Query Size Estimation Using Clustering Techniques

Xiaoyuan Su
xsu1@umsis.miami.edu

Miroslav Kubat
mkubat@miami.edu

Moiez A. Tapia
mtapia@miami.edu

Electrical & Computer Engineering
University of Miami, Coral Gable, FL 33124, USA

Chao Hu
hcsfds@ee.ualberta.ca
Electrical & Computer Engineering
University of Alberta, Edmonton, AB, T6G 2V4, Canada

Abstract

For managing the performance of database management systems, we need to be able to estimate the size of queries. Query Size Estimation (QSE) is difficult if the queries are associated with more than one attribute. Here, we propose, and experimentally evaluate, a novel technique that builds on cluster analysis. Empirical results indicate that, in particular, density-based clustering QSE techniques are beneficial for medium and large sized databases where they compare favourably with partitioning clustering QSE ones such as k-means. This is observed especially in the case of noisy and dense datasets.

1. Introduction

The size of a query is the number of the objects that satisfy the query's constraints. Query size estimation (QSE) is an important task of a database management system. On one hand, the estimated size of a given query is used to choose the cheapest execution plan to the database. On the other hand, the users of the database system use the query size estimation as a way to detect errors in the query and misconceptions about the database [1].

Query size estimation techniques can be classified into two classes: sampling based and non-sampling based techniques. Some sampling based techniques estimate the query size by collecting and processing random samples of the data, some others collect and compute the samples according to the occurrence of the attributes in previous queries. Non-sampling-based techniques include parametric, curve fitting, machine learning and histogram methods etc [1].

However, traditional QSE techniques find it hard to accurately estimate the sizes of joint queries, especially for large and high-dimensional database. Consider the following query

$$Q: (20 \le a_1 \le 40) \wedge (35 \le a_2 \le 55) \wedge (56 \le a_5 \le 88) \quad (1)$$

where each clause is the constraint on an attribute. The maximum number of clauses or joint queries can be same as the number of attributes.

The clustering approach proposed in this paper belongs to non-sampling techniques. Different from parametric methods and histogram methods, the clustering method needs no assumption about the distribution of attribute values and inter-attribute independency. The approach is based on the real distribution of attribute values. When estimating the size of joint queries that associate with more than one attribute, the clustering technique will consider all the attributes simultaneously.

To evaluate the performance of clustering QSE techniques, we compare the results with another non-sampling based QSE technique, the histogram QSE technique that estimates query size according to the attributes one by one, and then combines the partial results based on the independency assumption. In our work, we used the easy-to-implement equi-width histogram method [2].

Proceedings of the 17th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'05)
1082-3409/05 $20.00 © 2005 IEEE

2. Clustering Methods

A cluster is a collection of objects that are similar to one another within the same cluster and are dissimilar to the objects in other clusters [3]. The measurement of the similarity between objects can be in term of distance. The definition of distance is highly application-dependent, Euclidean distance is one of the most widely used distances.

$$dist(x, y) = \sqrt{\sum_{i=1}^{d} (x_i - y_i)^2} \quad (2)$$

where d is the dimension number of the object and x_i, y_i are the values of the ith dimension of object x and y, respectively. As illustrated in equation 2, clustering methods take into account all the attributes of the objects. This feature facilitates the size estimation for the joint queries consisting of the constraints on more than one attribute.

Clustering methods can be classified into three classes: partitioning methods, density-based methods, and hierarchical methods. The *k*-means clustering method proposed by [4] belongs to the partitioning methods. It has two main advantages: relative efficiency and easy implementation. Among its weaknesses are the need to specify the number of clusters (k value) in advance and the sensitivity to noise. To remedy the first weakness, we can use some criteria to evaluate the quality of clustering for the results of different k values, and take the best one. Silhouette-coefficient [5] is one of the criteria for the quality of a *k*-means clustering.

Density-based clustering methods typically search for dense clusters of objects separated by sparse regions that represent noise. The key idea is that for each data point of a cluster, the neighbourhood of a given radius has to have at least a minimum number of data points. DBSCAN, OPTICS and DENCLUE are well-known density-based clustering methods [3]. In our work, we use OPTICS [6] clustering method.

