International Journal of Pure and Applied Mathematics
Volume 117 No. 20 2017, 727-739
ISSN: 1311-8080 (printed version); ISSN: 1314-3395 (on-line version)
url: http://www.ijpam.eu
Special Issue

Effective Cost Models for Web Query Optimization

[1]H.R. Shashidhar, [2]G.T. Raju and [3]Vinayaka Murthy

[1]Reva University.

Department of CSE,

RNSIT, Karnataka, India.

shashi_dhara@yahoo.com

[2]Department of CSE,

RNSIT, VTU, Karnataka, India.

gtraju1990@yahoo.com

[3]School of Computer Science and Applications,

Reva University,

Karnataka, India.

dr.m.vinayakamurthy@revainstitution.org

Abstract

Classical query optimizers rely on sophisticated cost models to estimate the cost of executing a query and its operators. By using this cost model, an efficient global plan is created by the optimizer which will be used to execute a given query. This cost modeling facility is difficult to be implemented in Web query engines because many local data sources might not be comfortable in sharing meta data information due to confidentiality issues.

In this work, an efficient and effective cost modeling techniques for Web query engines are proposed. These techniques does not force the local data sources to reveal their meta data but employs a learning mechanism to estimate the cost of executing a given local query. Two cost modeling algorithms namely: Poisson cost model algorithm and Exponential cost model algorithm is presented. Empirical results over real world datasets reveal the efficiency and effectiveness of the new cost models.

Key Words: Cost models, web query optimization, mediator, operators.

1. Introduction

Web applications [1] have become a major tool in storing and accessing data. The data sources of Web applications are often located in different geographical locations. The Web application integrates these different data sources and builds an user required application. This framework even-though has provided an opportunity to integrate different data sources with minimal overhead, it suffers from performance bottlenecks due to bloated response times [2]. Consider a Web application which provides users information about properties in a particular city. Now the details of each real estate company and their corresponding property information is stored in their respective databases. The owners of such data sources might not be willing to migrate their entire data information due to confidentiality reasons. So, a Web application can be built by integrating these data sources without migrating the local data [2].

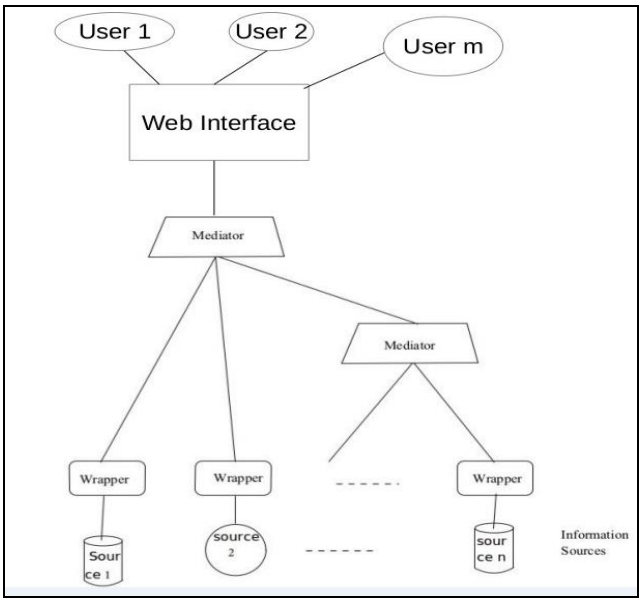

Figure 1: Web Query Engine

The Web application is built over a Web query engine shown in Figure 1 which is responsible for the execution of the query and providing the required answers. The Web application interface is connected with a component called as the Mediator. This Mediator manages the execution of the Web query. It divides the Web query into a group of local queries which will be mapped to the required local data sources. The local data sources also have a component called the Wrapper which acts as a interface between the local data source and the Mediator. The Mediator presents the local query to the Wrapper which executes this query. The executed result is converted to the desired format as specified by the Mediator. Then, Mediator combines the results from different Wrappers and joins the result to be

presented to the user of the Web application.

