Determining remote system contention states in query processing over the Internet

Weiru Liu
School of C&M
University of Ulster

Zhining Liao
School of C&M
University of Ulster

Jun Hong
School of C&M
University of Ulster

Abstract

In the environment of data integration over the Internet, three major factors affect the cost of a query: network congestion situation, server contention states (workload), and data/query complexity. In this paper, we concentrate on system contention states. For a remote data source, we first determine the total number of contention states of the system through applying clustering techniques to the costs of sample queries. We then develop a set of cost formulae for each of the contention states using a multiple regression process. Finally, we estimate the system's current contention state when a query is issued and using either a time slides method or a statistical method depending on the information we have about the system. Our method can accurately predict the system contention state so that the effect of the contention states on the cost of queries can be estimated precisely.

1 Introduction

To meet users' growing needs for sharing pre-existing data sources, query processing in data integration from a multitude of data sources over the Internet has become a research focus. One major challenge to wide applications processing is the dynamics and unpredictability of the workloads of both the network and remote servers. Another challenge is the autonomy of remote data sources. These sources may not provide metrics needed for accurate cost estimation. However the query processing procedure does need such information to re-write queries, and to decide where to send these sub-queries and how to integrate the returned data from different data sources. Therefore, methods of deriving cost models for autonomous data sources at a global level are significantly important in order to efficiently process queries. Several such methods have been proposed in the literature. A calibration method was proposed [3] to deduce necessary local (remote) cost parameters. This method was extended in [4] to calibrate cost models for object-oriented multi-database systems.

A query sampling method was proposed and applied in [11,12,13]. The key idea is to use sampling query technique to collect the cost model parameters for each local database and to keep the collected information in the MDBS catalog and to use these parameters during query optimization. The effects of the workload of a server on the cost of a query were examined from a different perspective in [14]. If a server was very busy, the server was likely to take longer time to answer the query, and the cost of the query was high. A method to decide the contention states of a server was developed and the cost models are constructed for each contention state in a multi-database system environment. Alternatively, an approach to combining a generic cost model with specific cost information exported by wrappers for local database systems was proposed in [8] and another approach to maintaining a cost vector database recording the cost information for every query issued to a local database was discussed in [1]. The estimated cost of a new query was calculated based on the costs of similar queries. Roth *et al* introduced a framework for calculating costs in a federated system, named *Garlic* [9]. Because *Garlic* was based on federated databases, it assumed that the optimizer had accurate information on the cost of each alternative query plan. The above-mentioned methods have all assumed that the network environment does not change significantly over time and some of methods are done in local network environment. They do not consider the impact of the wide area environment.

The importance of coping with a dynamic network environment has recently been addressed. The effects of network factors have been investigated in [5, 10] and a cost estimation model has been proposed that takes these factors into account by means of measuring the costs of the same query in different situations, e.g., different times of the day. Based on the feedback of queries, this method splits the time into time slides during seven days (of a week). So the costs of the same query at different time slides give the indication of whether the cost of other queries would be high or low at a particular time**.** Since the minimum time unit is one hour**,** this method cannot estimate the cost of a query accurately is the network is highly dynamic.

In this paper, we investigate how system contention state affects the cost of a query in wide area environment and propose our solutions to establish the relationship between them. There are three major contributions in the paper. First, we propose a clustering approach to determining the system contention states of a remote server. Given a particular server, one sample query is tested against it at fixed time intervals for 24 hours, so that a set of costs at those (clock) time points are collected. These costs will then be clustered into a number of groups. Each group determines one system contention state. Second, a set of cost formulae for each system contention state is constructed using a multiple regression model [2]. An adjustment of estimated cost is added to the cost formula in order to reflect the fact that the costs in each contention state can be diverse, that is, the difference between the minimum and maximum costs is big. This adjustment to the cost formulae enables more accurate prediction of the estimated cost. Finally, we predict the current contention state of system using either a time slides method or a statistical approach at the time when a query is submitted to a remote server. The time slides method is more suitable when the progression from one contention state to another is smooth. The statistical approach is more suitable when the progression is either fast or slow, but not beyond the scope of the two targeted contention states. Our cost model provides a more accurate cost of a query over the Internet.

