数据立方体选择的改进遗传算法

董红斌[1,2] 陈 佳[1]
(武汉大学软件工程国家重点实验室 武汉 430072)[1] (武汉大学国际软件学院 武汉 430070)[2]

摘 要 数据立方体选择问题是一个NP完全问题。研究了利用遗传算法来解决立方体选择问题,提出了一个结合局部搜索机制的遗传算法。这一算法的核心思想在于,首先运用一个基于单位空间最大收益值的预处理算法来生成初始解,然后该初始解经结合了局部搜索机制的遗传算法进行提高。实验结果表明,该算法在寻优性能上优于启发式算法和经典遗传算法。

关键词 查询优化,遗传算法,数据仓库,联机分析处理,视图选择
中图法分类号 TP311 **文献标识码** A

Genetic Selection Algorithm for OLAP Data Cubes

DONG Hong-bin[1,2] CHEN Jia[1]
(State Key Laboratory of Software Engineering, Wuhan University, Wuhan 430072, China)[1]
(International School of Software, Wuhan University, Wuhan 430079, China)[2]

Abstract The data cube selection problem is known to be an NP-hard problem. In this study, we examined the application of genetic algorithms to the cube selection problem. We proposed a genetic local search algorithm. The core idea of the algorithm is as follows. First, a pre-process algorithm based on the maximum benefit per unit space was used to generate initial solutions. Then, the initial solutions were improved by genetic algorithm having the local search of optimal strategies. The experimental results show that the proposed algorithm outperforms heuristic algorithm and canonical genetic algorithm.

Keywords Query optimization, Genetic algorithms, Data warehousing, OLAP, View selection

1 引言

数据仓库技术中,多维分析方法是一种重要的技术,也称联机分析处理OLAP(On Line Analytical Processing)[1]或数据立方体方法。OLAP分析的典型任务就是上卷和下钻。下钻可以通过用户的一系列连续查询从粒度较粗的数据查询到数据细节,从而使用户在多层数据中通过导航信息获得更多的细节性数据;上卷则是下钻的逆操作。因此,OLAP查询通常包含聚合函数,即在SQL语句中含有group-by子句。其查询语句非常复杂,需要对数据进行选择、投影、连接等处理,并且通常要访问上百万条记录,这是一个非常耗时的过程。然而,一个决策支持系统尤其是其即席查询要求能够被快速响应。解决这一矛盾通常采用的一个方法是:预计算并存储这类综合数据(立方体)来加速后续查询。这种综合数据(立方体)就是实视图(Materialized View)。实视图不同于视图的概念,视图只是一个虚表,是一个定义,查询时需要重新计算;而实视图就是用户常用的查询或最可能的查询模式计算出来的结果的物理存储。有了实视图,基本上不再需要对原始数据进行处理,而只需要在实视图的基础上进行一些简单的计算便可以完成复杂的查询。因此,实视图是提高数据仓库查询响应能力从而有效支持决策分析的重要手段。

但是,数据仓库中实视图(立方体)的数量非常巨大,而系统可用空间有限。同时,当基本事实表发生更新时,实视图也需要相应作更新,这就产生了视图维护代价的问题,实化视图数量过多,会导致视图维护代价过高。因此,在实际应用中不可能实化所有视图。OLAP数据立方实化问题可以描述为:给定查询集Q和可用的存储空间量S,选择立方体集M进行实例化,以便在被M占用的存储空间小于S的约束条件下,使得总的查询响应时间和总的维护代价最小。文献[2]已经证明从所有方体中选择合适的视图集进行实化是一个NP难问题。

对于这类问题,一个比较可行的解决方案就是随机优化算法。文献[3]最先提出用遗传算法解决视图选择问题,给出了用于产生MVPP的启发式算法和0-1整数规划算法。但是在视图选择时,没有考虑存储空间的约束。以后的许多相关研究[4-10]都与文献[3]大同小异。经典GA的不足在于:收敛速度慢,得到的解精度不高。而造成这种缺陷的其中一个原因在于遗传算法本身,它是一种更擅长全局搜索而局部搜索能力不足的概率算法。另外,遗传算法作为随机算法具有较强的通用性,无需利用问题的特殊信息,这固然是遗传算法

