IDEA GROUP PUBLISHING
701 E. Chocolate Avenue, Suite 200, Hershey PA 17033-1240, USA
Tel: 717/533-8845; Fax 717/533-8661; URL-http://www.idea-group.com
ITB12609

This paper appears in the book, *Emerging Trends and Challenges in Information Technology Management, Volume 1 and Volume 2* edited by Mehdi Khosrow-Pour © 2006, Idea Group Inc.

Implementing Real Time OLAP with MDDC (Multi-Dimensional Dynamic Clustering)

Michael W. Martin & Rada Chirkova
North Carolina State University, Raleigh, N.C. 27695, {mwmartin, chirkova}@csc.ncsu.edu

ABSTRACT

Data warehouse applications increasingly must update and maintain data in real time while simultaneously executing efficient OLAP (Online Analytical Processing) queries. The size and corresponding performance challenges of these data warehouses are growing at a staggering rate. This paper introduces, MDDC, a new technique that supports efficient OLAP queries without the extensive use of materialized views or secondary indexes. This technique is dynamic and does not require data reorganization. This is especially important for real-time data warehouses that must continually update data without downtime to reorganize data structures. This technique is similar to MHC (Multidimensional Hierarchical Clustering) but provides better symmetry and is completely dynamic. MDDC could utilize any number of host data structures including B-trees.

INTRODUCTION

Multidimensional data structures are very important for data warehouse and OLAP applications [kimball:98].

MHC

MHC (Multidimensional Hierarchical Clustering) combines Z-ordering and hierarchical clustering. In this section we review these concepts.

Z-ordering

MHC is partially based on the concept of bit interleaving from Z-ordering [orenstein:84,orenstein_:86]. This technique maps multidimensional spaces to one-dimensional spaces. When the Z-ordering curve is mapped in two dimensions it has a distinctive Z shape, hence the name.

The Z-ordering technique shuffles or interleaves the bits from multiple keys together to form one contiguous key. Consider a composite multidimensional key made up of the keys or dimensions A, B, and C. Assume that each key is 3 bits long. Given a set of values for A, B, and C of {101, 110, 001}, a new interleaved key or Z-address takes the first bit from each of the three keys, then the second bit, and finally the third bit from each key so that the resulting single key in interleaved bit format is 110010101.

Formally, a Z-address Z(k) is the ordinal number of a multidimensional composite key k from a tuple or record on the Z-curve and is calculated as follows:

$$Z(k) = \sum_{j=0}^{s-1} \sum_{i=0}^{d} k_{i,j} \bullet 2^{(j \bullet d + i - 1)}$$

The problem with Z-ordering relates to its requirement for a predetermined number of bits in each key coupled with differences in entropy among participating dimensions. A dimension has maximum entropy when it enumerates the maximum number of dimension key values with the minimum number of bits. The maximum entropy E or minimum number of bits for a dimension D with K keys is:

$$E = ceiling(\log_2(K))$$

Consider data with two dimensions D_1 and D_2. If dimension D_1 has 32 key values and a corresponding entropy of 5 bits while D_2 has 100 key values and a corresponding entropy of 32 bits, then dimension D_1 probably will dominate the collating sequence of the B-tree. Accordingly, queries restricting only D_1 will be efficient while those only restricting D_2 will not be efficient. This problem prevents balanced query performance and renders Z-ordering ineffective in producing symmetric access to OLAP data. The Z-ordering technique could use 7 bits for D_2 to improve symmetry but would limit the number of keys in this dimension to 128 without a complete reorganization of the data.

Hierarchical Clustering

MHC is also partially based on hierarchical clustering. Hierarchical clustering is a method that controls key values and structures these key values specifically for queries. Hierarchical clustering capitalizes on the fact that data is often composed of hierarchies. As an example, a customer dimension might contain customers each of which are related to one city. The customer dimension might further assign each city to a state and each state to a region. Hierarchical clustering incorporates these hierarchies into keys and their corresponding indexes. MHC extends this concept to multidimensional data.

To accomplish hierarchical clustering, MHC uses compound surrogate keys. These keys reserve a fixed number of bits for each level in a dimensional hierarchy. The fixed number of bits at each level depends on the number of unique values for all keys or parent keys of that level in the dimension hierarchy. For instance if the customer dimension has 6 regions overall, the maximum number of states in any of the 6 regions is 20, the maximum number of cities in any state is 150, and the maximum number of customers in any city is 17, then the compound surrogate would require 3 bits for the region level, 5 bits for the state level, 8 bits for the city level, and 5 bits for the customer level. Therefore the customer dimension would require a total of 21 bits for each of its primary keys. MHC also makes provision for variable length compound surrogate keys in the event that different keys have different numbers of hierarchical levels or unbalanced hierarchies but, MHC does not make provision for variable length bit strings for each hierarchical level individually. If MHC requires 8 bits for the city level in the customer dimension, then any primary key from the customer dimension that includes the city level must include all 8 bits and no state can contain more than 256 cities [markl:99]. The primary key structure for this customer dimension is shown in **Table 1**.

As just demonstrated, if MHC designates too few bits for a given hierarchy level, then the data might exceed this threshold and require full reorganization of all data structures that include the affected

Table 1: MHC Compound Surrogate for Customer Dimension

REGION	*STATE*	*CITY*	*CUSTOMER*
001	10101	00100001	00011

Table 2: MDDC Compound Surrogate for Customer Dimension

REGION	*STATE*	*CITY*	*CUSTOMER*
10000000	10101000	10000100	11000000

dimension before MHC can process any further updates. If MHC designates too many bits for a given hierarchy level, then some bits of the hierarchy level might remain empty for all data and therefore cause symmetry problems. Similarly, if one parent in a dimensional hierarchy contains 50 children when another parent on the same level in the dimension contains 3 children, then queries searching for children of the parent with 3 children or other parents with less than 50 children might experience symmetry and efficiency problems.

