任意组合称为方体 Cuboid，i 维方体的集合记为 $CU_i(i=1,2,\cdots,n)$，如 $CU_1=\{D_1,D_2,\cdots,D_3\}$ 为所有可能的一维方体集合；$CU_2=\{(D_1,D_2),(D_1,D_3),\cdots,(D_{n-1},D_n)\}$ 为所有可能的二维方体集合；依此类推，$CU_n=\{(D_1,D_2,\cdots,D_n)\}$ 为可能的 n 维方体集合。

定义 3(单元 Cell) 由 i 维方体中某维度值及其对应的度量值组成的集合称作 i 维单元，表示为 $Cell=\{d_1,d_2,\cdots,d_i,m_1,m_2,\cdots,m_j\}$，其中 d_i 表示维 D_i 的取值，m_j 表示度量属性 M_j 的取值。

若将一个 n 维 Cube 完全物化，则包含的方体个数为 2^n 个。可以看出，随着维数 n 的增加，方体个数以指数形式增长，系统空间开销的增加是惊人的。而实际上，用户可能对许多方体都是不感兴趣的[9]，OLAP 查询也往往集中在某些方体之上。因此，本文以查询频数为依据，设定用户兴趣度，建立冰山立方体，其中包含的方体仅占 Cube 中极小的一部分，大大降低了存储空间。

定义 4(兴趣度 I) 若用户对某方体的查询次数/系统总的查询次数$\geq I(0<I<1)$，则称该方体满足兴趣度 I。

定义 5(冰山立方体 Iceberg Cube) 满足给定兴趣度 I 的那部分方体组成的 Cube 称作冰山立方体 Iceberg Cube，表示为 IC。在这里，I 为构建冰山立方体所需满足的冰山条件。

定义 6(频繁 k-方体) 若 k-方体在查询过程中出现的频率满足设定的用户兴趣度 I，则称该方体为频繁 k-方体。

3 基于矩阵的 Iceberg Cube 构建方法 MICC

3.1 MICC 方法思想

本文以用户兴趣度为出发点，建立起一个冰山立方体，该方体由用户查询经常涉及到的方体，即频繁 1-方体，频繁 2-方体，…，频繁 n-方体所组成。整个过程建立在由 0，1 组成的矩阵基础之上。

首先，以查询次数为行数，以维的个数为列数，为用户查询建立矩阵 QM。其中每个行向量代表一次查询信息，其长度为事务数据表中维的个数。查询所涉及到的维标记为 1，未涉及维标记为 0。被查询事务记录和查询矩阵如表 1 和图 1 所示。

表 1 被查询事务记录(以 5 维为例)

Query	D(事务数据表中的维度，共 5 个)
1	D1,D2,D3,D5
2	D2,D3
3	D1,D5
4	D2,D5
5	D1,D2,D3
…	……

	D1	D2	D3	D4	D5
Q1	1	1	1	0	1
Q2	0	1	1	0	0
Q3	1	0	0	0	1
Q4	0	1	0	0	1
Q5	1	1	1	0	0
…	…	…	…	…	…

图 1 查询矩阵 QM

设定用户兴趣度 I，以矩阵 QM 中的列向量为单位，分别计算每个列向量中“1”的个数，所得结果即为每个维在所有查询中出现的次数。若查询频率超过 I，则该列向量所对应的维度为频繁 1-方体。频繁 1-方体的个数记为 N。

根据经典 Apriori 理论可知，非频繁方体的超集必定是非频繁方体。因此，由 $(k-1)$-方体计算频繁 k-方体时，只需考察频繁 $(k-1)$-方体，而忽略非频繁 $(k-1)$-方体。根据这个理论，以频繁 1-方体为基础，产生频繁 2-方体。建立一个 $(N-1)*N$ 的新矩阵 DRM 和一个 $N*1$ 向量 DRV，DRM 用以反映各个维度与其之前的所有维度之间的关系，两两进行与运算，若所得“1”的个数(即两维度组合的查询次数)超过给定兴趣度 I，则在矩阵 DRM 相应位置记为 1，否则记 0。DRV 则用来记录 DRM 每列中“1”的个数(即位于某维度之前的与之组合为频繁查询组合的维度个数)。矩阵 DRM 和向量 DRV 如图 2 和图 3 所示。

