CS412: An Introduction to Data Warehousing and Data Mining **Fall 2014**

Practice Questions

Huan Gui *Due Date: N/A*

Note: This is a set of sample questions with solutions. It aims at helping you better understand the data cube computation methods, especially Multiway Array Aggregation and BUC.

Multiway Array Aggregation

Considering the Example 5.4 on textbook, answer the following questions:

1. Does the memory requirement change if we have different scanning orders of chunks?
2. Does the minimum memory requirement for holding all relevant 2-D planes in chunk memory change if we set mini support to be 2? How about 100? Should we apply multiway array aggregation to iceberge cube computation?
3. What is the *curse of dimensionality* problem? Is multiway array aggregation a good solution for high-dimensional problem?

Answer:

1. Yes. As shown in the example, by scanning the chunks in different orders, the memory requirements differ by 10 times. When choosing the order of aggregation, you need to carefully calculate the memory requirements and select the one that minimizes the memory requirement cost.
2. No. As multiway array aggregation method starts from the base cuboids and progresses upward toward more generalized ancestor cuboids. It cannot take advantage of Apriori pruning, which requires a parent node to be computed before its child. In this case, no matter what the mini support is, the computation cost and minimum memory requirement cost would not change. This conclusion immediately yields the claim that multiway array aggregation cannot be used to iceberge cube computation.
3. The curse of dimensionality is the problem that there exists a huge number of subsets of combinations of dimensions. For multiway array aggregation, the number of cuboids to be computed is exponential to the number of dimensions. To this end, multiway array aggregation is not a good solution for high-dimensional problem.

Bottom-Up Computation

Consider a 3-D data array containing three dimensions A, B, C. The data contained in the array is as follows:

$(a_0, b_0, c_0) : 1$	$(a_0, b_0, c_1) : 1$	$(a_0, b_0, c_2) : 1$	$(a_0, b_0, c_3) : 1$
$(a_0, b_1, c_0) : 1$	$(a_0, b_1, c_1) : 1$	$(a_0, b_1, c_2) : 1$	$(a_0, b_1, c_3) : 1$
$(a_0, b_2, c_0) : 1$	$(a_0, b_2, c_1) : 1$	$(a_0, b_2, c_2) : 1$	$(a_0, b_2, c_3) : 1$

Suppose we construct the iceberge cube for dimension A, B, C with different orders of dimensions.

1. If we set mini support = 4 with the order of A, B, C, how many cuboids would be considered/computed?
2. If we set mini support = 4 with the order of C, B, A, how many cuboids would be considered/computed?
3. Based on results in 1 and 2, which order is better?

Answers

1. If we follow the order of A,B, C, the cuboids that need to be computed are as follows (37 in total):

   All $(*, *, *) : 12$ - expansion

   A $(a_0, *, *) : 12$ - expansion

   AB $(a_0, b_0, *) : 4$ - expansion

   AB $(a_0, b_1, *) : 4$ - expansion

   AB $(a_0, b_2, *) : 4$ - expansion

   ABC $(a_0, b_0, c_0) : 1$

   ABC $(a_0, b_0, c_1) : 1$

   ABC $(a_0, b_0, c_2) : 1$

   ABC $(a_0, b_0, c_3) : 1$

   ABC $(a_0, b_1, c_0) : 1$

   ABC $(a_0, b_1, c_1) : 1$

   ABC $(a_0, b_1, c_2) : 1$

ABC $(a_0, b_1, c_3) : 1$

ABC $(a_0, b_2, c_0) : 1$
ABC $(a_0, b_2, c_1) : 1$
ABC $(a_0, b_2, c_2) : 1$
ABC $(a_0, b_2, c_3) : 1$

B $(*, b_0, *) : 4$ - expansion
B $(*, b_1, *) : 4$ - expansion
B $(*, b_2, *) : 4$ - expansion

BC $(*, b_0, c_0) : 1$
BC $(*, b_0, c_1) : 1$
BC $(*, b_0, c_2) : 1$
BC $(*, b_0, c_3) : 1$

BC $(*, b_1, c_0) : 1$
BC $(*, b_1, c_1) : 1$
BC $(*, b_1, c_2) : 1$
BC $(*, b_1, c_3) : 1$

BC $(*, b_2, c_0) : 1$
BC $(*, b_2, c_1) : 1$
BC $(*, b_2, c_2) : 1$
BC $(*, b_2, c_3) : 1$

C $(*, *, c_0) : 3$
C $(*, *, c_1) : 3$
C $(*, *, c_2) : 3$
C $(*, *, c_3) : 3$

---

2. If we follow the order of C, B, A the cuboids that need to be computed are as follows (12 in total):

All $(*, *, *) : 12$ - expansion

C $(*, *, c_0) : 3$ - no expansion

C $(*, *, c_1) : 3$ - no expansion

C $(*, *, c_2) : 3$ - no expansion

C $(*, *, c_3) : 3$ - no expansion

B $(*, b_0, *) : 4$ - expansion

B $(*, b_1, *) : 4$ - expansion

B $(*, b_2, *) : 4$ - expansion

AB $(a_0, b_0, *) : 4$

AB $(a_0, b_1, *) : 4$

AB $(a_0, b_2, *) : 4$

A $(a_0, *, *) : 12$

---

3. It is easily to see that by order CBA which only needs 12 cuboids to be computed, compared with 37 cuboids computed by order ABC, the computation cost is significantly reduced.
   The BUC performance is sensitive to the order of the dimensions and to skew in the data. Ideally, the most discriminating dimensions should be processed first. Dimensions should be processed in the order of decreasing cardinality. The higher the cardinality, the smaller the partitions, and thus the more partitions there would be, therey providing BUC with a greater opportunity for pruning. Similarly, the most uniform a dimension is, the better it is for pruning.