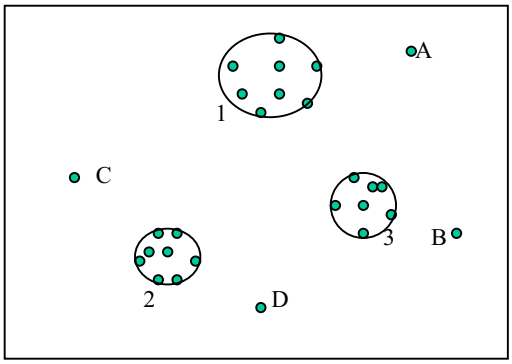

Figure 1. A simple example of clustering

The noise-sensitivity of *k*-means can be illustrated by the following example. In Figure 1, point A may be assigned to cluster 1 and point B to cluster 3. Estimating query size based on these clusters may wrongly estimate the actual size because of the sparsity of the clusters caused by noise. If we use density-based clustering methods, points A, B, C and D can belong to one sparse cluster, say cluster 4, whereas clusters 1, 2 and 3 are dense clusters.

The OPTICS algorithm [6] creates an ordering of the objects in a dataset, additionally recording the core-distance (the smallest distance value making object p a core object) and a suitable reachability-distance (the greater value of the core-distance of object p and the Euclidean distance between p and another object q) for each object. The ordering information produced by the OPTICS algorithm is sufficient for the extraction of all density-based clusters.

There is a variation of *k*-means clustering, called *X*-means [7], in which the shortcomings of *k*-means are alleviated and an optimal number of clusters are found together with the clusters. We will not implement *X*-means in this work.

3. Query Size Estimation Using Clustering

The histogram QSE technique, a traditional non-sampling QSE technique, computes the sub-result for each attribute with respect to the query using the histogram of that attribute, and combines the sub-results to get the final result under the independency assumption. But in practical database, the independency assumption may not hold. For example, in a company employee database, the values of the attributes "level of education" and "salary" usually depend on each other. If we have a joint query related to both attributes, we cannot get accurate size estimation by histogram method.

To deal with such queries, we propose the clustering approach that removes the independency assumption and considers the values of all attributes simultaneously.

3.1 Problem Formulation

In a relational database, a relation R consists of a set of objects. Each object has d attributes. A joint query to the relational database is a combination of some constraints on the attributes. For example, a query Q: $(10 \le x_1 \le 30) \wedge (20 \le x_2 \le 60)$ is a combination of two constraints on attributes x_1 and x_2.

In our approach, we treat the relation R as a d-dimensional set of points in a d-dimensional cube. Each object of R is a d-dimensional point. The answer of a query is a sub-cube of R. And the size of the query is the number of points in the sub-cube.

Proceedings of the 17th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'05)
1082-3409/05 $20.00 © 2005 IEEE

3.2 Framework

In our technique, the QSE is obtained by summing up the products of volumes of query-constrained sub-cubes within ranges of clusters and densities of the clusters obtained from cluster analysis.

After obtaining the clusters, we use them to estimate query sizes. For a given cluster C and a query Q, we compute the volume V of C and the volume V' of the sub-cube C' (for the portion within that cluster constrained by the Q). Then we estimate the size with

$$Q = V' \times \frac{N}{V} \tag{3}$$

where V' is the volume of the sub-cube within cluster C constrained by the query Q, N is the number of points in the cluster C, N/V is the density of cluster C (number of objects in each unit volume of the cluster).

To get the answer to the query, we need to sum up the estimated sizes for k clusters

$$S(Q) = \sum_{i=1}^{k} V_i' \times \frac{N_i}{V_i} \tag{4}$$

To calculate the sub-cube volume V_i', we use cluster shape assumptions, such as *rectangle assumption* and *ellipse assumption*. With the rectangle assumption, for each cluster i, suppose L_j and H_j are the low bound and high bound for dimension j of the cluster, x_j and y_j are low constraint and high constraint of the query on dimension j, R_j is the length of the query within the cluster for dimension j. We apply the rules in Figure 2 to define R_j (Figure 2).