The rest of the paper is organized as follows. Section 2 describes the clustering algorithm for determining the total number of system contention states for a remote server, and the extension of the contention state from discrete time points to continuous time points using the time slides method. Section 3 discusses how cost formulae can be obtained through a multi-regression process and how the adjustment is made to minimize the error of cost estimation. Section 4 investigates the selection of the right cost formula first and then introduces the statistical approach to determining the contention state when the time slides method is not applicable. In Section 5, we show some experimental results. Finally, Section 6 summarizes the paper.

2 Determination of System Contention States

In this section, we look at how the contention state of a system affects the time it takes to answer a query, so that we are able to establish the relationship between the contention state of a system and the cost of a query. To do so, a sample query is carefully designed. This query is of reasonable complexity and can be evaluated by the remote server quickly. It is tested on the remote server at a fixed time interval over 24 hours period. The costs of the query (T_i) at these time points (t_i) are collected.

2.1 Grouping costs of sample queries using clustering techniques

To determine an appropriate set of contention states for a dynamic environment, an algorithm often used for multi-dimensional data clustering [6] is modified to cluster one-dimensional data (*i.e.*, the cost of the sample query in terms of time spent by the server). The key idea underlying the algorithm is to place each data object (the cost of a sample query) in its own cluster initially and then gradually merge clusters into larger ones until a desired number of clusters have been found. The criterion used to merge two clusters C_1 and C_2 is to make their distance of minimal. One widely used distance measuring technique is the distance between the centroids or means. To assume $mean\ (C_1)$ and $mean\ (C_2)$ of two clusters, i.e.,

$$Dmean(C_1, C_2) = ||mean(C_1) - mean(C_2)||.$$

Let K be the maximum number of possible system contention states. The clustering algorithm can be used to obtain clusters $\Omega k = \{ C_1, C_2, \ldots, C_k \}$ such that $mean(C_i) < mean(C_{i+1})$ for $i = 1, 2, \ldots, k$ based on the costs of a chosen sample query at different time points. Then, $C = \{[min(C_1), max(C_1)], \ldots, [min(C_k), max(C_k)]\}$ gives a set of system contention states for a dynamic environment, where $min(C_i)$ and $max(C_i)$ stand for the minimum and maximum values in cluster C_i.

1.2 The relationship between contention states and time slides

As the response time of a query does have connections with the *Day* and the *Time* of a day, below we discuss how we link *Day* and *Time* with a system's contention states.

Let $T\_cost = \{T_1, T_2, \ldots, T_n\}$ be a set of costs of a sample query collected at clock time points $T\_time = \{t_1, t_2, \ldots t_n\}$ on a weekday. Under our assumption, $t_1 = 0.10$ (ten minutes after the midnight) is the first time point we send the sample query to obtain T_1 and $t_n = 24.00$ is the last time point to obtain T_n. The query is sent repeatedly at ten minutes intervals. (In fact, the time interval can be defined based on the requirement of a specific application.) Since the elements in *T_cost* have been divided into clusters C_i, the elements in *T_time* can be organized into the same number of clusters C_i', where C_i' contains those time points (t_i) at which the sample query costs fall into C_i.

For each time slide [t_i, t_j] where t_i and t_j are two neighboring time points in *T_time*, there are following two situations when we extend contention states.

Situation A: if costs T_i and T_j at time points t_i, and t_j are in the same cluster C_i, then a contention state at any time point in time slide [t_i, t_j] is defined by range [$min(C_i)$, $max(C_i)$] through cluster C_i.

Situation B: if costs T_i and T_j at time points t_i, and t_j are in two different clusters C_i and C_j, then a contention state at any time point in time slide [t_i, t_i+α) is defined by the range [$min(C_i)$, $max(C_i)$], a contention state in time slide (t_i+α, t_j] is defined by the range [$min(C_j)$, $max(C_j)$], and a contention state in time point t_i+α can be defined by either the range [$min(C_i)$, $max(C_i)$] or [$min(C_j)$, $max(C_j)$]. Here, α=(t_i+t_j)/2. However, using different contention states at time point t_i+α may result in a rather different estimated cost. In Section, 4 we will discuss how to choose the correct cost formula for this case.

