

***Vmhist*: Efficient Multidimensional Histograms with Improved Accuracy**

Pedro Furtado and Henrique Madeira
CISUC/DEI

Abstract: Data warehouses must be able to process and analyze large amounts of information quickly and efficiently. Small summaries provide a very efficient way to obtain fast approximate answers to complex queries that run for too long. This paper proposes an efficient hierarchical partitioning strategy *vmhist* achieving a large improvement in the accuracy of the summary while maintaining all scalability. This is achieved by pre-computation, localized updating and additivity of the error measures used in the partitioning process. Evaluation reveals that a significant accuracy improvement is obtained for summaries produced with *vmhist* without significant increase in histogram construction time cost.

1. Introduction

Multidimensional histograms provide a very practical way to produce small queryable summaries of the joint distribution of values of multidimensional data sets such as those used in OLAP. The most important metrics to evaluate the quality of algorithms used for the construction of histograms are the accuracy of the summary when answering queries and the scalability of the algorithm (the algorithm must be useful to reduce very large data sets). In fact, the scalability of the algorithm and the accuracy are indirectly related, as the most accurate summary construction algorithms are frequently not scalable.

The *mhst* algorithm was proposed in [3] for the construction of multidimensional histograms for selectivity estimation and in [5] for approximate OLAP query answering. This algorithm proposes a judicious way to construct histograms by using the marginal distributions to determine the partitioning into buckets instead of doing a more time-consuming analysis (the marginal distributions are small comparing to the data set size). The space partitioning metric used in [3] was based on the determination of the largest differences between neighboring values in the marginal distribution (*mhst-Maxdiff*). It is well-known that a heuristic based on the homogeneity of the resulting buckets, such as the V-optimal variance based constraint [4] would produce higher quality histograms, but this heuristic would imply exponential construction time and therefore scalability would be lost, rendering the approach unusable [3,4]. Recently, this problem was dealt with for the V-Optimal constraint on uni-dimensional histograms for data sets with a reasonably small cardinality of values [1]. The problem of finding a scalable high-quality solution for high cardinality multi-dimensional data which would provide an important improvement to the *mhst* algorithm remained unsolved.

The main contribution of this paper is to solve this problem by proposing a histogram construction strategy *vmhist* that is able to maintain scalability while using the variance of candidate buckets to conveniently probe the homogeneity of regions, and therefore produce a high-quality multidimensional histogram without extra

performance cost. The algorithm uses pre-computation of values, additivity of error measures and localized updating features to achieve very accurate summaries while maintaining scalability.

The paper is organized as follows: section 2 introduces hierarchical partitioning and section 3 presents the *vmhist* technique. Section 4 presents experimental results and section 5 concludes the paper.

2. Hierarchical Partitioning

In order to produce a histogram summary of a data set, a space partitioning strategy must be followed. Adaptable partitioning refers to strategies that try to determine the best possible partitioning of the space into regions [2]. This section describes the basic hierarchical partitioning algorithm for multidimensional spaces. Given a data set with point values $p(a_1, \dots, a_n, v_1, \dots, v_j, \dots, v_m)$, the marginal data distribution of attribute a_i for the value attribute v_j is the set $\text{marg_}a_i(x_l, \Sigma \text{all database tuples with } a_i = x_l \ v_j)$, where x_l represents each individual value taken by attribute a_i . The marginal distribution is used in the analysis that determines the partitioning instead of the whole data, with obvious performance gains. The algorithm successively divides one or more buckets into two by the end-to-end splitting line that gives the best error measure results.

Figure 1 shows a two-dimensional space with the arrays of marginal distributions (sum of values or occurrences in the row or column) which are analyzed to determine the splitting lines.