If $x_j, y_j < L_j$ or $x_j, y_j > H_j$, then $R_j = 0$
If $x_j < L_j$ and $y_j > H_j$, then $R_j = H_j - L_j$
If $x_j < L_j$ and $L_j \le y_j \le H_j$, then $R_j = y_j - L_j$
If $L_j \le x_j \le H_j$ and $y_j > H_j$, then $R_j = H_j - x_j$
If $L_j \le x_j \le H_j$ and $L_j \le y_j \le H_j$, then $R_j = y_j - x_j$

Figure 2. Rules for defining a query's dimensional length within a cluster

Then the sub-cube volume of query within cluster i can be determined by

$$V_i' = \prod_{j=1}^{d} R_j \tag{5}$$

and the volume of the cluster i can be

$$V_i = \prod_{j=1}^{d} (H_j - L_j) \tag{6}$$

The rectangle assumption incurs some errors. In the example from Figure 3, we may imagine that ellipse assumption for clusters will be more accurate. But in fact, clusters can acquire any shape, and the calculation of volumes of arbitrary-shaped clusters for high dimensional datasets would be highly complex.

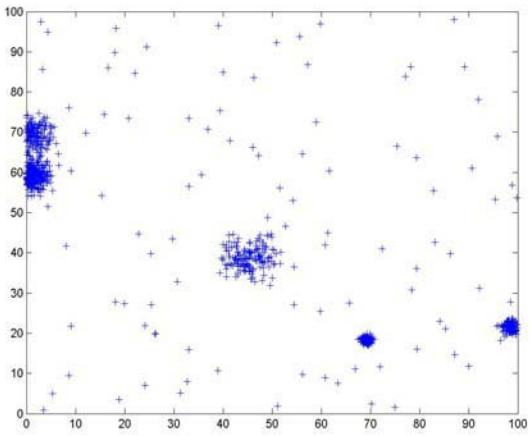

Figure 3. The 2-dimensional effect graph of *k1.data* (1,000 points, 6 dimensions)

As an interesting extension of our rectangle-assumption based query size estimation, we theoretically discuss the implementation of ellipse assumption based approach.

The volume of ellipse-shaped cluster i can be determined by

$$V_i = \frac{G_d}{2^d} \prod_{j=1}^{d} (H_j - L_j) \tag{7}$$

where G_d is the coefficients for the different dimension size d of the dataset, which can be determined using gamma function

$$G_d = \frac{\pi^{d/2}}{\Gamma(1 + \frac{d}{2})} \tag{8}$$

When dimension size d is even , $G_{d(even)} = \pi^{d/2}/(d/2)!$. When dimension size d is odd, $G_{d(odd)} = \frac{\pi^{d/2}}{\frac{d}{2}(\frac{d}{2}-1)(\frac{d}{2}-2)\ldots(\frac{3}{2})(\frac{1}{2})\pi^{1/2}}$. For example, when $d=2$, we get $G_2=\pi$, the resulting cluster volume value is $\pi(H_1-L_1)(H_2-L_2)/4$, which is the area of an ellipse; when $d=3$, we get $G_3=(4/3)\pi$, the resulting volume value

is $(4/3)\pi (H_1-L_1)(H_2-L_2)(H_3-L_3)/8$, which is the volume of an ellipse sphere.

The calculation of ellipse-assumption based sub-cube volume V' for high dimensional datasets will be extremely tedious and complex because of the boundary consideration, integration calculation etc. We will not implement the ellipse assumption in this paper.

Figure 4 illustrates our QSE technique. For the 2-dimensional joint query $(x_1<d_1<y_1)\wedge(x_2<d_2<y_2)$, we calculate the volumes V_1 and V_2 (areas in this example) of clusters C_1 and C_2 and densities of them using the rectangle assumption, calculate sub-cube volumes V'_1 and V'_2, and sum up the products of all the query-constrained sub-cube volumes and densities of clusters to get the overall query size.

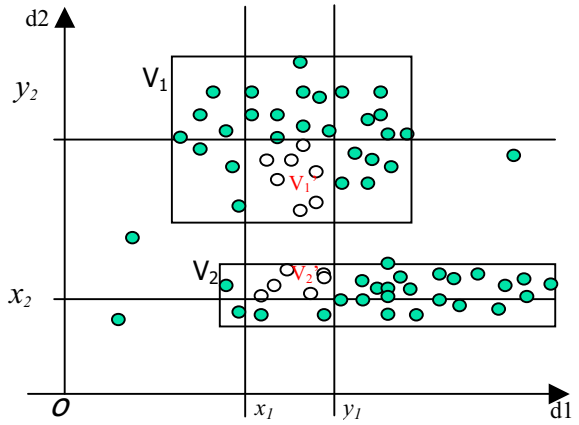

Figure 4. Query size estimation for the joint queries $x_1<d_1<y_1$ & $x_2<d_2<y_2$

For the special case of an equation query like $d = a$, we calculate the query's dimensional length R_j within a cluster in a special way. For example, in a certain attribute that has only 3 unique values 1, 2 and 3, we use 1/3 of the total dimensional length of the cluster for each equation query like $d = 1$.

