CACHING SERVICE IN CLOUD DATA

Ms.P.Gayathri,
Associate Professor, Dept. of IT
Bharath University, Chennai -600073

ABSTRACT

One of the emerging applications in the cloud computing is data management service. The cloud promises user satisfaction through quality of service and cost profit through adoption of 'pay for use model'. Caching at the back end provides quick access. Querying a massive data insists the implementation of caching for quick performance and effective resource utilization. Caching process will ensure that the query operation is performed over readily available data and its supported data structure. Static pricing strategy is against the basic characteristics of cloud, since cloud assures on-demand, on-command access with pay-use model. Cloud management should address both service economy and resource economy in a multi user environment with the guarantee of high customer satisfaction. The utilization and usability factors such as economy and customer satisfaction influence the adoption of optimal dynamic pricing through proper cache management.

Our study proposes an enhanced price-demand model for maximizing the utilization of cloud cache with reduced cost in a time-efficient manner. The experimental study shows the efficiency of the solution.

Keywords: cloud data caching service in massive querying.

I. INTRODUCTION

The leading trend for service infrastructures in the IT domain is called *cloud computing,* a style of computing that allows users to access information services. Cloud providers trade their services on cloud resources for money. The quality of services that the users receive depends on the utilization of the resources. The operation cost of used resources is amortized through user payments. Cloud resources can be anything, from infrastructure (CPU, memory, bandwidth, network), to platforms and applications deployed on the infrastructure.

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. 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 and Microsoft Azure have begun to offer data management services: the cloud enables the users to manage the data of back-end databases in a transparent manner. Applications that collect and query massive data, like those supported by CERN , need a caching service, which can be provided by the cloud.

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. Figure 1 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. Aprice 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.

II. EXISITING SYSTEM:

There are many cloud data service providers in the market. They focus data service either through provision of web services as provided by Amazon Elastic Compute Cloud (EC2), or through the support of server deployment given by Go Grid. Both web service and server deployment face two challenges while defining cache optimal pricing scheme.

(i) Need of balanced simplified model instead of oversimplified for achieving optimized pricing solution in terms of price demand dependency. Since a static pricing has a major shortcoming and it is not optimal when the query demand request has high fluctuations during peak seasons.

(ii) Also, a pricing scheme should address and adaptable to model errors, time- dependent model changes and adjust with the stochastic behavior of application. The servicedemand is always depends and influenced by the socioeconomic situations and external environmental factors.

For example, all the web applications need massive product image storage and network access during New

Year season. The online sale will be enormous during the festive season. The high hit ratio to the websites will cause failure to process the requested huge number of queries and introduce performance degradation. Also, it is not affordable to pay for each query access during the seasonal offers. This necessitates companies to adopt adopted cloud storage strategy to store the product image to maximize the service and efficiency and hence reduce the query delay. Static pricing cannot guarantee such cloud profit maximization. Static pricing for the above sales application will result into unpredictable, uncontrollable behavior and increase the pay-use amount. The storage of images and sales data in a distributed way as suggested in sales applications, recommends the adoption of distributed querying scheme with limited budget solutions to users. In short, the disadvantages of static pricing is listed as

- Static pricing is not optimal for elastic demand
- Static pricing makes fluctuations and collapse the predicted profit behavior.

III. PROPOSED SYSTEM:

Incorporation of caching minimizes query processing time at the back end and reduces network traffic and optimizes profit. The expected optimal pricing scheme can be achieved only by incorporating the correlation of existing cache service structures. Also, the pricing scheme should be dynamic to query request and time changes. The two important factors in adopting cloud data caching and prize optimization are:

(i) Price adaptivity to time changes: A finite long term horizon includes sequential non-overlapping intervals for scheduling. This concepts helps profit optimization in the following manner. The cloud is allowed to redefine the data availability in beginning of each interval with the comparison of both offline and online availability. Meanwhile, pricing proceeds in sliding time-window iterations for any online corrections and updations of new data. The demand of online data can be predicted and corrections are permitted at each sliding instant. It is also easy to predict the requirement of re-definition of the parameters.

(ii) Modeling structure correlations: Each pair of online and offline structures are compared for two:

- Competitive – No correlation and similarities and more deviations
- Collaborative – High correlation and similarities between the two.

The combination of index and column pairs are used to estimate the correlation measures and to find the type of correlation between online and offline data. The model proposed in our study is efficient computation of correlation of cache structure for cost optimization and suggest work-load compression technique based on template. The experimental study of dynamic pricing through caching will be carried out with the focus of :

- Minimize the query access, network traffic and achieve load balancing
- On demand dynamic cost optimization pricing model to optimize profit and elicit customer satisfaction.
- Balance price and time change through non-linear programming and provide efficient solution
- Increase computational efficiency through a correlation measure and dynamic pricing scheme.