	D1	D2	D3	D5
D1	0	1	1	1
D2	0	0	1	1
D3	0	0	0	0

图 2 矩阵 DRM

D1	0
D2	1
D3	2
D5	2

图 3 向量 DRV

在频繁 2-方体的基础上产生频繁 3-方体，依此类推，直至产生频繁 n-方体。在频繁 $(k-1)$-方体的基础上产生频繁 k-方体，参照频繁 $(k-1)$-方体的最后一个维度(即第 $k-1$ 个维度)，若第 k 个维度与其组成的组合为频繁方体，则进行进一步判断。若它同时与第 1 个，第 2 个，直至第 $k-2$ 个维度组成的组合都为频繁方体，则第 k 个维度与频繁 $(k-1)$-方体组成候选 k-方体，频繁 k-方体就从这些方体中产生。对候选 k-方体中的维度对应的列向量进行与运算，若得到的“1”的数量满足给定兴趣度 I，则此方体为频繁 k-方体。这一系列运算过程都在矩阵 QM，DRM 及向量 DRV 基础上进行。

根据得到的频繁 1-方体，频繁 2-方体，…，频繁 n-方体建立冰山立方体 IC，涉及这些方体的查询单元都加以物化，方便用户查询，并支持相应的 OLAP 操作。得到的方体由于都为满足一定用户兴趣度的方体，因此涵盖了大部分用户查询，使得系统能够对用户查询做出快速响应。

3.2 MICC 算法设计

算法 1 Matrix-based Iceberg Cube Construction

输入：事务数据集 TDB，查询矩阵 QM(由对 TDB 的查询记录组成)，用户兴趣度 I

输出：冰山立方体 IC，每个频繁方体出现次数的记录集合 SUM

过程：

```
Function ConstructMICC(TDB,QM,I)
Rcount=rows number of QM
For each Di∈D
    If sum of Di≥Rcount * I
      FreqM1(1,k)=Di,FreqM1(2,k)=sum(Di),
      put Di into CU1;
      save sum of Di in SUM;
put CU1 and its corresponding TDB Cells into IC;
DRM=CreateDRM(FreqM1),DRV=CreateDRV(FreqM1);
For each Di in FreqM1
    For each Dj in FreqM1 & (j>i)
        If S=sum of QM(ith column) ∧ QM(jth column)≥Rcount
        * I
          DRM(Di,Dj)=1,DRV(Dj,1)=+1,put
          (Di,Dj) nto CU2,save S in SUM;
Put CU2 and its corresponding TDB Cell into IC;
.....
For each frequent k- cuboid(D1,D2,…,Dk)
    If DRM(Dk,Dk+1)=1 & DRV(Dk+1,1) ≥k &
    DRM(D1,Dk+1)=…= DRM(Dk-1,Dk+1)=1
        put (D1,D2,…,Dk,Dk+1) into
        candidate-(k+1)- cuboid;
        If S=sum of QM(1th column) ∧ … ∧
        QM(k+1th column)≥Rcount * I
            put (1,D2,…,Dk,Dk+1) into CUk+1,
            save S in SUM;
put CUk+1 and its corresponding TDB Cell into IC;
```

通过以上算法，建立了冰山立方体 IC，组成它的频繁 1-方体，频繁 2-方体，…，频繁 n-方体都是用户在实际查询中频繁使用到的。当面对多维海量数据时，用户查询往往集中在一部分方体当中，因此根据用户兴趣建立起来的冰山立方体在大部分情况下是能够满足为用户查询提供快速响应的，并且大大节省了存储空间。

以用户兴趣驱动的基于矩阵的 MICC 方法的优势在于，其支持增量更新，能够根据用户兴趣的变化动态地更新被物化的方体，并且不需要重新遍历原始的查询记录集。

4 冰山立方体增量式更新方法 ICIU

4.1 ICIU 方法思想

在上节 MICC 算法及所得结果的基础之上，对冰山立方体 IC 进行增量更新，使物化方体能够适应用户兴趣的变化。方法基本思想如下：

设新增加的查询矩阵为 qm。易知，若某方体在 QM 与 qm 中都为非频繁的，那么其在 QM+qm 组成的新的查询记录集中必然是非频繁的。因此，只需在由 QM 和 qm 分别产生的频繁方体中确定新的频繁方体，进行物化。

首先使用 MICC 算法，对 qm 进行处理，找出满足最小支持度阈值的频繁 1-方体，频繁 2-方体，…，频繁 n-方体，与由 QM 产生的频繁方体进行比较，将出现以下 3 种情况：