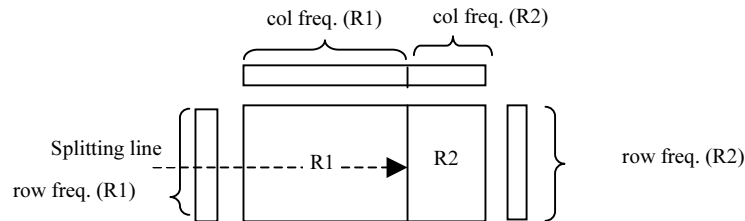


Figure 1- Hierarchical Partitioning and Marginal Distribution Arrays

Alternative partitioning constraints can be used to determine the best splitting line from the marginal distributions. For Maxdiff histograms (*mhist-Maxdiff*) [3], one with the largest difference in source values between adjacent values is used. For V-Optimal (variance optimal) histograms [4], the splitting line with maximum variance of source parameter values is used.

A naive implementation of a variance-based constraint in multidimensional data sets incurs in exponential histogram construction time due to the need to re-compute the measure in each iteration using the individual values of partitioned regions.

The next section proposes the *vmhist* technique, which uses pre-computed additive parameters and localized updates to render variance-based constraints useful in the multidimensional data set context.

3. Variance-Based Hierarchical Partitioning (*vmhist*)

The technique *vmhist* proposes a new scalable strategy to implement complex variance-based heuristics within the multidimensional partitioning context. The key concepts of this algorithm are:

- **Pre-computation** - almost all the parameters that are needed in the iterative partitioning process are pre-computed in extended marginal distribution arrays;
- **Additivity** - the parameters that are pre-computed and used in the computation of the homogeneity of regions are additive. This way, it is possible to compute the homogeneity without accessing all the values from the regions. Instead, the pre-computed parameters are used to derive a variance measure. Furthermore, only the region that was split needs an update of the homogeneity measure;
- **Localized updates** - only a very small set of the pre-computed values, corresponding to the region(s) that were split, have to be updated in each iteration.

The algorithm *vmhist* applies the policies described above to the basic hierarchical partitioning strategy. In particular, the variance can be computed additively for any region B_i from the values n_{B_i} (n° of values), ls_{B_i} (linear sum of values) and ss_{B_i} (square sum of values), as shown next:

$$\text{Variance} = \frac{\sum_{i=1}^n (v_i - \mu)^2}{n}$$

$$\text{Additivity: } V_{B_i}(n_{B_i}, ss_{B_i}, ls_{B_i}) = (ss_{B_i} - ls_{B_i}^2) / n_{B_i}$$

3.1. Pre-computing and Localized Updating of Partitioning Parameters in *vmhist*

The split decision in *vmhist*-MaxDiff [3] is taken locally (neighbor marginal distribution values). In contrast, the variance-based approach proposed here probes the variance resulting from all possible hierarchical splits in all dimensions to determine the split that produces the highest overall reduction in variance in each iteration. After the split, only the variances of the two resulting buckets need to be recalculated and this is done efficiently using pre-computed values. This means that *vmhist* uses the variance measure without the need to re-compute the values of the partitioning constraint except locally. This approach is only possible because the marginal distribution contains all the values that are needed to calculate the variance without accessing the source table.

As we have shown, the variance is easily expressed as a function of the additive parameters n_{B_i} , ss_{B_i} and ls_{B_i} . It can be computed for the union of any number of regions using these parameters that must be available for each region:

$$V_{B_i \cup B_j \cup \dots \cup B_l} = V(n_{B_i} + \dots + n_{B_j}, ls_{B_i} + \dots + ls_{B_j}, ss_{B_i} + \dots + ss_{B_j})$$

This way, the pre-computed quantities that must be kept to speedup the computation of variance are simply the additive parameters for each position of the marginal distributions, which have the structure:

```

dimension          // identifies the dimension attribute
bucket            // identifies the region
a                // value taken by the dimension attribute
-----
counti          // number of occurrences
lsi            // sum of values
ssi            // square sum of values
mvi          // the splitting condition to be maximized

```

The statement that creates the marginal distributions becomes:

```

SELECT count(di) , sum(vi) , sum(vi * vi)
FROM sourcetable
GROUP BY di;

```

Figure 2 illustrates the extended marginal distribution with the parameters described above and also additional cumulative sums that are discussed next.

