Performance of Optimal Service Pricing for Correlations

P. Rajdeepak, K. John Paul

Dept. of CSE, NOVA College of Engineering & Tech., Jangareddygudem, AP, India

Abstract
This work proposes a novel pricing scheme designed for a cloud cache that offers querying services and aims at the maximization of the cloud profit. We define an appropriate price-demand model and we formulate the optimal pricing problem. The proposed solution allows: on one hand, long-term profit maximization, and, on the other, dynamic calibration to the actual behavior of the cloud application, while the optimization process is in progress. We discuss qualitative aspects of the solution and a variation of the problem that allows the consideration of user satisfaction together with profit maximization. The viability of the pricing solution is ensured with the proposal of a method that estimates the correlations of the cache services in an time-efficient manner.

Keywords
Cloud Data Management, Data Services, Cloud Service Pricing

I. Introduction
Cloud management necessitates an economy, and, therefore, incorporation of economic concepts in the provision of cloud services. The goal of cloud economy is to optimize: (i) user satisfaction and (ii) cloud profit. While the success of the cloud service depends on the optimization of both objectives, businesses typically prioritize profit. To maximize cloud profit we need a pricing scheme that guarantees user satisfaction while adapting to demand changes. Recently, cloud computing has found its way into the provision of web services [15, 18]. Information, as well as software is permanently stored in Internet servers and probably cached temporarily on the user side. Current businesses on cloud computing such as Amazon Web Services [14] and Microsoft Azure [19], have begun to offer data management services: A cloud cache manage the data of back-end databases in a transparent manner. Applications that collect and query massive data, like those supported by CERN [17], need a caching service, which can be provided by the cloud [31]. The goal of such a cloud is to provide efficient querying on the back-end data at a low cost, while being economically viable, and furthermore, profitable; depicts the architecture of a cloud cache. Users pose queries to the cloud through a coordinator module, and are charged on-the-go in order to be served. The cloud caches data and builds data structures in order to accelerate query execution. Service of queries is performed by executing them either in the cloud cache (if necessary data are already cached) or in a back-end database. Each cache structure (data or data structures) has an operating (i.e. a building and a maintenance) cost. A price over the operating cost for each structure can ensure profit for the cloud. In this work we propose a novel scheme that achieves optimal pricing for the services of a cloud cache. The cloud makes profit from selling its services at a price that is higher than the actual cost. Setting the right price for a service is a non-trivial problem, because when there is competition the demand for services grows inversely but not proportionally to the price. There are two major challenges when trying to define an optimal pricing scheme for the cloud caching service. The first is to define a simplified enough model of the price demand dependency, to achieve a feasible pricing solution, but not oversimplified model that is not representative. For example, a static pricing scheme cannot be optimal if the demand for services has deterministic seasonal fluctuations. The second challenge is to define a pricing scheme that is adaptable to (i) modeling errors, (ii) time-dependent model demand for services, for instance, may depend in a no predictable way on factors that are external to the cloud application, such as socioeconomic situations. A representative model for the cloud cache should take into account that the cache structures (table columns or indexes) may compete or collaborate during query execution. The demand for a structure depends not only on its price, but also on the price of other structures. For example, consider the query select A from T where B = 5 and C = 10. Out of the set of candidate indexes to run the query efficiently, indexes Ib = T(B), Ic = T(C), and Ibc = T(BC) are most important, since they can satisfy the conditions in the ‘where’ clause. If the cache uses Ibc, then the indexes sIb and Ic, will never be used, since Ibc can satisfy both conditions. Therefore, the presence of Ibc has a negative impact on the demand for Ib and Ic. Alternatively, if the cache uses Ib, then Ic can also improve query performance via index intersections, hence increasing the profit for the cloud. Therefore, indexes Ibc and Ic have positive impact on each other’s demand. An appropriate estimation method is necessary to model price-demand correlations among cached structures. The peculiarity of the pricing problem for the application of the cloud DBMS, in comparison with other businesses, is that the selling good is not a consumable product, but a persistent service. A consumable product diminishes with demand and has to be ordered, whereas a cloud cache service can satisfy infinite demand as long as it is maintained. Moreover, the demand for a cache service pauses if this service is not available. A consumable product may cost to maintain depending on the stored amount, whereas the maintenance cost of a cache service depends only on time. Moreover, a cache service may have a setup cost each time it is loaded in the cloud. A big challenge for the cloud is to optimize the set of offered services, i.e. decide which services to offer and when, depending on their demand while they are available. Roughly, the cloud has to schedule online and offline periods of the offered services, which affects the maintenance and the setup cost. Furthermore, the optimization of the cloud profit has to be scheduled for a long period in time while it is flexible during this period to adjust to the real evolution of the service consumption. The long-term profit optimization is necessary in order for the cloud to schedule ahead associative actions for the maintenance of the cloud infrastructure and the cloud data. Moreover, the cloud can schedule the service availability according to the guarantees for the overall revenue estimated by the long term optimization. Nevertheless, it is important that the long-term optimization process is flexible enough to receive corrections while it is still in progress. The corrections may refer to the difference between the estimated and the actual price influence on the demand of services.