4. Experiments and Results

We have experimented with three datasets, each with different sizes, dimensions, and noise rates. The *wine* domain is a real world data from the UCI machine learning repository [8]. It has 178 examples and 13 attributes (excluding the class column), and is known to have low noise. We normalize the values of each attribute to values between 0 and 100. We systematically generate two artificial datasets, with Gaussian distributions, pre-set dimensions, cluster numbers and noise rates, one of which (*k1.data*) has 1,000 data points, 6 attributes and 5 clusters with 10% of noise, and another one (*k10.data*) has 10,000 data points, 10 dimensions and 10 clusters and 20% of noise. All values in the artificial datasets are between 0 and 100. We estimate joint queries for each dataset with the joint query numbers from 2 to 6, using two different clustering QSE techniques, *k*-means clustering QSE technique and OPTICS clustering QSE technique (all with rectangle assumption), as well as one histogram QSE technique, equi-width histogram QSE technique. The estimation results (error rate) are averaged over 10 different queries for each joint query number.

Some examples of the queries for the dataset *k10.data* (2Q means 2 joint queries):

2Q: $(40\le d5\le 50)\wedge(10\le d6\le 30)$
3Q: $(30\le d6\le 40)\wedge(20\le d7\le 40)\wedge(10\le d8\le 30)$
6Q: $(20\le d1\le 60)\wedge(10\le d2\le 55)\wedge(53\le d4\le 100)$
$\wedge(20\le d7\le 50)\wedge(30\le d8\le 70)\wedge(20\le d9\le 60)$

We get the true value of the joint queries by scanning the dataset. Then we calculate the absolute estimation error rate of the QSE techniques using estimated value and true value of the query size.

$$E = \frac{|p - r|}{r} \times 100\ \% \qquad (9)$$

where p is the estimated query size and r is the true value of the query size. We only consider the queries with the true size of at least 3 to avoid irrationally high error rates. An error rate over 100% means the estimated query size is several times of the true query size.

The result shows, for the sparse and low noise rate dataset *wine.data*, the density-based clustering (OPTICS) QSE technique performs equivalent to partitioning clustering (*k*-means) QSE technique, and both of them perform better than the histogram QSE technique (Figure 5(a)) for joint query number bigger than 3. This shows that there is no apparent advantage of density-based clustering QSE technique over partitioning clustering QSE technique for query size estimation of low noise rate datasets. Although better than the histogram QSE technique, the 70% error rate of the clustering QSE techniques shows this approach may not be very accurate for sparse datasets.

We then worked on a medium-size and noisy dataset, *k1.data*, and observed that the density-based clustering (OPTICS) QSE technique had an average absolute error rate of 11.7%, much better than partitioning clustering (*k*-means) QSE technique's 379.9% and histogram QSE technique's 229.0% (Figure 5(b)). This indicates that the noise-handling ability of the density-based clustering QSE technique contributes to its superior performance.

On the larger and noisier dataset, *k10.data*, we made similar observations, with 15.1% error rate for density-based clustering (OPTICS) QSE technique, much better than *k*-means QSE's 169.5% and histogram QSE's 304.1% (Figure 5(c)). Performance of *k*-means QSE

technique is improved on *k10.data* compared with *k1.data*, with *k10* has higher dimensions than *k1* (10 for *k10* and 6 for *k1*), but it's still not an ideal QSE technique for dense, high-dimensional and noisy datasets because of its poor robustness against noise.

Of the three QSE techniques, histogram QSE technique, a traditional QSE technique, has the worst performance for high-dimensional datasets (*wine.data* and *k10.data*), which shows its inability to estimate size of joint queries for high-dimensional datasets.