到稿日期:2009-12-28 返修日期:2010-03-16 本文受国家自然科学基金(60573038)资助。
董红斌(1964-),女,教授,博士生导师,主要研究方向为数据库等;陈 佳(1982-),女,博士生,主要研究方向为数据库等,E-mail:chenjiawh@sina.com。

备受青睐的优点，但同时造成了对问题已知信息的浪费。因此，在本文中，用预处理算法来生成初始解，并且结合局部搜索机制对经典 GA 进行优化。实验表明，本文提出的算法在寻优性能上优于启发式算法和经典 GA 算法。

2 相关工作

1996 年，V. Harinarayan[2] 等人发表的文章中，研究了数据仓库中数据立方体的实视图选择问题，提出了包含级联的多维复合依赖格结构，给出了基于维依赖格的代价、收益模型，在此基础上提出一个贪心算法来选出一个比较优的视图集进行实化。在文章中，作者阐明了战略性地选择实视图可以产生相当大的收益。这篇论文为后来的相关研究奠定了基础。文献[11]延伸了他们的工作，提出了索引选择；文献[12]将数据立方体的格模型形式化表示，提出了在立方体选择处理前减少空间的算法；文献[13]提出了只依赖立方体大小的贪婪算法，算法和 Harinarayan 的贪婪算法有同样的结果，但是更有效。

1997 年，H. Gupta 给出了实视图选择的理论框架，提出在空间限制下使查询响应时间和视图维护代价总和最小的实视图选择代价模型，同时使用贪心算法解决了这个问题[14]。它检查小部分状态空间，使实视图满足空间条件限制的同时达到了时间要求，但这种方法性能不是很好。后来，H. Gupta 又提出在维护代价一定的条件下，使总查询时间最小的代价模型，给出 A* 算法来解决这个问题[15]。S. R. Valluri 根据一个视图的选择可能影响其他视图的利益，从而影响总查询代价和维护代价，定义了视图关联的概念，给出了视图关联矩阵，提出了在视图关联上的代价模型和算法。同时与贪心算法比较，证明在空间限制和修改频率很高时，该算法有更好的性能[16]。

J. Yang 另辟蹊径，给出了一个结构和算法，其基本思想是依据多视图处理计划（Multiple View Processing Plan MVPP），通过合并可以“共享”的公共子视图得到最好的 MVPP，然后从 MVPP 中选择视图实化[17]。后来，M. Indulska 提出这个结构和算法的几点限制：首先，分组属性在查询树中不明确；其次，确认“共享”结果的方法在某些情况下不是总有效的。因此，对此扩展、扩充了查询树的定义，解决了包含不同维、不同粒度上查询共享结果的问题，证明属性级别可提高共享结果的确认，从而提高实视图选择的性能[18]。

另外也有将遗传算法初步运用到实视图选择问题上的[3,6-8,19,20]。文献[8]的实验只包含 8 个视图和 8 个维度，数据集规模很小。文献[3,7]没有用惩罚函数来抑制不可行解数量。在文献[6]中用实验验证了无论维护代价权重如何，遗传算法都表现很好，但当维护权重增加后，所有算法都逐步优化。文献[6]是目前提出的最好的进化方法。