The classical query optimizers use sophisticated cost models to compute the cost of executing a particular query. This query cost is composed of individual cost of the operators that are part of the final query plan. Each query might generate multiple plans and the minimum cost plan is selected for query execution. The same facility is absent in Web query engines and it has remained an open issue [3–6]. The reason being, local data sources have their own individual operator/plans to execute a given local query. Local data sources might not be comfortable in sharing this execution plan information with the Mediator. Due to the absence of cost information the Mediator is handicapped to produce an efficient global plan. So, the mediator might produce an inefficient plan which can result in bloated response time for the Web application user thereby, increasing the dissatisfaction of the user in adapting that Web application. So, it is crucial to develop some efficient and effective techniques to model the cost of executing a local query without relying on the local data sources to provide that information.

In this work, the problem of developing cost models for Web query execution engine is addressed and the following contributions are made:

1. Empirical investigations were conducted to determine the best fitting cost model. Two cost models were found suitable to be applied on this problem. They are, Poisson cost model and the Exponential cost model.
2. The Poisson cost model is illustrated by describing the parameter estimation technique and cost model calculation algorithm.
3. Similarly, the Exponential cost model is presented with its parameter estimation technique and cost model calculation algorithm.
4. Empirical validation is performed on DBLP dataset. The effectiveness of these cost models are exhibited.

2. Related Work

There are 4 frameworks for designing Web Query execution engines. The cost based framework [7–10] produces the best plan which has the minimum cost among the set of competing plans. This framework has a similar design when compared with classical database optimizers. The other 3 remaining frameworks deal with the result quality [11, 12], failure adaptability [13–15] and data source quality [16–19]. These 3 frameworks do not aim to produce the minimum cost plan but, provide other functionalities such as, better quality of result, recovery from system failures and analyzing the properties of local data sources.

Effective cost models for Web query execution engine are still elusive. Lack of cooperation from the local data sources has lead to the design of

ineffective techniques which can suffer from frequent bloated response time problem [7–10].

Until, effective cost models are de- signed the Web query execution engine will continue to suffer from performance bottlenecks.

3. Problem Framework

Let, x and y be the number of tuples retrieved and the execution cost in seconds for a local query Q executed at the local data source L. The training set is given by, Training Set = $[x_1y_1, x_2y_2,...x_n y_n]$. Here, x_j and y_j $(1 \leq j \leq n)$ be the number of tuples retrieved and the execution cost in seconds for a local query Q_j .The task is to compute the execution cost $\hat{y}_i$ for a non training set query Q_i which has an estimated number of tuples $\hat{x}_i$.

4. Poisson Cost Model

The Poisson cost model is developed by using the Poisson distribution. For a random variable y, the Poisson density function is illustrated in Equation 1. The expected value for y is $E(y) = \mu$ and variance of y is $Var(y) = \mu$.

$$f(y) = \frac{e^{-\mu}\ \mu^y}{y!} \qquad y = 0, 1, 2.... \tag{1}$$

The regression function which models the relationship between x_i and y_i is shown in Equation 2.

$$y_i = E(y_i) + \varepsilon_i \qquad i = 1, 2....n \tag{2}$$

Here, $E(y_i) = \mu_i$.

The link function g( ) and its relation with μ_i is illustrated in Equa- tions 3 and 4.

$$g(\mu_i) = \eta_i = \beta_0 + \beta_1 X_1 + .. + \beta_k X_k = X_i^0 \beta \tag{3}$$

$$\mu_i = g^{-1}(\eta_i) = g^{-1}(\widehat{X_i^{/}\beta}) \tag{4}$$

If g( ) is an identity link then, its relationship with μ_i is shown in Equation 5.

$$\mu_i = g(\mu_i) = (X_i^{/}\beta) \tag{5}$$

If g( ) is a log link then, its relationship with μ_i is shown in Equation 6 and 7.

$$\log \mu_i = g(\mu_i) = (X_i^{/}\beta) \tag{6}$$

$$\mu_i = g^{-1}(X_i^{/}\beta) = e^{X_i^{/}\beta} \tag{7}$$

The parameter β needs to be estimated from the likelihood function shown in Equation 8.

$$L(y,\beta) = \prod_{i=1}^{n} f_i(y_i) = \prod_{i=1}^{n} \frac{e^{-\mu_i}\mu_i^{y_i}}{y_i!} = \frac{\prod_{i=1}^{n} \mu_i^{y_i} \exp(-\sum_{i=1}^{n}\mu_i)}{\prod_{i=1}^{n} y_i!} \tag{8}$$