Therefore, given any specific clock time point *t,* we can estimate the system's contention state by finding the time slide it falls in.

3 Cost Estimation Formulae

To determine appropriate cost models for system contention states, we carry out multiple regression analysis to build cost formulae [2,12]. The multiple regression process allows us to establish a statistical relationship between the costs of queries and the relevant contributing (explanatory) variables, as listed below. Such a statistical relationship can be used to establish a cost estimation formula for other queries in the same query class under the same system contention state.

3.1 Factors affecting the cost of a query

There are mainly five factors affecting the cost of a query in the wide area environment that we have mentioned as following:

1. *How many tuples in an operand table*.
2. *How many tuples in the result table*.
3. *The cardinality of an intermediate table*.
4. *The tuple length of the result table.*.
5. *Contention about the system, including system factors, such as* CPU, I/O, *or data items*, *and network factors, such as, network speed and data volume.*

The first two factors are often used by the query cost estimation. The 2nd and 4th factors decide the data volume that is being transferred on the Internet in a unary query. For a join query, the 2nd - 4th factors should be considered for the same reason. Factor 5th covers the overall environment affecting the cost of a query in addition to factors 1-4. Other factors (i.e. the tuple length of an operand) are less important in wide area environment and some other factors (i.e. the physical size, index of database) are not available in most cases in query processing systems over the Internet, since every data source is autonomous.

3.2 Multiple linear regression cost models

As we know the factors affecting the cost of query, we can construct the cost model as following. Let $X_1, X_2, \ldots, Xp$ be p explanatory variables, which correspond to the factors we discussed above in query processing. They do not have to represent different independent variables. It is allowed, for example, that $X_3 = X_1 * X_2$. The response (dependent) variable Y (which is the query cost in this paper) tends to vary in a systematic way with the explanatory variables X_s. If the systematic way is a statistical linear relationship between Y and X_s, which we assume is true in our application, a multiple linear regression model is defined as:

$$Y=B_0+B_1 X_1 +B_2 X_2 + \ldots + B_p X_p +\sigma \qquad \ldots (1)$$

Assume there are n observations (given n values) of dependent variable Y through p explanatory variables $X_1, X_2, \ldots, X_p$, each observation defines an equation:

$$Y = B_0 +B_1 X_{i1} +B_2 X_{i2} + \ldots + B_p X_{ip} +\sigma \qquad (i = 1,2, \ldots , n) \qquad \ldots(2)$$

where X_{ij} $(j = 1, 2, \ldots , p)$ denotes the value of the *j-th* explanatory variable X_j in the *i-th* observation (or experiment); Y_i is the value of *i-th* observation of dependent random variable Y corresponding to $X_{i1}, X_{i2}, \ldots X_{ip}$. B_0, $B_1, \ldots, B_p$ are regression coefficients for p explanatory variables which are unknown constants and will be determined by solving multiple (more than p) equations like (1) in each contention state (see details in the next section). $X_1, X_2, \ldots, X_p$ are known values as explained above.

In this paper, we discuss unary queries and join queries separately because these two kinds of queries require different parameters.

For an unary query, we let R_U be the operand table, S_U be its selectivity of a conjunctive term, N_U be the cardinalities of the operand table, N_{result} be the cardinalities of the result table, L_{result} is the tuple length of the result table. Then $LN_{result} = N_{result} * L_{result}$ is the data volume that is to be transferred to the user. The cost estimation formula for unary queries is:

$$Y = B_0 + B_1 * N_U + B_2 * N_{result} +B_3 * LN_{result} \qquad \ldots(3)$$

For a join query, we let R_{U1} be one of the operand tables and R_{U2} be the other operand table, N_{U1} and N_{U1} be its cardinalities of the operand tables, N_{result} be the cardinality of the result table, L_{result} be the tuple length of the result table. Then $LN_{result} = N_{result} * L_{result}$ is the data volume of the result table that is to be transferred to the user.

$$Y = B_0 + B_1 * N_{U1} + B_2 * N_{U2} + B_3 * N_{result} +B_4 * LN_{result} \qquad \ldots(4)$$

Given a system contention state [$min(C_i)$, $max(C_i)$] with l costs falling into it, it is possible to calculate B_0, $B_{1|\nu}$

not guaranteed to be the same. As a consequence, these different sets of coefficients will result in different cost estimation of the same query. We will show how to solve this problem in the next subsection.