II. Existing System
The cloud cache is a full-fledged DBMS along with a cache of data that reside permanently in back-end databases. The goal of the cloud cache is to offer cheap efficient multi-user querying on the back-end data, while keeping the cloud provider profitable. Our motivation for he necessity of such a cloud data service provider derives from the data management needs of huge analytical data,
such as scientific data [31], for example physics data from CERN [17] and astronomy data from SDSS [20] Furthermore, a viable, and moreover, profitable data service provider can achieve cost and time efficient management of smaller scientific collections or any type of analytical data, such as digital libraries, multimedia data and a variety of archived data. Users pose queries to the cloud, which are charged in order to be served. Following the business example of Amazon and Google, we assume that data reside in the same data center and that users pay on-the-go based on the infrastructure they use, therefore, they pay by the query. Service of queries is performed by executing them either in the cloud cache or in the back-end database. Query performance is measured in terms of execution time. The faster the execution, the more data structures it employs, and therefore, the more expensive the service. We assume that the cloud infrastructure provides sufficient amount of storage space for a large number of cache structures. Each cache structure has a building and a maintenance cost, presents at a high level the query execution model of the cloud cache. The names of variables and functions are self-explanatory. The user query is executed in the cache iff all the columns it refers to are already cached. Otherwise it is executed in the backend databases. The result is returned to the user and the cost is the query execution cost (the cost of operating the cloud cache or the cost of transferring the result via the network to the user). The cloud cache determines which structures cached columns, views, indexes) S to build in order to accelerate query execution and reduce the query execution cost. Initially S is empty and gradually it is filled with structures that would have or have benefitted past queries. How S is populated and how the costs of building and maintaining cache structures as well as the query execution cost are computed is an input to the presented optimal pricing scheme. More details on these issues can be found in [7]. Periodically on predefined time intervals t[i] the cloud performs the pricing scheme proposed in this work. The pricing scheme schedules the availability Δ and sets the prices P of the structures S for a time horizon T as described in the rest of the paper. The goal is to maximize the provider’s profit and at the same time ensure that the user is not overcharged.

III. Proposed Work

The pricing scheme depends on the estimated values of price-demand correlations for all structures, which as stored in the matrix V (see the constraint 10). The key to the maximization of profit is the maintenance of collaborations and the elimination of competitions between structures, by pricing the structures appropriately. The success of the scheme depends greatly on the accuracy of the estimation of the correlation degree for all candidate structures. We refer to the elements, vij, i, j = 1, . . . , m of V, as correlation coefficients, defined as follows: Definition 2: For any pair of structures Si and Sj we define the symmetric correlation coefficient vij = vji that represents the combined usage of Si and Sj in executed query plans. In order to construct a measure for correlation estimation, we define the following requirements:

1. The correlation coefficient vij should satisfy the following requirements: R1 vij is negative if Si can replace Sj and the opposite, positive if they collaborate, and zero if they are used independent of each other in query plans. R2 vij can be normalized for any pair of Si and Sj, R3 vij is easy to compute.

2. The sign of the coefficient vij denotes the competitive or collaborative behavior between a Si and Sj. Their presence does not affect each other, the coefficient should be zero.

3. We give an example. Example 1: In a workload with only one query, Q = select A from T where B = ‘b’ and C = ‘c’, the columns B and C should have positive correlation, while the indexes IA−D = T(A,B,C,D) and IA−E = T(A,B,C,D,E) should have negative correlation, and an irrelevant to the query index T(E,F) should have zero correlation. It is straightforward that the pricing scheme requires these properties from the correlation coefficients V.

4. The pricing scheme V determine the price of all the structures in the cloud cache (see constraint 10). If their values are not normalized, the pricing scheme is biased towards specific structures with high coefficient values. R3: It is necessary to compute all correlation coefficients V before the structures are materialized or even selected by the cloud cache. Materialization and selection of cache structures is an online procedure performed for each query execution. Therefore, the correlation coefficients must be computed efficiently and scalable. With respect to these requirements, we discuss a recently proposed correlation measure and its limitations. Then we propose a new measure that satisfies all the requirements.