(1)在 QM 和 qm 中均为频繁方体。这样的方体必然是新的频繁方体，应保留在已经物化的冰山立方体 IC 中。

(2)在 QM 中为频繁方体，而在 qm 中为非频繁方体。这种情况下，需要考察该频繁方体在 qm 中出现的次数(即向量中“1”的个数)，与 QM 中出现的次数(MICC 算法中已经记录)相加，所得结果与兴趣度阈值进行比较。满足条件的，则选定为新的频繁方体，保留在 IC 中；否则为非频繁方体，应从 IC 中去除。

(3)在 qm 中为频繁方体，而在 QM 中为非频繁方体。这种情况下，需要考察该频繁方体在 QM 中出现的次数，与 qm 中出现的次数相加，所得结果与兴趣度阈值进行比较。满足条件的，则选定为新的频繁方体，添加到 IC 中；否则为非频繁方体，应从 IC 中去除。

至此，新的冰山立方体 NewIC 产生，它通过在 IC 中添加或去除一些方体得来，表示当新的查询产生时满足用户兴趣度从而需要物化的方体的集合，整个过程不需要重新遍历原查询记录集，是一种增量式的冰山立方体更新方法。

4.2 ICIU 算法设计

算法 2 Iceberg Cube Incremental Update

输入：事务数据集 TDB，增量查询矩阵 qm(由对 TDB 的新增查询记录组成)，用户兴趣度 I，原冰山立方体 IC，每个频繁方体出现的次数 SUM

输出：新的冰山立方体 NewIC(即修改过的 IC)

过程：

```
Function ConstructICIU(TDB,qm,I,IC,SUM)
[IC,SUM']= [{CU1',CU2',…,CUn'},SUM']
          =ConstructMICC(TDB,qm,I);
For i from 1 to n
    For each frequent i- cuboid FC in CUi
        If FC in CUi'
          Remain FC and its corresponding TDB Cells in CUi;
        Else if SUM. FC + SUM'. FC ≥Rcount * I
            Remain FC and its corresponding TDB Cells in CUi;
        Else delete FC and its corresponding
        TDB Cells in CUi;
For i from 1 to n
    For each frequent i- cuboid FC' in CUi'
        If FC' not in CUi &SUM. FC'+SUM'. FC'≥
        Rcount * I
          Add FC' and its corresponding TDB Cells in CUi;
    Else if FC' not in CUi & SUM. FC'+
    SUM'. FC'< Rcount * I
      Delete FC'and its corresponding TDB Cells in CUi;
NewIC={CU1,CU2,…,CUn};
```

通过以上算法，将新增查询记录中的频繁方体与使用 MICC 方法得到的频繁方体进行比较，根据得到的结果判断新的立方体中应包含的频繁方体，使方体得到增量更新。

5 实验及结果

本文实验部分采用模拟方法获得数据。假设用户对各维度的查询兴趣是递增的，采用二项分布函数随机生成实验数据，并用同样的方法生成测试数据。实验环境中的硬件平台为 Pentium(R) 2.93GHz CPU，1.46GB 内存，操作系统为 Windows XP Professional 的计算机，算法采用 Matlab 7.0 实现。

经实验证明，使用 MICC 方法建立起的数据立方体比完全物化大大节省了存储空间，且随着数据维数的增加这种效果越发明显。图 4 显示的是当维度总数为 12 时分别用完全物化方法和 MICC 方法建立起来的立方体中包含 1～12 个维

度的方体个数。由于 MICC 方法是由用户兴趣驱动的，充分考虑到了用户实际的查询通常是集中的，且一般极少同时以过多维度作为查询条件的情况，因此建立起来的数据立方体得到了极大的压缩。包含不同维度数的方体个数可以比较直观地从图 4 中看出(此时 $I=5\%$)。表 2 显示的是当维度数不断增大时使用两种方法物化的方体总数。当维度为 n 时，完全物化的方体数为 2^n，每当增加一个维度时，方体个数呈指数形式增长。图 5 显示的是随着维数的增加使用 MICC 方法得到的立方体被压缩的倍数。可见，维数越多，使用 MICC 方法得到的压缩效果越趋于明显。该方法对多维数据有很好的适应性。