3.3 Adjustment of cost formulae

In this section, we use the unary query as an example to discuss how to select costs to calculate coefficients and how to adjust the cost variant within a single contention state. Assume there are l costs ($l>>4$) falling into cluster C_i (for contention state [$min(C_i), max(C_i)$]) after applying the clustering algorithm to costs in $T\_cost=\{T_1, T_2, ..., T_n\}$. We choose 4 costs ($T_i$) among l cost within C_i which are the closest to the $mean(C_i)$. These four costs are then used to calculate the coefficients in Formula (2) for contention state [$min(C_i), max(C_i)$]. Obviously, if we had chosen four smallest (or largest) costs within C_i to get the coefficients, we would have had rather different values for the coefficients. To solve this problem, we modify Formula (2) as follows.

$$Y' = B_0 + B_1 * N_U + B_2 * N_{result} + B_3 * LN_{result} + \sigma(Y, C_i) \quad ... (5)$$

$\sigma(Y, C_i)$ is called the adjustment of the cost model and it is defined as:

$$\sigma(Y, C_i) = ((T_i^{tj} - mean(C_i))/mean(C_i)) * Y \quad ... (6)$$

Here $Y=(B_0+B_1 * N_U + B_2 * N_{result} + B_3 * LN_{result})$ is the estimated cost of query Q at time point t using Formula (2) at system contention state [$min(C_i), max(C_i)$] from cluster C_i. $T_i^{tj} \in T\_cost$ is the cost within C_i at observing time point $t^j \in T\_time$ which is the closest to time point t. The adjustment says what fraction of the estimated cost should be subtracted off (added on) to the estimated cost, in order to show that the estimated cost is only *for the average cases*. For example, when T_i^{tj} almost equals to $mean(C_i)$, the adjustment is close to 0 which implies that using Formula (2) itself can get the correct estimation. This adjustment is significant when $|min(C_i)-mean(C_i)|/mean(C_i)>=10\%$ (or $((max(C_i)-mean(C_i))/mean(C_i)>=10\%)$, since this condition says that within this contention state, this is a big drop (increase) between the lowest (highest) cost and the average cost of the same query. On the other hand, if $|min(C_i)-mean(C_i)|/mean(C_i) < \varepsilon$ (a pre-defined threshold, e.g., 10%) (or $((max(C_i)-mean(C_i))/mean(C_i) < \varepsilon)$, the adjustment loses its meaning, since this condition says that there is not much difference between any two costs of the same query obtained from two time points in this same contention state.

4 Selection of the correct cost formula

4.1 Selection of the correct cost formula in the time slides method

Assume query Q is submitted to a remote server for which we have derived a set of contention states using the clustering method defined above. We also assume that the system contention states have been extended to any time point as discussed in Section 2.3. Let t be the clock time when Q is submitted, t should fall between two time points t_i and t_j (or time slide [t_i, t_j]) in T_time. Let T_i and T_j be the costs obtained at time points t_i and t_j. Assume that the two clusters containing T_i and T_j are C_i and C_j. Based on the discussion in Section 2.3, there are two situations in which the contention states can be extended. Below we will use unary query as an example to see how to select the correct cost formula (especially the variable in the adjustment) for these situations. Situation B is further divided into two cases.

Situation A: where $C_i = C_j$, it is said that the system is at a steady workload situation and the contention state is [$min(C_i), max(C_i)$]. Using Formula (3), the cost of Q is estimated as

$$Y' = B_0 + B_1 * N_U + B_2 * N_{result} + B_3 * LN_{result} + \sigma(Y, C_i)$$

with $\sigma(Y, C_i) = \{(T_i^{ti} - mean(C_i))/mean(C_i)\} * Y$, where $T_i^{ti}=T_i$ when $|t_i-t|<|t_j-t|$; $T_i^{ti}=\mathrm{T_j}$ when $|t_i-t|>|t_j-t|$; and $T_i^{ti}=mean(C_i)$ when $|t_i-t|=|t_j-t|$.