本文提出一种启发式预处理算法来生成初始解，并且结合局部搜索机制对经典 GA 进行优化。实验表明，本文提出的算法在寻优性能上优于启发式算法和经典 GA 算法。

3 视图选择问题的形式化定义

3.1 立方体选择问题

对于视图（立方体）的选择问题（View Selection Problem），最初的动机来源于数据仓库的设计，要决定哪些视图存储在数据仓库中能得到最优的性能。后来几个商业数据库系统提出另一个动机，即支持实视图增量修改，使用实视图加速查询估算。但是，数据仓库中存储的实视图占据着大量的存储空间，而且它们的一致性维护也需要占用大量的 CPU 时间。例如在数据立方体中，如果不考虑维的层次关系，则此时维所对应的坐标轴的值域为 0 和 1。如果数据立方体共有 n 个维，则可能的视图有 2^n（包括空视图）个这种立方体。如果考虑维的层次，则维中的视图个数为 $Num_1 \times \cdots \times Num_n$ 个。其中 Num_i 为 D_i 维层次结构中元素的个数。单个视图分组属性的不同排列视为同一视图。考虑数据立方体中各维的层次结构以后，数据立方体的视图将更大。在存储空间有限、用于维护的视图的 CPU 时间有限，同时又要最大限度缩短 OLAP 查询时间的情况下，对所有视图都进行实化耗费的空间代价和维护代价又是十分巨大的，也是不合理的。因此需要选择部分视图进行实化，这就是实视图选择问题。其形式化描述如下。

定义 1（实视图的选择问题） 给定一个多维数据格 L，其中 L 带有 n 个视图集合 $V=\{V_1,V_2,\cdots,V_n\}$，查询集合 $Q=\{q_1,q_2,\cdots,q_k\}$，查询频度集合 $F_q=\{f_{q1},f_{q2},\cdots,f_{qk}\}$，更新频度集合 $U_m=\{u_{m1},u_{m1},\cdots,u_{mn}\}$ 以及空间限制 S。实视图选择问题是指在 $\sum_{v\in M}|V|\leqslant S$ 前提下，寻找 V 的子集合 M，使得代价函数 $\sum_{i=1}^{k} f_{q_i} E'(q_i,M)+\sum_{j=1}^{n} U_{m_j} E(v_j,M)$ 最小化。其中 $E'(q_i,M)$ 表示的是在视图集合 M 被实化前提下查询 q_i 的查询代价，$E(v_j,M)$ 表示的是在视图集合 M 被实化前提下 v_j 的更新代价。该定义简单地说，就是选择一些视图进行实化，使得所耗代价最小。

3.2 代价模型

本文采用的代价模型主要由两部分组成：查询代价和维护代价。

对于查询代价，在 OLAP 分析中，查询一般都为聚合函数，因此采用文献[3]中的代价模型。下面给出其定义。

定义 1 对于某个结果集，总查询处理代价 $E'=\sum_i (f_{q_i} * Q_{v_i})$。其中 f_{q_i} 为视图 i 的查询频率，Q_{v_i} 为产生结果集的过程中视图 i 的操作代价。不同的视图有不同的代价，总查询处理代价表示为以表及实视图为起点、以结果集为终点的所有操作的处理代价之和。

对于维护代价，由于数据仓库中数据一般不做修改而是不断增加数据，因此只考虑数据仓库 refresh 时数据装载的维护代价。具体定义如下。

定义 2 总维护代价 $E=\sum_i (U_{m_i} * M_{v_i})$，其中，$M_{v_i}$ 代表实视图 i 更新的平均代价，U_{m_i} 代表源关系的更新反映到实视图 i 的变化传播频率。

结合查询代价和维护代价，代价函数如式(1)所示：

$$C(Q,M)=\sum_{i=1}^{k} f_{q_i} E'(q_i,M)+\sum_{j=1}^{n} U_{m_j} E(v_j,M) \quad (1)$$

4 结合局部搜索机制的遗传算法

基于达尔文的进化思想提出的遗传算法，是一种通用的搜索策略和优化方法，特别适用于求解组合优化问题。然而它是一种更擅长全局搜索而局部搜索能力不足的概率算法，因而收敛速度慢，得到的解精度不高。本文利用逆序算子增强局部搜索性能的特点，并且最大限度地减少对全局搜索的

