*International Journal of Advanced Computer Research (ISSN (print): 2249-7277 ISSN (online): 2277-7970)*
*Volume-2 Number-4 Issue-6 December-2012*

# Economic Price Estimation for cloud business

**Deepak Mishra[1] , Manish Shrivastava[2]**
Information Technology, LNCT Bhopal ,Bhopal (M.P.), India

## Abstract

***A cloud server connection consists of an occurrence of shared database architecture server and at least one front-end network server. When users request data from cloud server, the cloud application, running on the front-end network server, retrieves all the relevant data from backend to handle the manipulator request. Although this guarantees that all manipulators always see and find out the most up-to-date data, he has the weakness of needful a sub-optimal quantity of statistics traffic between cloud server and the front-end network servers and consequently hypothetically sub-optimal performance and responsiveness to user requests, in addition to imperfect scalability of the hardware to achieve higher throughput heights. The cloud server uses caches built information retrieval and caches run on the front-end network servers of your cloud server farm. Every cache upholds reproductions of data nearby on the front-end network server so that cloud manipulator supplies can be checked by using the cached data and wherever applicable, circumventing unnecessary circulation amid backend server and the front-end network servers and CPU data management on the front-end network server, thus refining performance and scalability.***

## Keywords

***front-end network, data management, refining performance***

## 1. Introduction

The cloud is becoming increasingly pervasive and mobile browsers are getting better every day, providing the ability to access the cloud and its applications Organizations are deploying more and more SaaS-based applications and, assuming they have enough bandwidth, there is no reason that mobile workers cannot access those applications on their devices with a browser that can actually fully handle web and cloud standards.

The goal of cloud based resources is to optimize: 1)Enabling self-services across open system interconnection, 2)Just-in-time Infrastructure, 3) More efficient resource utilization, 4) Usage-based costing, 5) Potential for shrinking the processing time, 6) User gratification, 7) cloud profit for offered services.

Although the success of the cloud provision depends on the optimization of above seven objectives, to exploring businesses on cloud should be typically prioritize profit and to maximization of cloud profit we need optimal pricing methodology with some Key feature to construction an economical application is using on-demand resources in your proposal. It's inefficient to compensation for infrastructure that is sitting idle which is adoptable, Price optimization technique is giving cloud offered services in minimum cost.

## 2. Related Work

Cloud businesses may offer their services for free, such as Google Apps and Microsoft Azure or based on a pricing scheme. Amazon Web Service (AWS) clouds include separate prices for infrastructure elements, i.e., disk space, CPU, I/O, and bandwidth [2], Pricing schemes are static, and give the option for pay as-you-go [2]. Static pricing cannot guarantee cloud profit maximization, static pricing results in an unpredictable and, therefore, uncontrollable behaviour of profit [2].
The problem of revenue management through dynamic pricing is well studied [2]. Based on the rationale that price and demand are dependent qualities, numerous variations of the problem have been tackled, for instance businesses that sell products to retailers, seasonal products, stochastic demand [2]. Electronic businesses, and therefore cloud businesses can benefit from dynamic pricing policies [2]. Cache services are distinguished from consumable products in two major ways: 1) they are not exhausted while they are consumed, and 2) the demand for a specific service pauses while this is not available [2]. To the best of our knowledge, this is the first work that tackles the problem of optimal

pricing of competitive data services within the same cloud cache provider [2].

## 3. Our Proposal

The cloud server uses caches built information retrieval and caches run on the front-end network servers of your cloud server farm. Every cache upholds reproductions of data nearby on the front-end network server so that cloud manipulator supplies can be checked by using the cached data and wherever appropriate, circumventing unnecessary circulation amid backend server and the front-end network servers and CPU data management on the front-end network server, thus refining performance and scalability.

### 3.1 Dynamic Cooperative caches modelling

Dynamic cooperative caches provides top four benefits this is quickness, price, virtualized managed cloud client loads, cost effective prediction price. Users can be up and running in minutes instead of weeks or months and this will come from parallelism of dynamic load balance distribution by using elastically scalable grid architecture, since cloud computing is pay for only what you use, functions at high scale and is extremely automated, the price and effectiveness of cloud computing is very compelling as well. if let-down condition will come, then this one is effecting only sharing caches file, By using caching method getting highly benefits like 1)On-demand self-service, 2)Broad network access, 3)Rapid elasticity, 4)Measured service, 5)Elastic scalability, 6) Low upfront costs, 7)Economies of scale, 8)Operating expense, 9)Simpler to manage, 10)Greater control of security, compliance and quality of service, 11)Resource pooling. So caching model is removed unpredictable demand patterns problems because soft computing method is doing scaling up or scaling down of resources for a given application on demand (as shown in figure 1).