Situation B1: when $min(C_i)<min(C_j)$, it is said that the server's workload is increased from state [$min(C_i), max(C_i)$] to [$min(C_j), max(C_j)$]. Using Formula (3), the cost of Q is estimated as

$$Y' = B_0 + B_1 * N_U + B_2 * N_{result} + B_3 * LN_{result} + \sigma(Y, C_i)$$

with $\sigma(Y, C_i) = \{(T_i^{ti} - mean(C_i))/mean(C_i)\} * Y$ where $T_i^{ti}=max(C_i)$ when $|t_i-t|<|t_j-t|$; $T_i^{ti}=min(C_j)$ when $|t_i-t|>|t_j-t|$; and $T_i^{ti}=(max(C_i)+ min(C_j))/2$ when $|t_i-t|=|t_j-t|$.

Situation B2: when $min(C_i)> min(C_j)$, it is said that the server's workload is decreased from state [$min(C_i), max(C_i)$] to [$min(C_j), max(C_j)$]. Using Formula (3), the cost of Q is estimated as

$$Y' = B_0 + B_1 * N_U + B_2 * N_{result} + B_3 * LN_{result} + \sigma(Y, C_i)$$

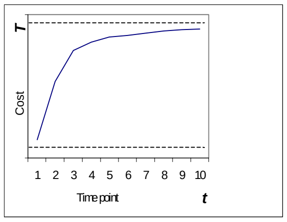
with $\sigma(Y, C_i) = \{(T_i^{ti} - mean(C_i))/mean(C_i)\} * Y$ where $T_i^{ti}=min(C_i)$ when $|t_i-t|<|t_j-t|$; $T_i^{ti}=max(C_j)$ when $|t_i-t|>|t_j-t|$; and $T_i^{ti}=(min(C_i)+ max(C_j))/2$ when $|t_i-t|=|t_j-t|$.

The last two estimations are suitable for situations where a server's workload changes from one state to another steadily. If the increase (decrease) of the workload is not steady, we will use a statistical method.

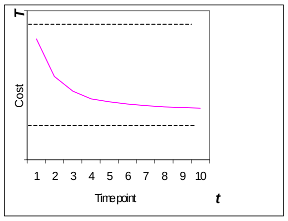
4.2 Determination of contention states using a statistical method

In this section, we develop a statistical model that uses a statistical function to describe the progression from system contention state S_i to S_j, when the progression is not smooth. Assume we have collected a number of costs of the sample query within the two time points t_i and t_j (or time slide [t_i, t_j]), and the contention states at these two time points are S_i and S_j. The distribution of these costs can be illustrated using one of the figures below. Figure

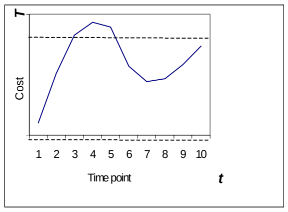
1a shows that during time interval (slide) [t_i, t_j], the cost increases and Figure 1b shows the cost decreases. If some of the costs within the interval are above the maximum cost or below the minimum cost in S_i and S_j (Figure 1c), it indicates that there is at least another system contention state occurring within the time interval. Therefore this time slide should be split. This topic is beyond the scope of this paper.



a



b



c

Figure 1. The possible cases from one state to another. (*t* and *T* stand for time point and the cost of the sample query respectively.)

In this section, we assume that all the additional observed costs within the time slide [t_i, t_j] are falling into interval [$min(min(S_i), min(S_j)), max(max(S_i) \; max(S_j))$]. The best mathematical simulation of the observed data as shown in Figure 1a and Figure 1b is a *function* as defined below.