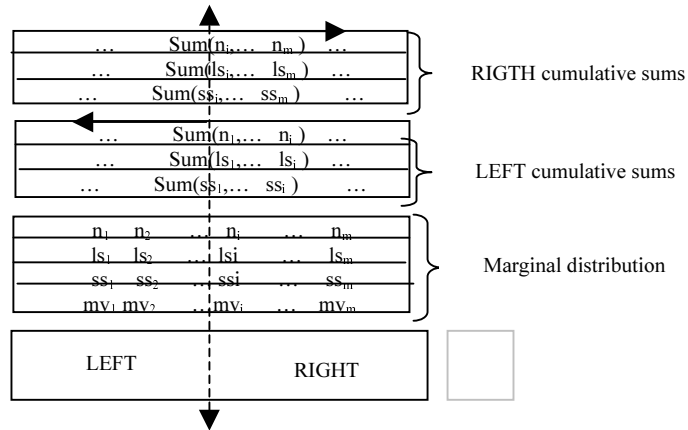


Figure 2 – Histograms of Cumulative Sums

The pre-computation and localized updating strategy is modified further in *vmhist*, to render the computation of the variance constraint extremely fast. The variances of the buckets formed before and after each candidate splitting line can be computed at any moment from pre-computed cumulative sums of the parameters that are used to derive the variance additively. The structure used to store the pre-computed parameters is a cumulative histogram that computes the sums of *ls*, *ss* and *count*, as shown in Figure 2. The variance and the cumulative values are both pre-computed in the beginning and only updated locally in each iteration. The LEFT/RIGHT histograms are pre-computed or updated by summing the values from the marginal distributions to the left or right of each value. For instance, the following statement updates the LEFT histogram:

```

sum(countj), sum(lsj) , sum(ssj), (sum(ssj) - sum(lsj) × sum(lsj) /sum(countj)
FROM marginal distribution
WHERE aj <= ai.

```

3.2. The Variance-based Partitioning Constraint of *vmhist*

In *vmhist*, a simple variance-based partitioning constraint was used which achieves very good accuracy results. The algorithm determines the best splitting lines as those with largest decrease in variance between the region and the candidate subregions for each possible splitting line. The *mv* parameter of the marginal distribution arrays is pre-computed and updated locally. The partitioning constraint maximizes the difference of variances between the bucket and the two buckets resulting from a possible partitioning:

$$\text{Variance}(B) - (\text{Variance}(B1) + \text{Variance}(B2)) \\ (mv = B.n \times B.var - \text{left}.n \times \text{left}.var - \text{right}.n \times \text{right}.var)$$

The best splitting line(s) are computed by maximization of the *mv* parameter.

4. Experimental Results

The main issues in the comparative evaluation between histogram construction algorithms are the construction time cost and the accuracy of the resulting summary. The algorithms must be scalable, as they are intended to summarize very large data sets. Figure 3 shows the construction time cost of three strategies: *mhist-Maxdiff* [3] (the basic implementation of *mhist* using the Maxdiff partitioning constraint); *vmhist* implementation of the Maxdiff (*vmhist-Maxdiff*), which also precomputes the partitioning constraint, and variance variant of *vmhist* (*vmhist-Variance*). The results show that the pre-computation and localized update features of *vmhist* resulted in faster execution than the basic *mhist* implementation that needs to re-compute the partitioning constraint in each iteration. The results also show that *vmhist-Variance* is practically as fast as *vmhist* using the Maxdiff constraint. In fact, it can be seen from the figure that the *vmhist* strategy is more scalable than *mhist-Maxdiff*.