### 3.2 System Architecture

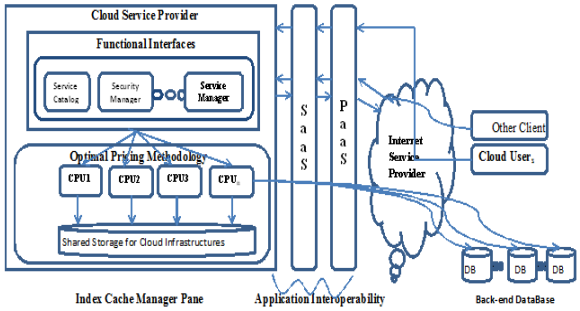


**Figure 1: System Architecture for optimal service pricing for cloud based services**

When user requesting the item information, cloud environment to either select information that has been cached on the machine you are working on or create a new cached for a particular item and add them to cache structure. If the information string is already presented on cached, simply select that information from the list and to add it on the list of cache structure in the **index cache managers pane**, If a information does not exist in the cache, information is fetched from the backend, Cache Transform Allows data fetching and its related information is periodically updated on **index cache managers pane** online/Offline and that is used for full integration with your favourite third-party and custom applications services for multiple lookup components, The optimal pricing methodology is keeping control of query fetching and receiving process and also provide elastic caching platform( as shown in figure 2).

### 3.3 Query Posing/Retrieving Architecture

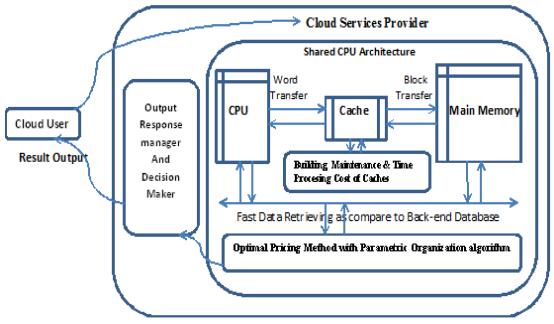


**Figure 2: Query Posing/Retrieving Architecture for cloud based offered services**

To combine/retrieving data from different caches file that exists could uses the Merge Join Transformation, which combines datasets by joining the rows across a set of common keys [4]. This transformation allows an inner join, a left outer join, or a full outer join [4]. You could also use a Lookup Transformation to associate data from two sources [4]. The Lookup can cache a table in memory and, through matching columns, can return new columns to the data flow [4] [1].

## 4. Contribution

1. By using cloud cache structure (index cache managers pane), getting two primary benefits:
   a. It moves static objects closer to the visitor requesting them, which makes their distribution faster.
   b. It decreases loads on the origin web server. Caching plays a big part in how we are talented to, on regular; decrease server loads, Network bandwidth costs, prediction price estimation and page load times by more than half.
2. Cloud network is highly trained for updating his caches file with different related cloud network caches & periodically updated his caches file; this structure is giving fast retrieving information as comparison to backend.
3. Cloud Optimization technique system as an elastic caching platform, and support for database Consistency, scalability for data stores.
4. Soft computing technique is giving parallel and distributing cooperative caches and which are done by taking self-decision when any user poses a query on cloud, for finding information about price prediction for particular demand at a time and model is highly trained as self-tuning by using cooperative caches (index cache managers pane) data.

## 5. Proposed System

Requesting information from the cloud can be cached on Index_Cache_Managers_Pane (as show in Figure 1.), allowing for quick retrieval of frequently used satisfied. An item cache is twisted and achieved using Cloud Server Manager. One content cache can exist for each server that has the Content Cache Service connected. An organizational helpfulness is provided for monitoring the cache, this is also a powerful driver for cloud computing. The ideal platform is small exclusive to size. The servers, networking equipment, data storage/backup, power [1], redundant high-speed connectivity, etc. can result in a huge start-up cost for a single product or project [3]. Add to this the fact that most development efforts fail, and the economics simply don't make sense for investment of this level in each project [3]. Cloud computing enjoys economies of scale, because that same investment can be amortized over a large number of projects [3]. If one project fails, it can be replaced by a number of new projects that continue to amortize the initial investment **[3],** in a cloud environment, backup is highly automated, whereby that same IT person can oversee simultaneous backups for hundreds or thousands of customers [3].

**Algorithm: Query Execution Model for Optimal Pricing**

**Global:** Query q, ItemName N, Price P, PriceRange_ListItem P1, Index_Cache_Managers_Pane C, Optional = PredictionPrice P2,ToatlPrice P3.