The log likelihood function shown in Equation 9 is maximized wrt the parameter β. The parameter β obtained by this maximization procedure will be its estimated value which will be denoted as $\hat{\beta}$.

$$\log L(y,\beta) = \sum_{i=1}^{n} y_i \log(\mu_i) - \sum_{i=1}^{n} \mu_i - \sum_{i=1}^{n} \log(y_i!) \tag{9}$$

So, the regression function describing the relationship between $\hat{y}_I$ and $\hat{\beta}$ is described in Equation 10.

$$\hat{y}_i = g^{-1}(X_i^{/}\hat{\beta}) \tag{10}$$

Finally, the estimated value of y_i by using identity link function is given in Equation 11 and by using log link function is given in Equation12.

$$\hat{y}_i = X_i^{/}\hat{\beta} \tag{11}$$

$$\hat{y}_i = exp(X_i^{/}\hat{\beta}) \tag{12}$$

If, instead of x_i its estimated value $\hat{x}_i$ is used then, the Equations 11 and 12 become as shown in Equations 13 and 14.

$$\hat{y}_i = \widehat{X_i'}\hat{\beta} \quad (13)$$

$$\hat{y}_i = exp\,(\widehat{X_i'}\hat{\beta}) \quad (14)$$

The Algorithm 1 describes the procedure to estimate the cost of running a local query over a given local data source. In the pre processing module, the parameter $\hat{\beta}$ is estimated by log likelihood function maximization through the parameter β. The value of β that maximizes this log likelihood function will become the estimated value $\hat{\beta}$.

During query execution stage, the mediator system needs to calculate the cost of executing the local query. So, it estimates the cost $\hat{y}_i$ by using Equation 13 or Equation 14. The mediator system uses this cost information to build an efficient global plan for executing the Web query. After the executing, the actual number of tuples in result set of L_i and its execution cost is updated in training set to recalculate the parameter $\hat{\beta}$ for future cost calculation.

Algorithm 1: Poisson Cost Model Algorithm

[Pre-Processing Step]

Calculate the parameter

$\hat{\beta}$ from the training set by maximizing the log likelihood function given in Equation 9 w.r.t parameter β. [Query Execution Module]

Let L_i be a local query provided to a local data source by the Mediator system.

Let $\hat{x}_i$ be the number of tuples that can be retrieved for L_i.

Calculate the estimated cost of executing L_i by using the Equation 13 or Equation 14.

Send the estimated cost $\hat{y}_i$ to the mediator system. [Post-Processing Module]

After executing L_i, update the information about the actual cost of execution yi and actual result set size x_i to the training set.

Re perform the Pre-Processing Module.

5. Exponential Cost Model

The exponential cost model is built over the double exponential distribution shown in Equation 16. The regression model shown in Equation 15 can also be termed as robust regression model because unlike classical regression model, it does not assume normal observations of the training set.

$$y_i = \beta_0 + \beta_1 x_i + \varepsilon_i = x_i'\beta, \quad i = 1, 2, ..n \quad (15)$$

$$f(\varepsilon_i) = \frac{1}{2\sigma} e^{-\frac{|\varepsilon_i|}{\sigma}}, \quad -\infty < \varepsilon_i < \infty \tag{16}$$

The likelihood function to estimate the parameters β_0 and β_1 is shown in Equation 17. This involves minimizing the errors $\sum_{i=1}^{n} \varepsilon_i$.

But, the regression model does not assume normal errors and hence, least square parameter estimator used in maximizing the likelihood function fails to provide good estimates because it requires normal error distribution. So, robust estimators shown in Equation 18.

$$L(\beta_0, \beta_1) = \prod_{i=1}^{n} \frac{1}{2\sigma} e^{-\frac{|\varepsilon_i|}{\sigma}} = \frac{1}{(2\sigma)^n} \exp\left(-\sum_{i=1}^{n} \frac{|\varepsilon_i|}{\sigma}\right) \tag{17}$$

$$\min_{\beta} \sum_{i=1}^{n} \rho(e_i) = \min_{\beta} \sum_{i=1}^{n} \rho(y_i - x_i' \beta) \tag{18}$$

Equation 19 is an scale invariant version of Equation 18.