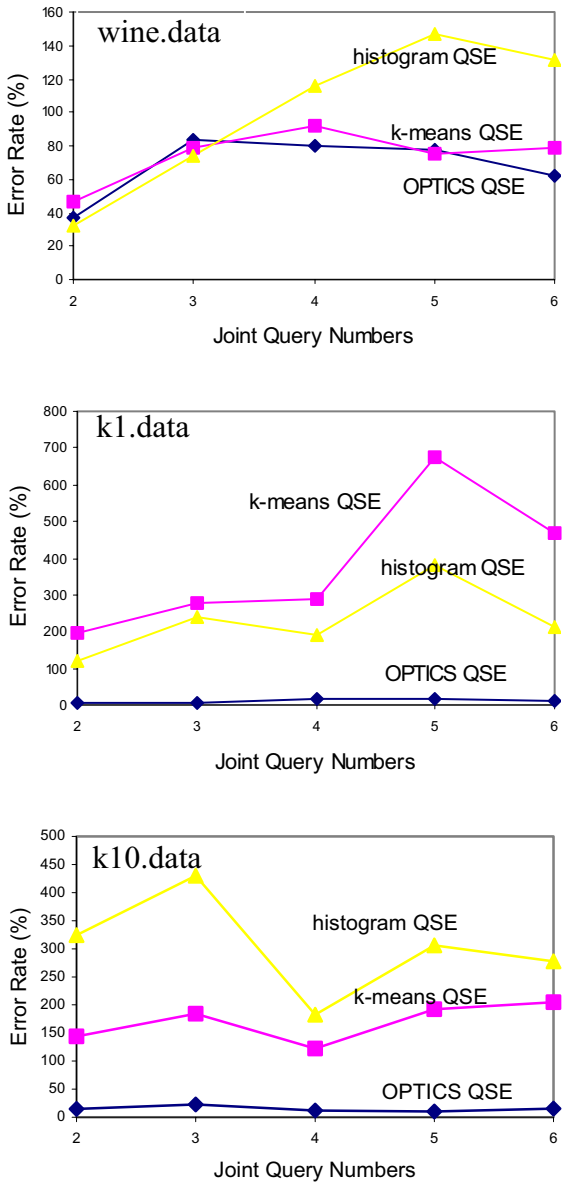

Figure 5. Performance of the 3 QSE techniques ((a), (b), (c) from up to down)

5. Conclusion

Contributions: We proposed and implemented a novel query size estimation technique for joint queries of medium-size and large-size high dimensional datasets: clustering query size estimation technique. Empirical experiments indicate that a density-based clustering QSE technique can very accurately estimate the size of joint queries for medium-size and large-size datasets, outperforming traditional non-sampling QSE techniques (such as histogram QSE technique) for all sizes of datasets, being better than the partitioning clustering (such as *k*-means) QSE technique for dense, high-dimensional and noisy datasets. We used the rectangle-shape assumption for the clusters when estimating the query size. We also theoretically discuss the QSE approach with ellipse shape assumption for clusters.

Acknowledgement: The research was partly supported by the NSF grant IIS-0513702. We thank Jörg Sander, Guohua Liu and Qihong Shen for their valuable suggestions and contributions.

References

[1] B. Harangsri, "Query Result Size Estimation Techniques in Database Systems", *Ph.D Thesis, the University of New South Wales*, Australia, 1998.

[2] G. Piatetsky-Shapiro and C. Connell, "Accurate Estimation of the Number of Tuples Satisfying a Condition", *Proceedings of ACM SIGMOD Conf.*, pp. 256-276, 1984.

[3] J. Han and M. Kamber, "Data Mining: Concepts and Techniques", *Morgan Kaufmann Publishers*, 2001.

[4] J. B. MacQueen, "Some methods for classification and analysis of multivariate observations", *Proceedings of the 5th Symposium on Math, Statistics, and Probability*, pp. 281-297, 1967.

[5] L. Kaufman and P.J. Rousseeuw, "Finding Groups in Data: an Introduction to Cluster Analysis", *Wiley*, 1990.

[6] M. Ankerst, M. M. Breunig, H.-P. Kriegel and J. Sander, "OPTICS: Ordering Points To Identify the Clustering Structure", *Proceedings of ACM SIGMOD Conf.*, 1999.

[7] D. Pelleg and A. Moore, "X-means: Extending K-means with Efficient Estimation of the Number of Clusters", *Proceedings of the 17th ICML*, pp. 727-734, 2000.

[8] UCI Repository of Machine Learning Databases.