$$y = a * e^{-b/x}, \; (b>0) \qquad \ldots (7)$$

where *a* and *b* are two parameters**.** *x* is the time point when a query is submitted. When *x* is given, this function is used to estimate the cost *y* of the query at this time point. Parameters *a* and *b* are determined using the observed data (costs y_i) at certain time points (x_i). It is easy to see that this function is exponential. To reduce the computational complexity, we adopt the method proved in [7] and transform this function into a linear regression formula by changing the variables. Formula (7) is thus in the form

$$y' = \beta_0 + \beta_1 * x' \qquad \ldots(8)$$

where $y' = \log_e y$ and $x' = 1/x$. The relationship between these two pairs of variables is $\beta_0 = \log_e a$ and $\beta_1 = -b$. β_0 and β_1 are calculated using equations below, known as *the least squares method*, a method of determining the parameters that best describe the relationship between expected and observed sets of data minimizing the sums of the squares of deviation between observed and expected values.

$$\beta_0 = \bar{y}' - \beta_1 \bar{x}', \qquad \ldots (9)$$

$$\beta_1 = \left(\sum_{i=1}^{n} x'_i * y'_i - n * \bar{x}' * \bar{y}'\right) / \left(\sum_{i=1}^{n} x_i'^2 - n * \bar{x}'^2\right) \ldots(10)$$

where $\bar{x}' = \frac{1}{n}\sum_{i=1}^{n} x'_i$, $\bar{y}' = \frac{1}{n}\sum_{i=1}^{n} y'_i$. Here y_i ($y_i' = \log_e y_i$) are the observed costs at time points x_i ($x_i' = 1/x_i$). Finally, the parameters (*a* and *b*) in the function are calculated using Equation (11).

$$a = e^{\beta_0}, \qquad b = -\beta_1. \qquad \ldots (11)$$

Once the values of *a* and *b* are known, we can use Equation (7) to estimate the cost of a sample query, denoted as y_Q, at time point *x* in time slide [t_i, t_j]. Based on the value of y_Q, we will be able to decide which contention state the system is in and then to choose the right cost formula as discussed above to estimate the cost of query *Q*. For example, if the progression curve is as shown in Figure 1a from state S_i=[$min(C_i)$, $max(C_i)$] to S_j=[$min(C_j)$, $max(C_j)$], and if the query is submitted at time point 3.5, then the estimated sample query cost indicates that the system contention state is in S_j and the cost formula is $Y' = B_0 + B_1 * N_U + B_2 * N_{result} + B_3 * LN_{result} + \sigma(Y, C_j)$ with $\sigma(Y, C_j) = \{(max(C_j) - mean(C_j))/mean(C_j)\} * Y$.

However, the time slide approach would have indicated that the current system contention state was S_i (because $|t_i - t| > |t_j - t|$) which was not correct.

5 Experimental Results

We carried out the following steps in our experiments when a query was submitttted.

1. Decide the current system contention state using either the time slide or the statistical methods.
2. Choose the right formula for the contention state
3. Estimate the parameters in the formula (N_u, N_{result}, LN_{result}).
4. Calculate the cost of the query.
5. Compare the estimated and observed costs.

We carried out experiments to evaluate the accuracy of our cost formulae. Our experiments were conducted between a remote server and a user. A server has been established at the University of Ulster using Windows 2000 as the operating system and Oracle 9.0 as database management system with 56 tables installed with tuple numbers ranging from 200 ~ 800,000. A sample query was sent from a user in China to this server a number of times at fixed time points and the corresponding query costs were collected. In our experiment, there are four contentions states. Figure 3~6 shows the relationship between the estimated costs and the observed costs in each contention states (In these figures, estimated verse observed costs of test queries for both unary and join queries in each contention state. Dashed lines with   are for estimated costs of unary queries and solid lines with solid squares are for the observed costs of the same unary queries. Similarly, a dashed line with solid • and a solid line with × are estimated and observed costs for join queries respectively). Table 1 and 2 are the parameters of cost estimation formulae for unary and join query in each contention state. Table 3 and 4 are the average estimated costs, observed costs and error rates for unary query and join query respectively. We drew following observations fro our experiments:

(1) Both the cardinality and the length of the result table are significant in both query classes.
(2) The maximum error rate is 26%. Most of error rates are around 10%. The error rate is acceptable when estimating the cost of a query in wide area environment.
(3) From table 4, we know that the error rate of join queries is higher than that of unary queries. The reason could be caused by the way to estimate the parameters in the formula (e.g., N_u, N_{result}, $LN_{result\ldots}$). We will continue to work on the problem.