$$\min_{\beta} \sum_{i=1}^{n} \rho\left(\frac{e_i}{s}\right) = \min_{\beta} \sum_{i=1}^{n} \rho\left(\frac{y_i - x_i' \beta}{s}\right) \tag{19}$$

The parameter **s** is estimated according to Equation 20.

$$s = \frac{\text{median}\left|e_i - \text{median}(e_i)\right|}{0.6745} \tag{20}$$

To minimize Equation 19, the first partial derivatives of ρ w.r.t. β_j (j = 0, 1, ..k) are equated with 0 which provides the required conditions for minimization. This results in a system of p = k + 1 equations shown in Equation 21.

$$\sum_{i=1}^{n} x_{ij} \psi\left(\frac{y_i - x_i' \beta}{s}\right) = 0 \tag{20}$$

The Equation 21 can be rewritten as Equation 22 where the weights w_{i0} are given by Equation 23.

$$\sum_{i=1}^{n} x_{ij} \psi\left(\frac{y_i - x_i' \beta}{s}\right) = \sum_{i=1}^{n} x_{ij} \omega_{i0} (y_i - x_i' \beta) = 0, \quad j = 0,1,..k \tag{22}$$

$$\omega_{i0} = \begin{cases} \dfrac{\psi(y_i - x_i' \hat{\beta}_0)/s}{(y_i - x_i' \hat{\beta}_0)/s} & \text{if } y_i \neq x_i' \hat{\beta}_0 \\ 1 & \text{if } y_i = x_i' \hat{\beta}_0 \end{cases} \tag{23}$$

By using matrix notation Equation 22 becomes as shown in Equation 24.

$$X^{/}W_0 X\beta = X^{/}W_0 y \tag{24}$$

The estimated parameter $\hat{\beta}$ is shown in Equation 25.

$$\hat{\beta} = (X^{/}W_0 X)^{-1} X^{/}W_0 y \tag{25}$$

The Algorithm 2 describes the procedure for calculating the cost of executing a local query using the exponential cost model. This algorithm works on the similar lines of Algorithm 1 but instead of maximizing the likelihood function, minimization of robust regression function shown in Equation 19 is performed w.r.t. β to obtain the estimated parameter $\hat{\beta}$. Finally, the cost of executing the local query $\hat{y}_i$ is estimated by using the parameter β.

Algorithm 2: Exponential Cost Model Algorithm

[Pre-Processing Step]

Calculate the parameter $\hat{\beta}$ from the training set by minimizing the robust regression function given in Equation 19 w.r.t. parameter β.

[Query Execution Module]

Let L_i be a local query provided to a local data source by the Mediator system.

Let $\hat{x}_i$ be the number of tuples that can be retrieved for L_i.

Calculate the estimated cost of executing L_i by using the Equation 15 in which x_i is replaced by its estimated value $\hat{x}_i$ and parameter $\hat{\beta}$ is calculated by Equation 25.

Send the estimated cost $\hat{y}_i$ to the mediator system. [Post-Processing Module]

After executing L_i, update the information about the actual cost of execution y_i and actual result set size x_i to the training set.

Re-perform the Pre-Processing Module.

6. Experiments

The DBLP dataset is used for empirical study to demonstrate the performance efficiency of the proposed techniques for cost modeling. The dataset has a size of 650mb. The Web Query Engine was simulated by dividing each table in the DBLP dataset into simulated local data sources. Some of the tables were given exclusive security and autonomous privileges. Also, a modified DBLP dataset was created by injecting skew into the original tables. This skew version helps in evaluating the robustness of the new cost modeling techniques. The performance study involved both Poisson cost model algorithm(PCM) and Exponential cost model algorithm (ECM).

The first empirical study evaluates the performance of the 2 cost modeling techniques against the actual runtime costs. In Figure 2, both PCM and ECM perform similarly. This is because both the models have a good fitting for the cost estimation problem.