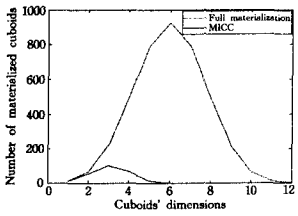

图 4 不同方法下各维物化方体数

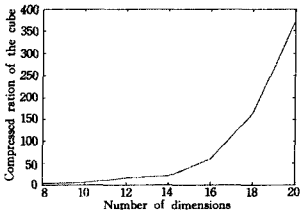

图 5 不同维度总数下 MICC 压缩倍数

表 2 不同维度数情况下两种方法物化的方体总数

物化方法 \ 维度数	8	10	12	14	16	18	20
完全物化方法	256	1024	4096	16384	65536	262144	1048576
MICC 方法	72	186	252	791	1107	1589	2812

数据立方体物化的目的是满足 OLAP 查询的快速响应。由于 MICC 方法建立起来的数据立方体以用户的实际兴趣为基础，使得用户查询基本集中在物化方体的范围之内，较高程度地保证了查询请求执行的效率。图 6 是当支持度在 0～5%之间变化时物化方体对查询的覆盖率。可以看出，覆盖率保持在 70%以上，且 I 越小，覆盖率越高。兴趣度可以根据实际的需求加以调整。

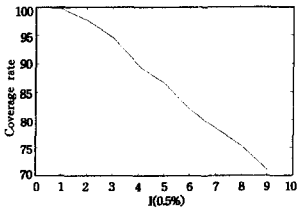

图 6 物化方体对查询的覆盖率

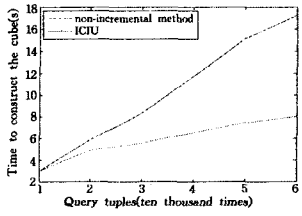

图 7 建立数据立方体耗用时间比较

MICC 方法的一个优势在于，在其基础上设计的 ICIU 方法能够增量地构建数据立方体，实现冰山数据立方体随用户兴趣的变化进行增量更新，避免因历史数据发生丢失而导致新的数据立方体构建不准确的情况，同时大大提高数据立方体构建过程的效率。图 7 显示的是分别使用非增量式的方法和 ICIU 方法建立数据立方体所耗用的时间比较。从曲线趋势可以看出，随着查询次数的不断增加，ICIU 方法越来越显示出优越性，使数据立方体的建立过程所需时间大大减少，从而能够根据用户兴趣的变化动态地更新物化的方体。

结束语 本文将基于矩阵的方法引入冰山数据立方体的构建当中，以用户查询的实际情况为出发点，对满足用户兴趣度的方体进行物化，组成冰山立方体，并且根据用户兴趣的变化动态地更新冰山立方体。实验结果表明，通过 MICC 算法得到的冰山立方体大大节省了存储空间，且能够较大程度地支持用户查询；利用 ICIU 算法进行立方体增量更新，能够大大提高物化效率。

另外，本文冰山立方体的构建借鉴了数据挖掘中的经典 Apriori 理论，与关联规则挖掘具有共同的理论基础，因此可考虑在构建的立方体的基础之上进行关联规则挖掘，将 OLAP 与数据挖掘结合起来，进一步实现数据的有效分析。

参 考 文 献

[1] Fang M, Shivakumar N, et al. Computing Iceberg Queries Efficiently[C]// Proceeding of the 24th VLDB Conference. New York, USA, 1998

[2] Beyer K, Ramakrishnan R. Bottom-up computation of sparse and iceberg CUBEs[C]// Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data. Philadelphia, USA, 1999

[3] Xin D, Han J, Li X, et al. Star-cubing: computing iceberg cubes by top-down and bottom-up integration[C]// Proceedings 2003 VLDB Conference. Berlin, Germany, 2003

[4] Harinarayan V, Rajaraman A, Ullman J D. Implementing data cubes efficiently[C]// Proceeding of the 1996 ACM SIGMOD International Conference on Management of Data. New York, USA, 1996

[5] Yang J, Karlapalem K, Li Q. Algorithms for materialized view design in data warehousing environment[C]// Proceeding of the 23rd International Conference on VLDB. San Francisco: Morgan Kaufmann, 1997: 136-145

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

[7] Kalnis P, Mamoulis N, Papadias S. View selection using randomized search[C]// Proceeding of ACM SIGMOD International Conference on Management of Data. Amsterdam: Elsevier Science Publishers, 2002: 322-333

[8] 牛小飞，石冰，卢军，等. 挖掘关联规则的高效 ABM 算法[J]. 计算机工程，2004，30(11)：118-120

[9] Han Jiawei, Kamber M. Data Mining: Concepts and Techniques (Second Edition)[M]. Beijing: China Machine Press, 2006