MDDC

MDDC removes the trade-off between symmetry and dynamic updates in MHC with a simple but powerful modification to the MHC key structure and bit interleaving technique. With hierarchical clustering, MHC imposes a partial ordering on data. Since hierarchical clustering already forces a partial ordering that orders data according to the dimensional hierarchies, MDDC further takes advantage of this partial ordering. Research related to MHC has shown that mirroring the bits of each level in compound surrogates improves symmetry in some cases [markl:00]. In addition to mirroring bits, MDDC uses a mechanism for variable length bit strings for each hierarchical level in the compound surrogate. This allows MDDC to be completely dynamic while maintaining the same or better symmetry than MHC.

Variable Length Bit Strings

MDDC allocates the last bit of each byte as a continuation bit so that 7 bits of each byte are usable. Each level in a variable length compound surrogate requires at least one byte and also uses a whole number of bytes for each level. This affects the length of keys that MDDC must store in key fields but does not affect symmetry since MDDC does not use the continuation or filler bits when interleaving bits. It is also important to point out that MDDC might also have an overall variable length byte for compound surrogates just as MHC does to accommodate unbalanced hierarchies.

Dynamic Bit Interleaving

If order preserving bit strings are used, the relative position of these bits in a Z-address must shift. This alters the overall Z-ordering and requires the data to be completely reorganized as bit string lengths change. Using mirrored bits in conjunction with variable length bit strings overcomes this problem. No matter how long the bit strings grow, the relative position of bits in the interleaving does not change. MDDC simply adds the additional bits for larger level values to the end of the bit string for that level in the compound surrogate. This does not alter the relative position of previous keys at any level in the overall bit interleaving and therefore does not require data reorganization. Mirrored bits have one more important advantage. Except for the value of 0, each mirrored bit string ends with a 1 by definition. This allows MDDC to trim the remaining filler bits, which are all zeros, when interleaving bits. With this technique, MDDC always exhibits symmetry as good as MHC and better in some cases. If MHC specifies 8 bits for a given level in a compound surrogate but only uses one, then MHC wastes 7 bits in the Z-ordering. In such cases, some dimensions that are using all bits in their compound surrogates might dominate the dimensions that have unused bits in the collating sequence of the data structure. For the previously listed reasons, MDDC does not include these unused bits in its dynamic bit interleaving process.

The customer example presented in the MHC section has the MDDC format depicted in **Table 2**.

Variable length, mirrored bit strings in conjunction with dynamic bit interleaving provide as good or better symmetry as compared to MHC. In addition, MDDC does not impose any preset limits on the number of distinct key values from any hierarchy level. Thus, MDDC is also completely dynamic unlike MHC.

Queries

MDDC is like MHC in that it only alters the content of dimension keys and the collating sequence in host data structures and does not alter any of its other properties.

Point Queries

Point queries are very simple in MDDC. When the query specifies the full key for each dimension, MDDC simply computes the single address for the complete search key with dynamic bit interleaving as it makes comparisons with other keys in the data structure to locate the record or records with the specified key values.

Range Queries

MDDC uses wild card or "don't care" bits in the dynamic bit interleaving for dimension key values or parts of dimension key values that the query does not specify and then proceeds to query the resulting key values.

Maintenance

MDDC inserts, deletes, and updates records with standard data structure methods with the caveat that MDDC dynamically interleaves the bits for all keys involved in comparisons in the data structure. The interleaved bits determine the placement and organization of records within the data structure. In doing so, MDDC does not in any way alter the properties of the host data structure.

MDDC and B-trees

Like MHC, MDDC works especially well with B-trees. Therefore, B-trees containing keys that the MDDC algorithm organizes retain all maintenance advantages of B-trees including the perfect balance, shallow depth, granular concurrency control, and recoverability. Unlike MHC, MDDC does not require data reorganization. Therefore, it is completely dynamic and can readily accommodate real-time updates.

CONCLUSIONS AND FUTURE WORK

MDDC provides dynamic, real-time updates and efficient symmetry for OLAP queries. MDDC does not require data reorganizations, demonstrates suitability for the most common OLAP queries, and has the capability to reduce the size and number of secondary indexes and materialized views. This paper outlines the basic concepts of MDDC.

In the near future we hope to complete experiments and demonstrate the effectiveness of MDDC.

REFERENCES

Kimball, R., Reeves, L., Margy Ross, M. &Thornthwaith,W. *The Data Warehouse Lifecycle Toolkit*,John Wiley & Sons Inc.,New York, Chichester, Weinheim, Brisbane, Singapore, Toronto, 1998.

Markl, V., Ramsak, F., & Bayer, R. (1999). *Improving OLAP performance by multidimensional hierarchical clustering*, proceedings of ideas'99 in Montreal Canada, IEEE.

Orenstein, J.A. & Merritt,T.H. (1984, April). A Class of Data Structures for Associative Searching, *ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*,pp. 181—190.

Orenstein, J.A. (1986). Spatial Query Processing in an Object-Oriented Database System, *Communications of the ACM*, pp. 326—333.

Mistral, V. M. (2000). Processing Relational Queries using a Multidimensional Access Technique, *Datenbank Rundbrief, 26*, pp. 24-25.