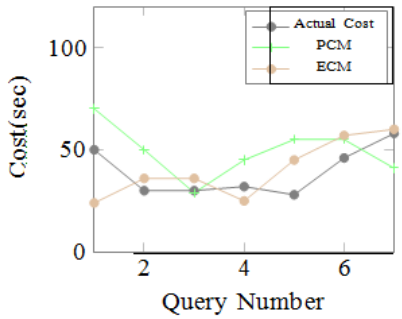

Figure 2: Cost vs Query(DBLP)

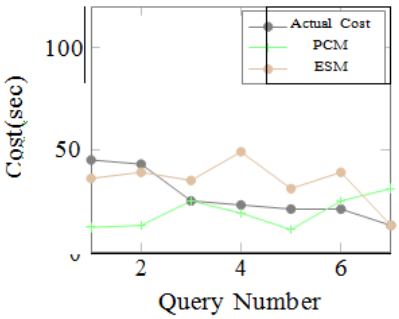

Figure 3: Cost vs Query(Skew DBLP)

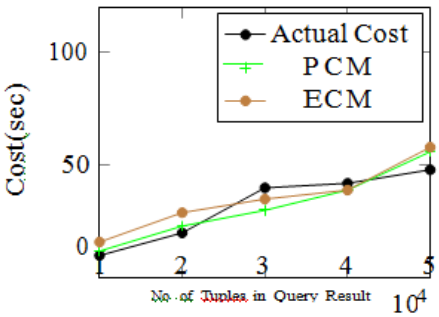

Figure 4: Cost vs No of Tuples(DBLP)

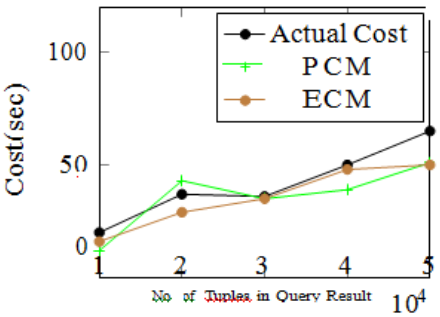

Figure 5: Cost vs No of Tuples (Skew DBLP)

The cost modeling techniques have a tendency to perform poorly in the presence of skew. So, the next empirical analysis shown in Figure 3 evaluates the robustness of PCM and ECM techniques. As seen in Figure 3, both techniques demonstrate considerable performance robustness in the presence of skew.

The analysis of cost of executing a query by varying the query result size is shown in Figures 4 and 5. The query result size has little influence on the estimated cost quality of the new cost model techniques.

The influence of database size on predicted costs of the new cost models are analyzed in Figures 6 and 7. This analysis involves, executing 2 queries on different sizes of the same database. As seen in Figures 6 and 7, the new cost models exhibit performance effectiveness even when there is variation in the number of tuples inside the underlying database.

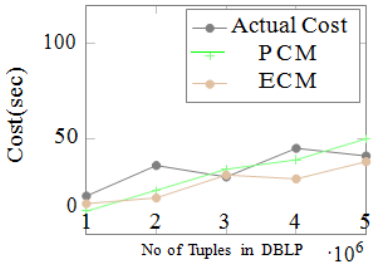

Figure 6: DB Size vs Cost

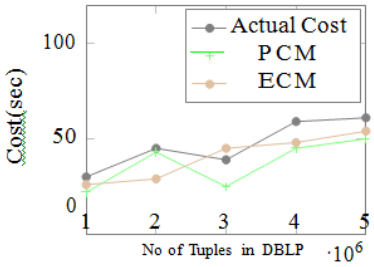

Figure 7: DB Size vs Cost

7. Conclusion

In this work, the problem of estimating the execution cost of a local query that needs to be executed at a local data source is addressed. Two techniques namely: Poisson cost modeling technique and exponential cost modeling technique is proposed. Both the approaches have proved their effectiveness as seen in the empirical results. These 2 models have been selected after thorough empirical evaluations on many real world datasets such as IMDB, Wikipedia and DBLP. Currently, these cost models can be directly used in many real world Web applications. In future, if some new dataset exhibits that these models are not suitable then, new model selection and evaluation is required.

In future, better cost models can be built which will use the partial metadata information that can be procured by the local data sources. Also, integrating the classical cost modeling technique and the Web query engine cost modeling technique would prove beneficial.

References

[1] Berners-Lee T., Hendler J., Lassila O., The semantic Web Scientific American 284 (5) (2001).

[2] Bertino E., Bouguettaya A., Introduction to the special issue on database technology on the Web, IEEE Internet Computing 6 (4) (2002).

[3] Ruiz A., Corchuelo R., Duran, Toro M., Automated support for quality requirements in web-based systems, Proceedings of the 8th IEEE Workshop on Future Trends of Distributed Computing Systems, Bologna, Italy, (2001).