影响，把全局搜索算子和局部搜索算子优化组合，构造一种基于逆序算子的优化组合遗传算法。下面给出结合局部搜索机制的 GA 应用于视图选择问题的过程描述：

(1)利用启发式方法，生成二进制编码的初始化种群，初始全局最优解为 P；

(2)计算种群各个染色体的适应度；

(3)根据排序选择算法对群体进行排序选择，如果本代的最优解优于 P，则 P 被本代最优解替代；

(4)如果属于全局搜索阶段，按全局搜索阶段的交叉和变异算子生成满足条件的新种群；

(5)如果属于局部搜索阶段，按局部搜索阶段的交叉和变异算子生成满足条件的新种群，并且每隔 30 代启动一次逆序局部搜索算子；

(6)判断是否满足优化组合遗传算法的结束条件，如果满足，则输出最终解 P，否则转向步骤(2)；

(7)输出结果。

本算法把搜索过程分为全局搜索和局部搜索两个阶段，在前四分之三代，启动全局搜索；后四分之一代启动基于逆序算子的局部搜索算子，为局部搜索过程；采用每代迭代法来确定最终全局最优解。

4.1 初始解的构造

为了应用遗传算法，我们首先对视图选择问题的解进行染色体编码。每个染色体是由固定数目的二进制(0 或 1)串构成的，每个多维数据格编码成一个二进制串。其中，编码串的长度是多维数据格中的候选视图的数目，字符串中的 0 表示对应的节点(视图或查询)在数据仓库中还没有被实化，字符串中的 1 表示对应的节点(视图或查询)在数据仓库中已经被实化。

在 GA 搜索中，初始解对于最终解的质量起着至关重要的作用。但是大多数的 GA 研究侧重于随机产生初始种群，这不仅需要很长的搜索时间，并且降低了发现最优解的可能性。本文采用启发式产生初始解，即实化最大单位空间收益值的视图，然后用遗传算法进行提高。

4.2 适应度函数

视图选择模型的目标是最小化总的查询响应时间和维护代价，因而本文采用式(1)作为适应度函数。

$$F(G,M)=\sum_{i=1}^{k} f_{q_i} E'(q_i,M)+\sum_{j=1}^{n} U_{m_j} E(v_j,M)$$

式中，$F(G,M)$ 表示适应度函数，$E'(q_i,M)$ 表示总的查询代价，$E(v_j,M)$ 表示总的维护代价。

4.3 遗传算子

因为本文采用全局搜索和局部搜索两个阶段，所以在每个阶段采用的遗传算子略有不同。

在全局搜索阶段，采用对全局搜索有效的均匀杂交方式。均匀杂交方式破坏模式的概率大，在搜索过程中能以较大的概率搜索到点式杂交无法搜索到的模式。

在局部搜索阶段，采用单点杂交模式进行搜索。

另外，由于转盘式选择法不能使传统遗传算法收敛至全局最优解，锦标赛选择策略能避免超级个体的影响，但对局部搜索不利，因此无论在哪种搜索阶段，本文均采用排序选择的方法作为选择算子。

4.4 启发式修补算法

由于 GA 在编码方案中没有嵌入约束条件，而数据仓库的存储空间并不是大得此足够实化所有的视图，因此在进化的过程中(如交叉或变异)可能产生无效的解。所以，本文对不满足约束的染色体采用“启发式”方法修正。例如 1 1 1 0 0 1，该染色体表示实化 $V1,V2,V3,V6$，但不满足空间或其他约束，则对其修正的方法如下：

1)分别计算 $V1,V2,V3,V6$ 的增益值；

2)按增益值大小从大到小依次选择要实化的视图，直到约束条件不满足为止。例如，增益值从大到小为 $V2,V6,V3,V1$；则依次选择 $V2,V6,V3,V1$；若当依次选择 $V2,V6$ 后，再选择 $V3$ 就不满足约束，则染色体变为 0 1 0 0 0 1。

5 实验与分析

为了评价本文提出的视图选择遗传算法，我们把它与启发式算法[3]和经典遗传算法[6]进行了比较。测试环境：硬件平台 P4 3.0GHz，1G RAM；操作系统 Windows 2000。测试数据集来自 TPC-D 基准测试数据[13]，本文采用了其中的 1G 数据库。

本文算法的参数设置如下：最大迭代次数为 200；种群大小为 100；在全局搜索阶段，交叉概率和变异率分别为 0.4 和 0.05；在局部搜索阶段，交叉概率和变异率分别为 0.65 和 0.1。本文考虑 3 种不同的视图查询频率：1)均匀查询频率，2)随机查询频率；3)热查询分布。查询的分布满足 2～8 原则，即 80%的查询量产生于 20%的查询。对于空间约束，考虑 4 种情况，即占所有空间的 5%，20%，50%，90%。图 1 分别给出了相应于以上 3 种视图查询频率的实验结果图。