Figure 4 shows comparative accuracy results between *mhist-Maxdiff* and *vmhist*, considering range-sum aggregation queries on a Sales data set and summaries with 5% of the original data set size. The aggregation categories are shown in the *x*-axis and the average number of values per group result are also shown. It is important to note that the accuracy is very good for any summary when large ranges are considered in all dimensions of the data set, but much larger errors result when queries detail some dimensions further or select fewer values. The errors of the group results are measured relative to the average value returned by the query. The figure shows that *vmhist* was always able to produce much better estimations than *mhist*.

Conclusions and Future Work

This paper presents a new hierarchical partitioning summary construction technique. This technique uses error measure additivity, pre-computation and localized updates to derive very accurate summaries. These optimization features allow variance constraints to be used scalably to summarize very large data sets,

providing a significant improvement over the previous strategies. The experimental results show the scalability and improved accuracy of the technique.

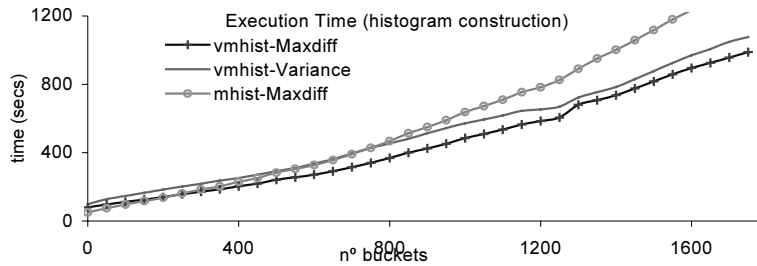


Figure 3–Histogram Construction Cost

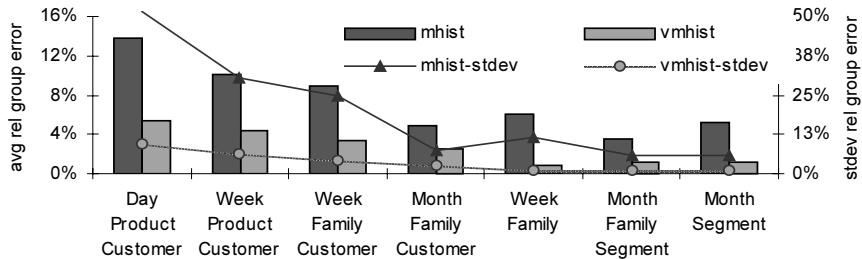


Figure 4-Comparative Accuracy of *vmhist* and *mhist*

References

1. H.V. Jagadish, N. Koudas, S. Muthukrishnan, V. Poosala, K. Sevcik, and T. Suel, "Optimal Histograms with Quality Guarantees", in Proc. 24th International Conference on Very Large Data Bases, August 1998, pp. 275-286.
2. S. Muthukrishnan, V. Poosala and T. Suel. "On Rectangular Partitionings in Two Dimensions: Algorithms, Complexity and Applications", in Procs. of the International Conference on Database Theory, Jerusalem, 1999.
3. V. Poosala, Y. Ioannidis, "Selectivity Estimation Without the Attribute Value Independence Assumption", Proceedings of the 23rd VLDB Conference, Athens, Greece, 1997.
4. V. Poosala, "Histogram-Based Estimation Techniques in Database Systems". PhD Thesis, University of Wisconsin – Madison, 1997.
5. V. Poosala, V. Ganti, "Fast Approximate Answers to Aggregate Queries on a Data Cube". 11th International Conference on Scientific and Statistical Database Management, Cleveland, Ohio, USA, 1999. IEEE Computer Society.
6. Special Issue on Data Reduction Techniques of the Bulletin of the Technical Committee on Data Engineering of the IEEE Computer Society, December 1997, Vol. 20, n 4.
7. T. Zhang, R. Ramakrishnan and M. Livny, "BIRCH: an Efficient Data Clustering Method for Very Large Databases", Proc. 1996 SIGMOD.