Cont. States	B0	B1	B2	B3
State 1:	*1.84168657223*	*-8.01832057 E-5*	*0.24199654066*	*–4.83872528E-6*
State 2:	*-5.165334945*	*6.062131E -5*	*0.258267878*	*–2.8606061E-4*
State 3:	*2.09676928248E1*	*–1.8766981E-4*	*0.272761212*	*3.9059394 E-4*
State 4	*8.3451255319E1*	*– 7.1109199E-4*	*0.153626666*	*9.83466667E-4*
Single state:	*2.4497292496E1*	*–2.295809239E-4*	*0.2316508*	*2.707889437E-4*

Table 1. The parameters of cost estimation formulae for unary queries in each contention state

Cont States	B0	B1	B2	B3	B4
State 1:	*0.9538376*	*0.13936762E-3*	*0.1707536E-3*	*2.1415252E-4*	*6.0141673E-6*
State 2:	*1.336124*	*0.2215472E-3*	*0.216111E-3*	*3.507236E-4*	*1.1179426E-5*
State 3:	*1.91621544*	*0.5764138E-3*	*0.7265162E-3*	*9.0576512E-4*	*2.172937 E-5*
State 4:	*7.42515906E1*	*1.1109199E-3*	*2.27620745 E-3*	*2.55655532 E-3*	*7.717473 E-5*
Single state	*6.977701891E1*	*5.07278115 E-3*	*0.91683423 E-3*	*1.19382323 E-4*	*3.55945865E-5*

Table 2. The parameters of cost estimation formulae for join queries in each contention state

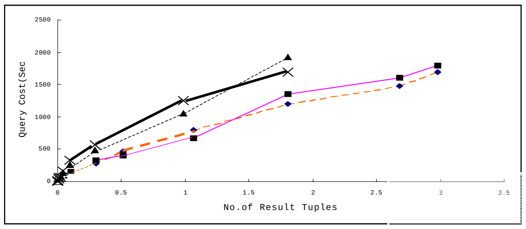


Figure 3. Query cost in contention state 1

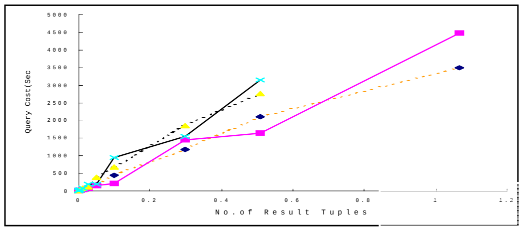


Figure 4. Query cost in contention state 2

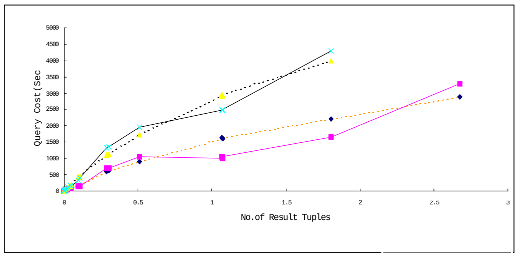


Figure 5. Query cost in contention state 3

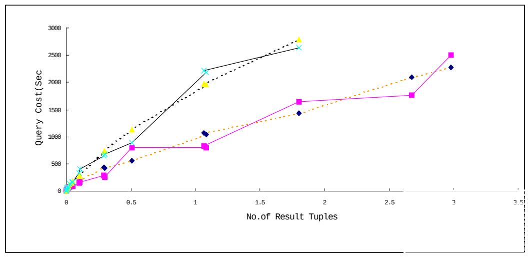


Figure 6. Query cost in contention state 4

System Contention State	$Y_{estimated}$	$Y_{observed}$	Error Rate (%)
Cont state 1	64.4687	70.7	7.399
Cont state 2	87.3663	80.580	9.421
Cont state 3	102.5289	118.735	13.648
Cont state 4	123.724	142.480	15.1627

Table 3. The average estimated cost, observed cost and Error rate for unary queries

System Contention State	$Y_{estimated}$	$Y_{observed}$	Error Rate (%)
Cont state 1	44.4813	40.67	9.371
Cont state 2	54.613	65.512	16.636
Cont state 3	72.5824	98.1359	26.038
Cont state 4	112.357	132.289	18.0627

Table 4. The average estimated cost, observed cost and Error rate for join queries

6 Conclusion and future work

A major challenge for performing global query optimization in a wide area application is that some local cost information may not be available at the global level. Most techniques proposed so far in the literature considered only static system environments. However, the cost of a query changes dramatically in a realistic dynamic environment. In this paper, we have investigated the relationship between the system contention states and the response time of a query using different techniques and made the following contributions.