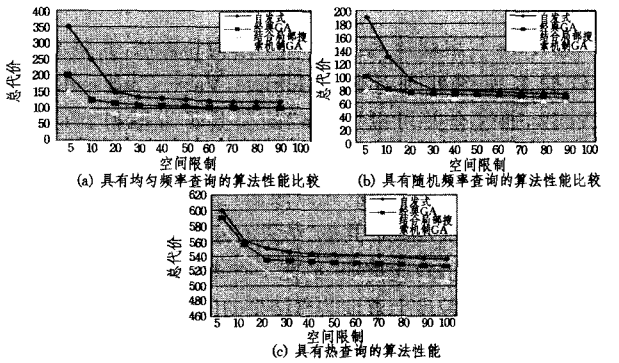

图 1

从图 1 的实验结果可以看出，本文提出的算法在所有情况下，不论视图查询频率如何分布，都优于启发式算法和经典的遗传算法。这恰好说明了我们采用的启发式和局部搜索策略在局部搜索阶段的作用，搜索结果分布相对集中，搜索到的优化值的效率较高，有良好的全局搜索性能和局部搜索性能。

表 1 比较不同视图选择算法的运行时间。本文通过实验来比较在随机查询频率、空间约束为 10%时，它们发现最优解所花费的时间。从实验结果可以看出，随着节点数的增加，所有算法的时间都大大增加，然而本文算法所用的时间增长不大。当节点数大于 16 时，本文算法所用的时间少于启发式算法和经典遗传算法所用的时间；在节点数大于等于 256 时，启发式方法无法处理。

表 1　算法运行时间比较

结点数	结合局部机制的 GA	经典 GA	启发式算法
8	3Sec	2Sec	1Sec
16	3Sec	3Sec	5Sec
32	3Sec	3Sec	50Sec
64	60 Sec	75Sec	300 Sec
256	300 Sec	635Sec	——

综合上述，无论是考虑视图维护代价还是考虑查询响应时间，本文提出的算法都优于启发式算法和经典遗传算法。

结束语　针对实视图选择问题是数据仓库中一个非常重要的问题，本文着重论述了如何选择合适的数据立方体进行实化，以便使总的查询响应时间及视图维护代价最小。并提出了一个结合启发式和局部搜索机制的增强视图选择遗传算法。实验结果表明，本文提出的算法在寻优性能上优于启发式算法和经典遗传算法。

参 考 文 献

[1] Codd E F, et al. Providing OLAP(on-line analytical processing) to user-analysts: an IT mandate[R]. IBM Research Lab, 1993

[2] Harinarayan V, Rajaraman A, Ullman J D. Implementing data cubes efficiently [C]// Proc. 1996 ACM Int. Conf. Management of Data. June 1996: 205-216

[3] Zhang C, Yang J. Genetic algorithm for materialized view selection in data warehouse environments[C]// Mohania M K, Tjoa A M, eds. Proc. of the 8th Int'l Conf. on Data Warehousing and Knowledge Discovery (DaWaK'99). Florence: Springer-Verlag, 1999: 116-125

[4] Lawrence M. Multiobjective genetic algorithms for materia-lized view selection in OLAP data warehouses[C]// Cattolico M, eds. Proc. of the 2006 Genetic and Evolutionary Computation Conf. (GECCO 2006). Seattle: ACM Press, 2006: 699-706

[5] Horng J T, Chang Y J, Liu B J. Applying evolutionary algorithms to materialized view selection in a data warehouse[J]. Soft Computing-A Fusion of Foundations, Methodologies and Applications, 2003, 7(8): 574-581

[6] Yu J X. Yao X, Choi C H, et al. Materialized view selection as constraint evolutionary optimization[J]. IEEE Trans. on Systems, Man and Cybernetics-Part C, 2003, 33(4): 485-467