IV. COST MODEL

The cost *C* of a query plan *PQ* is the sum of the cost of executing the query plan, *Ce*(*PQ*) and the amortized cost of any structure *S*€*S* used by the query plan, *Ca*(*PQ*). $C(PQ) = Ce(P_Q) + Ca(P_Q)$ (1)

The execution cost of a query plan is analyzed in Section V-B. The amortized cost of a query plan comprises the respective cost for all the structures employed:

$$Ca(PQ) = \sum_{s \in S} Ca(S) \quad (2)$$

The amortized cost of a structure *S* depends on the initial infrastructure cost that is necessary to build it, *BuildS*(*S*). This dependence is expressed by a function that also considers the number of queries *n* that benefit from each *S*, and to which the initial building cost is disseminated to:

$$Ca(I) = fS(n, BuildS(S)) \quad (3)$$

The building cost *BuildS*(*S*) is analyzed in Section V-C. The function *fS* quantifies the manner of cost amortization to the *n* queries that use *S*. In this work we consider that the initial building cost of *S* is amortized equally to the *n* queries, thus:

$$fS(n, BuildS(S)) = BuildS(S)/n \quad (4)$$

Selecting *n* is a challenging problem in itself, as it depends on the provider's risk aversion, arrival pattern of the queries, and infrastructure costs. We intend to study this problem in our future research

V. CLOUD INFRASTRCTURE

The architecture of the cloud which incorporates the proposed economy is shown in Figure 1. The user requests for query execution from the Internet and contacts the Coordinator node. The latter distributes the query to the appropriate CPU node, or to the back-end databases. We assume that the cloud infrastructure

provides unlimited amount of storage space, CPU nodes, and very high speed intra-cloud networking. Also, the CPU nodes in the architecture are all identical to each other. Compared to TCP bandwidth on the Internet, the inter cloud bandwidth is orders of magnitude faster and we ignore the overhead associated with it. The storage system is based on a clustered file system, such as, where the disk blocks are replicated and stored close the CPU nodes accessing them. With this infrastructure we can assume that the virtual disk is a shared resource for all the CPU nodes in the cache.

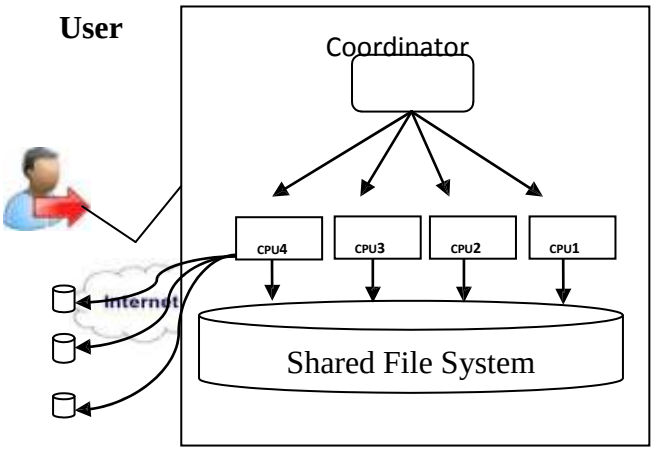

Figure 1.Cloud Infrastructure

VI. QUERY EXECUTION MODEL

The cloud cache is a lull-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 the necessity of such a cloud data service provider derives from the data management needs of huge analytical data, such as scientific data , for example physics data from CERN and astronomy data from SDSS . 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, themore 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.

VII. CONCLUSION

In this paper, we propose an economic model for a cloud cache suitable for the querying service of large scientific datasets. The proposed economy is self-tuned to three policies. These ensure high and increasing quality of individual and overall query service, but also, guarantee profit for the cloud. The economy is based on a cost model that takes into account all the necessary infrastructure resources, namely: network bandwidth, disk space and CPU time. The cloud profit from the query services is invested in building physical structures in the cache, which expedite query execution. The presented experimental study shows that the proposed economy is viable for a variety of workloads and data.

REFERENCES:

1. Samrat Bhattacharjee, Kenneth L. Calvert, and Ellen W. Zegura. Selforganizing wide-area network caches. In *IEEE Infocom'98*, 1998.
2. Chunming Chen, Muthucumaru Maheswaran, and Michel Toulouse. Supporting co-allocation in an auctioning-based resource allocator for grid systems. In *IPDPS*, pages 89–96, 2002.
3. http://aws.amazon.com/ec2/.
4. http://code.google.com/appengine/.
5. http://www.sdss.org/.