QueryExecution(N,P,P1,P2),DecisionParameters(ChecqLogic),PredictionPrice(ChecqLogic),QueryChequeResult(ChecqLogic),OptimalPricing(P,P1,P2,OfferedPrice,OfferedItems,SeasionalRelaxation,TotalPrice),ItemAvailability(),AddNewCachesStructures(q)
**{**
**1**. q is firstly forward to **ControllerQueryChecquer()** & they will taking the decision with check conditions of query information(q) like ItemName, PriceRange_ListItem, PredictionPrice etc. */* Here two condition will come, which are important either user want prediction price Yes/No, because this decision will charge extra cost to user. */*
**2.If**(ControllerQueryChecquer(q(ItemName))←runQueryfromcaches(q(ItemName))) OR q(ItemName))←runQueryfromDatabase(q(ItemName)))And DecisionParameters()←'No' */*If user query q of Item Name is not present on cloud Cache & Cloud Database*/*
**{**
**2.1** ItemAvailability**()**←'No'
*/*This item is not available On cloud List*/*
**2.2** Return ItemAvailability()
*/*No Such a Item is present on cloud list*/*
**2.3** Return DecisionParameters() *//Required Query Item is not present*
**}** */*end 2nd Level*/*
**3. Else If** (**DecisionParameters ()** ←'Yes') then
**{ 3.1. If** (**PredictionPrice()**←**'**Yes') then
{ **3.1.2 ControllerQueryChecquer()** make a list of informationretrieval ( N, P1, P2, OfferedPrice,

OfferedItems, SeasionalRelaxation, P3) and send to Next Phase /* **ControllerQueryChecquer(List), List**← N,P1, P2, OfferedPrice,OfferedItems, SeasionalRelaxation, P3 ***/**

3**.1.3. If ControllerQueryChecquer(List)** related information is present on Index_Cache_Managers_Pane ( ) Then QueryChequeResult()←'Yes' {

**3.1.3.1**PredictionPrice()←Neuro-GeneticPrice ( runQueryfromCaches (q) )

**3.1.3.2**OptimalPricing(P,P1,P2,OfferedPrice,OfferedItems,SeasionalRelaxation,P3)←runQueryfromCaches(q)

**3.1.3.3Return** OptimalPricing(P,P1,P2,OfferedPrice, OfferedItems,SeasionalRelaxation,P3)

**3.1.3.4 Return** QueryChequeResult() */*Yes*/*

**3.1.3.5 Return** PredictionPrice() */*Yes*/*

**3.1.3.6 Return DecisionParameters()** */*Yes*/*

**}***/*3.1.3close If ControllerQueryChecquer (List) from Caches to search*/*

**3.1.4.** Else(QueryChequeResult()← No) */*Search From beckend with prediction price */*

**{** */*Optimal Pricing depends on the time interval of query fetching on the backend, Network accessibility cost with depends on the cloud administrator & how they will minimize the price of item. */*

**3.1.4.1**PredictionPrice()←Neuro-GeneticPrice ( runQueryfromCaches (q) )

**3.1.4.2**OptimalPricing(P,P1,P2,OfferedPrice,OfferedItems,SeasionalRelaxation,P3)←runQueryfromDatabase(q)

**3.1.4.3**C←AddNewCachesStructures(runQueryfrom Database(q))

**3.1.4.4**Return OptimalPricing (P, P1, P2, OfferedPrice, OfferedItems, SeasionalRelaxation, P3)

**3.1.4.5** Return QueryChequeResult() */*No*/*

**3.1.4.6** Retrurn PredictionPrice() */*yes*/*

**}** */*3.1.4 Close Else Part, Else(QueryChequeResult()← No) with prediction price from backend*/*

**3.1.5** Return **DecisionParameters ()** */*Yes*/*

**3.1.6** Return ItemAvailability() */*Yes*/*

**}** */*3.1 Close If (PredictionPrice()←'Yes') */*

***3.2. Else*** **(PredictionPrice()←'No')** then */*3.2 Else Part start If (PredictionPrice()←'No') */*

**{**

**3.2.1 ControllerQueryChecquer()** making a list of information retrieval(N,P1, P2, OfferedPrice, OfferedItems, SeasionalRelaxation, P3) and send to Next Phase.

*/*ControllerQueryChecquer(List),List←N,P1, P2, OfferedPrice,OfferedItems, SeasionalRelaxation, P3 */*

**3.2.2** Set PredictionPrice()←'Null'

**3.2.3. If ControllerQueryChecquer(List)** related information is present on Index_Cache_Managers_Pane ()

Then QueryChequeResult() ← 'Yes' **{**

**3.2.3.1**OptimalPricing(P,P1,P2,OfferedPrice,OfferedItems,SeasionalRelaxation,P3)←runQueryfromCaches(q)

**3.2.3.2 Return** OptimalPricing(P, P1, P2, OfferedPrice ,OfferedItems,SeasionalRelaxation, P3)

**3.2.3.3 Return** QueryChequeResult() */*Yes*/*

**3.2.3.4 Return** PredictionPrice() */*No*/*

}*/*Close 3.2.3 of If Part serches from cloud cache */*

***3.2.4***. Else (QueryChequeResult()← No) */*Start search from backend with no prediction price*/*

{

*/*Optimal Pricing depends on the time interval of query fetching on the backend, Network accessibility cost with depends on the cloud administrator & how they will minimize the price of item. */*