- A clustering technique has been used to determine system's contention states.
- A set of regression cost formulae is given in each system state, and an adequate adjustment of the cost formula is defined to minimize the estimation error.
- A system's current contention state can be determined by two alternative approaches. One is the time slide approach where contention states at discrete time points are extended to any time points, and the system's current contention state is solely decided based on the time point when the query is submitted. Another is a statistical approach that is more suitable for slow (or fast) progression from one contention state to another, so that a more accurate prediction of the current contention state of the system can be determined.

Experimental results had shown that our method has high accuracy in estimating the cost of a query in wide area environment. Our immediate future work is to study the refinement of clusters when either two sets of clusters generated by two sample queries disagree with each other or query feedback suggest the existence of another contention state within a time slide. Our next step future work is to investigate the behavior of the Internet and the estimation of parameters involved in a query.

Acknowledgements

Many thanks to Zhiyong Luo and Zhifang Liao for acting as users during the experiment.

References

[1] Adali, S., K.S. Candan, Y. Papakonstantinou, and V.S. Subrahmanian, Query caching and optimization in distributed mediator systems. *In Proc. of ACM SIGMO*D, 1996, pp. 137–48.

[2] Chatterjee, S. and B. Price, *Regression Analysis by Example*, 2nd Ed. John Wiley & Sons, Inc., 1991.

[3] Du, W., R. Krishnamurthy, and M..C. Shan, Query optimization in heterogeneous DBMS. In *Proc. of VLD*B-1992, pp. 277–291.

[4] Gardarin, G., F. Sha, and Z.H.Tang, Calibrating the query optimizer cost model of IRO-DB, an object-oriented federated database system. In *Proc. of VLDB*-1996, pp. 378–389.

[5] Gruser, J. R., L. Raschid, V. Zadorozhny, and T. Zhan, Learning response time for web-sources using query feedback and application in query optimization. *VLDB Journal*, **9**(1), 2000, pp.18-37.

[6] Guha, S., R. Rastogi, and K. Shim**,** CURE: An Efficient Clustering Algorithm for Large Databases. In *Proc. of SIGMO*D, 1998, pp. 73–84.

[7] Hald, A.*Statistical theory with engineering application*. Jon Wiley & Sons, Inc. 1952.

[8] Naacke.H., G. Gardarin, and A. Tomasicl, Leveraging mediator cost models with heterogeneous data sources. In *Proc. of ICDE'*1998, pp.351–360.

[9] Roth, M.T., F. Ozcan, and L. M. Haas, Cost models DO matter: providing cost information for diverse data sources in a federated system. In *Proc. of VLD*B-1999, pp.599–610.

[10] Zadorozhny, V., L. Raschid, T. Zhan, and L. Bright, Validating an Access Cost Model for Wide Area Applications, *Coop. Inf. Sys. 2001*, pp.371-385.

[11] Zhu, Q. and P.A. Larson, A Query Sampling Method of Estimating Local Cost Parameters in a Multidatabase System. *Proc. of ICDE-1994*, pp.144-153.

[12] Zhu, Q. and P.A. Larson, Building Regression Cost Models for Multidatabase Systems. *PDIS'96,* pp.220-231.

[13] Zhu, Q., and P.A. Larson, Solving local cost estimation problem for global query optimization in multidatabase system. *Dist. and parallel databases*, 1998, pp.373-421.

[14] Zhu, Q., S. Motheramgari, Y. Sun, Cost estimation for large queries via fractional analysis and probabilistic approach in dynamic multidatabase environments, *Proc. of DEXA' 2000,* pp.509-525.