[4] Arocena G.O., Mendelzon A.O., Web OQL: Restructuring documents, databases and Webs, IEEE 14th International Conference on Data Engineering (1998), 24-33.

[5] Batini C., Lenzerini M., Navathe S.B., A comparative analysis of methodologies for database schema integration, ACM computing surveys (CSUR) 18(4) (1986), 323-364.

[6] Berners-Lee T., Services and Semantics: Web Architecture, (2001).

[7] Tomasic A., Raschid L., Valduriez P., Scaling heterogeneous databases and the design of disco, IEEE 16th International Conference on Distributed Computing Systems (1996), 449-457.

[8] Haas L.M., Kossmann D., Wimmers E.L., Yang J., Optimizing queries across diverse data sources, Proceedings of the 23rd International Conference on Very Large Data Bases (VLDB), Athens, Greece (1997).

[9] Ambite J.L., Knoblock C.A. Flexible and Scalable Query Planning in Distributed and Heterogeneous Environments, AIPS (1998), 3-10.

[10] Adali S., Candan K.S., Papakonstantinou Y., Subrahmanian, V.S., Query caching and optimization in distributed mediator systems, ACM SIGMOD Record 25 (2) (1996), 137-146.

[11] Braumandl R., Keidl M., Kemper A., Kossmann D., Kreutz A., Seltzsam S., Stocker K., Object Globe: Ubiquitous query processing on the Internet, The Journal of VLDB 10(1) (2001), 48-71.

[12] Naumann F., Lesser U., Quality driven integration of heterogeneous information systems, Proceedings of the 25th International Conference on Very Large Data Bases (VLDB), Edinburgh, UK (1999).

[13] Hellerstein J.M., Franklin M.J., Chandrasekaran S., Deshpande A., Hildrum K., Madden S., Shah M.A., Adaptive query processing: Technology in evolution, IEEE Data Eng. Bull., 23(2) (2000), 7-18.

[14] Ives Z.G., Florescu D., Friedman M., Levy A., Weld D.S., An adaptive query execution system for data integration, ACM SIGMOD Record 28 (2) (1999), 299-310.

[15] Amsaleg L., Bonnet P., Franklin M.J., Tomasic A., Urhan T., Improving responsiveness for wide area data access, IEEE Data Engineering Bulletin, 20(3) (1997).

[16] Garcia-Molina H., Papakonstantinou Y., Quass D., Rajaraman A., Sagiv Y., Ullman J.D., Vassalos V., Widom J., The TSIMMIS approach to mediation Data models and languages, Journal of Intelligent Information Systems 8(2) (1997).

[17] Levy A., Rajaraman A., Ordille J., Querying heterogeneous information sources using source descriptions, Proceedings of the 22nd International Conference on Very Large Data Bases (VLDB), Bombay, India, (1996).

[18] Duschka O.M., Genesereth M.R., Query planning in infomaster, in Proceedings of the Twelfth Annual ACM Symposium on Applied Computing, SAC 97, San Jose, CA, USA, (1997).

[19] Florescu D., Levy A., Manolescu I., Suciu D., Query optimization in the presence of limited access patterns, Proceedings ACM SIGMOD International Conference on Management of Data, Philadephia, Pennsylvania, USA, (1999).

[20] Liu, Mengmeng, Ives, Zachary G., Loo, Boon Thau, Enabling Incremental Query Re-Optimization, Proceedings of the International Conference on Management of Data, (2016).

[21] Ramachandra, Karthik, Sudarshan S., Holistic Optimization by Pre fetching Query Results, Proceedings of the ACM SIGMOD International Conference on Management of Data, (2012).

[22] Moh, Teng-Sheng, Irani, Jehaan, Random Selection Assisted Long Web Search Query Optimization, Conference on Proceedings of the 50th Annual Southeast Regional, (2012),

[23] Lipton, Richard J., Naughton, Jeffrey F., Schneider, Donovan A., Practical Selectivity Estimation Through Adaptive Sampling, Proceedings of the ACM SIGMOD International Conference on Management of Data, (1990).

[24] Hellerstein, Joseph M., Haas, Peter J., Wang Helen J., Online Aggregation, Proceedings of the ACM SIGMOD International Conference on Management of Data, (1997).