An important issue is to estimate the appropriate length of the time period, in which we seek to optimize the cloud profit. Specifically, we have to determine the value of T which represents the optimization horizon of Eq. 4. Intuitively, a long horizon allows the optimization procedure to take into account the inertia of the system, whereas a short horizon may preclude the procedure from taking into account important long-term effects of current optimization decisions. Example 2: Assume a structure S with demand λS(t) and an optimization procedure of two short phases [0,Tsmall] and [Tsmall ,Tbig) or a procedure with one long phase [0,Tbig). For simplicity, the demand is a step function, i.e. λS(t) = λ1, t ∈ [0,Tsmall] corresponding to price p1 and λS(t) = λ2, t ∈ [Tsmall ,Tbig) corresponding to price p2 (for simplicity we ignore structure correlations). Assume that the building cost of S is BS and the maintenance cost is MS(t) = a · t and S is built once at time t = 0. The cloud profit in [0,Tsmall] is rsmall = λ1 · p1 − BS − MS(Tsmall). If rsmall < 0, the cloud decides to discard S and the second optimization phase starts with S not available. Since the demand is significant in (Tsmall ,Tbig), the cloud may decide to build S again, at t ≥ Tsmall, resulting in profit rbig−small ≤ λ2 · p2 − BS − MS(Tbig − Tsmall). For the long-term optimization...
the profit is: \( r_{\text{big}} = \lambda_1 \cdot p_1 + \lambda_2 \cdot p_2 - B_S - M_S(T_{\text{big}}) \). Obviously, \( r_{\text{big}} > r_{\text{small}} + r_{\text{big-small}} \). Therefore, the result of the two-phase short-term optimization procedure is not as optimal as that of the one-phase long-term procedure. Naturally, the prediction of future behavior of a system is subject to unpredictable perturbations. Hence, the longer the horizon is, the more error-prone the optimization procedure is, as the prediction accuracy of the behavior of demand, tends to decrease with time. We have assumed that the parameters of the constraints in Eq. 10 are constant. Yet, it is possible that in a real system the dependency of demand on the prices changes with time, because of any reasons. This means that the parameters, \( A, B, \Gamma \) should be time-varying. Even though the dynamics of Eq. 10 would be more realistic, they would highly increase the complexity of the problem, as there is no way, without a priori knowledge to determine time varying parameters with more confidence than fixed parameters contrary to what can happen for physical systems where degradation, e.g., of physical parameters can be models. Hence the problem falls in the scope of optimization of uncertain systems (potentially subject to model mismatch or parametric uncertainty or disturbances), which is an active research domain [12, 34]. In this context it can be shown that the use of measurements and of feedback is able to reject a part of the detrimental impact of parametric uncertainty on the optimal performances. In our case, real demand values are fed back as the optimization horizon slides, which increases the robustness of the proposed approach. As mentioned, Model Predictive Control has been widely used in Industry, where accurate dynamic models are almost never available. In these situations using tendency models (i.e. models that capture the main trends of a process) and measurements is generally sufficient to improve the process performances up to such a level that the costly efforts for identifying a more accurate process model are not justified by the loss of optimality [28]. Finally, as the optimization proceeds, new data is collected and this data can clearly be used to reidentify the price/demand model periodically.

**IV. Results**

In order to implement these three algorithms on the basis on time complexity and space complexity.

**V. Conclusions**

Cloud applications that offer data management services are emerging. Such clouds support caching of data in order to provide quality query services. The users can query the cloud data, paying the price for the infrastructure they use. Cloud management necessitates an economy that manages the service of multiple users in an efficient, but also, resource economic way that allows for cloud profit. Naturally, the maximization of cloud profit given some guarantees for user satisfaction presumes an appropriate price-demand model that enables optimal pricing of query services. The model should be plausible in that it reflects the correlation of cache structures involved in the queries. Optimal pricing is achieved based on a dynamic pricing scheme that adapts to time changes. This paper proposes a novel price-demand model designed for a cloud cache and a dynamic pricing scheme for queries executed in the cloud cache. The pricing solution employs a novel method that estimates the correlations of the cache services in an time-efficient manner. The experimental study shows the efficiency of the solution.
References


[22] Jiadao Li, Ramin Yahyapour, "Negotiation model supporting co-allocation for grid scheduling", In ICGC'06.


[31] "Series of meetings of the EPFL-IC-IIIF-DIAS lab with the data management group of the European Organization for Nuclear Research (CERN) started on the December 9th, 2008.


Mr. RAJ DEEPAK POLUMATI is a student of NOVA college of engineering & Technology, Jangareddygudem. Presently he is pursing his M.TECH from this college and He received graduation from JNTU University In the year 2005.
Mr. K.JOHN PAUL, well known Author and excellent teacher Received B.Tech from jntu hyderabad, and M. Tech (CSE) from jntu kakinada is working as Associate Professor and HOD, Department of IT, Nova college of Engineering and Technology, he has 5 years of teaching experience in various engineering colleges. To his credit couple of publications both national and international conferences journals. His area of Interest includes Data Base management systems, computer networks, information security, flavors of Unix Operating systems and other advances in computer applications.