[7] Zhang C, Yao X, Yang J. An evolutionary approach to materialized view selection in a data warehouse environment[J]. IEEE Trans. on Systems, Man and Cybernetics-Part C, 2001, 31(3): 282-293

[8] Lee M, Hammer J. Speeding up materialized view selection in data warehouses using a randomized algorithm[J]. Int'l Journal of Cooperative Information Systems, 2001, 10(3): 327-353

[9] Wang Z Q, Zhang D X. Optimal genetic view selection algorithm under space constraint[J]. Int'l Journal of Information Technology, 2005, 11(5): 44-51

[10] Lin W Y, Kuo I C. A genetic selection algorithm for OLAP data cubes[J]. Knowledge and Information Systems, 2004, 6(1): 83-102

[11] Gupta H, Harinarayan V, Rajaraman A, et al. Index Selection for OLAP[C]// Proc. ICDE. 1997: 208-219

[12] Baralis E, Paraboschi S, Teniente E. Materialized view selection in a multidimensional database[C]// Proc. of the 23th VLDB Conference. 1997: 156-165

[13] Shukla A, Deshpande P M, Naughton J F. Materialized view selection for multidimensional datasets[C]// Proc. of the 24th VLDB Conference. 1998: 488-499

[14] Gupta H. Selection of Views to Materialize in a Data Warehouse [C]// Proceedings of the 23th VLDB Conference. Athens, Greece, 1997: 156-165

[15] Gupta H, Mumick I S. Selection of Views Maintenance Cost Constraint [C]// Proc. of the 7th Intl. Conf. on Database Theory. 1999: 453-470

[16] Valluri S R, Vadapalli S, Karlapalem K. View relevance driven materialized view selection in data warehousing environment [J]. Australian Computer Science Communications, 2002, 24 (2): 187-196

[17] Yang Jian, Karlapalem K, Li Qing. Algorithm for Materialized View Design in Data Warehousing Environment[C]// VLDB' 97. 1997: 20-40

[18] Indulska M. Shared Result Identification for Materialized View Selection[C]// Proceedings of the 11th Database Conference (ADC). 2000

[19] Horng J T, Chang Y J, Liu B J, et al. Materialized view selection using genetic algorithms in a data warehouse[C]// Proceedings of World Congress on Evolutionary Computation. Washington, DC, July 1999: 2221-2227

[20] Lin W Y, Kuo I C. OLAP data cubes configuration with genetic algorithms[C]// IEEE. 2000: 1984-1989

(上接第 147 页)

图 5 为系统可靠性对比图，图中横坐标为订单流入速度，纵坐标为系统可靠性。

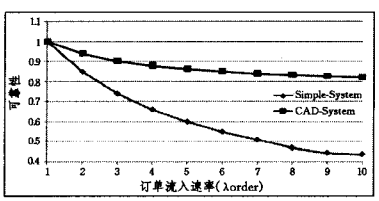

图 5　可靠性对比图

结束语　本文简要介绍了柔性工作流中组合服务的可协调定义及判定方法，并根据可协调定义及判定方法，可以对系统流程进行组合服务的可协调性的定义以及分析，基于组合服务可协调性的定义以及分析方法可以事先对系统中组合服务间的协同合作关系进行分析预测，本文下一步研究的重点是根据该分析可以对组合服务间的协同程度进行评估，然后根据评估结果可以对服务之间的组合进行优化重组，并以此提高系统提供可靠服务的能力，通过可协调性判断的反馈结果给出服务自动组合的方法。

参 考 文 献

[1] Andrew S. Tanenbaum, Computer Networks(3 Editions)[M]. ISBN 7-302-02410-3/TP0. 212

[2] Kumara A, Wainer J. Meta workflows as a control and coordination mechanism for exception handling in workflow systems [J]. Decision Support Systems, 2005, 40(1): 89-105

[3] Bush S F, Kulkami A B. Book: Active Networks and Active Network Management[M]. e-ISBN: 0-306-46981-2

[4] Aura T. Crytographically Generated Addresses(CGA)

[5] 涂序彦. 大系统控制论 [M]. 北京：北京邮电大学出版社, ISBN 756351096