**3.2.4.**1OptimalPricing(P,P1,P2,OfferedPrice,OfferedItems,SeasionalRelaxation,P3)←runQueryfromDatabase(q)

**3.2.4.1**C←AddNewCachesStructures(runQueryfrom Database(q))

**3.2.4.2**Return OptimalPricing( P, P1, P2, OfferedPrice, OfferedItems, SeasionalRelaxation, P3)

**3.2.4.3.** Return QueryChequeResult() */*No*/*

**3.2.4.4** Retrurn PredictionPrice() */*No*/*

**3.2.4.5**. Return **DecisionParameters ()** */*yes*/*

} */*3.2.4 Closed Loop*/*

**3.2.5** Return **DecisionParameters ()** */*yes*/*

**3.2.6** Return ItemAvailability() */*yes*/*

*}/*3.2. Close Here loop , Else* (**PredictionPrice()←'**No') then */*

**}** */*3. Loop will end here , Else (DecisionParameters () ←'Yes') then */*

**Main()**

**{**

**1**.cloud user forward query to the Cloud server, **QueryExecution**(N,P,P1,P2)←q

**2.**If ( ItemAvailability**()**←'No' And DecisionParameters**()** ← 'No') then

**2.1** ResultOutput(): No such Item Information is present on the cloud list.**3**.Else If(QueryChequeResult**()**←'Yes',PredictionPrice**()**←'Yes',**DecisionParameters()**←'Yes') then

**3.2**ResultOutput(): OptimalPricing(P, P1, P2, OfferedPrice, OfferedItems, SeasionalRelaxation, P3)

*/*This will come from Cloud Caches or Index_Cache_Managers_Pane with prediction Price*/*
**4.**ElseIf(QueryChequeResult**()**←'No',PredictionPrice**()**←'Yes', **DecisionParameters()**←'Yes')
**4.1**ResultOutput(): OptimalPricing(P, P1, P2, OfferedPrice, OfferedItems, SeasionalRelaxation, P3)
*/*This will come from Cloud Backend or Primary Storage with prediction Price */*
**5.**Else If (QueryChequeResult**()**←'Yes', PredictionPrice**()** ←'No', **DecisionParameters()** ←'Yes' ) then
**5.1**ResultOutput():OptimalPricing(P,P1,P2,OfferedPrice,OfferedItems,SeasionalRelaxation, P3)
*/*This will come from Cloud Caches or Index_Cache_Managers_Pane */*
**6**.ElseIf(QueryChequeResult**()**←'No',PredictionPrice**()**←'No',**DecisionParameters()**←'Yes')
**6.1**ResultOutput():OptimalPricing(P,P1,P2,OfferedPrice,OfferedItems,SeasionalRelaxation, P3)*/*This will come from Cloud Backend or Primary Storage */*
**}**
The goal of the cloud cache is to offer cheap efficient multiuser querying on the back-end data [2], while keeping the cloud provider profitable Users pose queries to the cloud, which are charged in order to be served [2], As a result, operation rates of several thousand queries executed in per seconds can be realized with just a single cloud virtual business controller. Cloud virtual business controller springs whole, out-of-the-box construction for generating scalable data requests. We switch all the throw, gather, failover retrieval, logging, prediction and data planning by using caches structure (Index_Cache_Managers_Pane) and last user will satisfy and getting profit.

## 6. Experimental Result

The goal of cloud is to provide efficient querying on the back-end data at a little charge, although existence economically feasible, and additionally, gainful. Users pose queries to the cloud through an IaaS by using PaaS & SaaS are charged on-the-go in order to be served. The cloud caches data are come figures data constructions in order to quicken request finishing. Service of queries is performed by executing them either in the cloud caches (if compulsory facts are previously cached) or in a back-end database. Each caches structure (data or data structures) has an operating (i.e., a Building, Maintenance & Network Access) cost. A price over the Operating cost for each structure can ensure profit for the cloud & by this we are getting practical results & the prediction process is going on future work.

### 6.1 Function description of Cloud Parameters (Admin or Clients side)

**Table 6.1: Basic function descriptions of Cloud Parameters (Admin or Clients side)**

| S. N. | Cloud Admin (a) | Cloud Client Users (b) |
|---|---|---|
| 1 | Home Page | Home Page |
| 2 | WebPageInfo→UploadCloudData | CloudData→RetrieveCloudData(Here Cloud client Purchsing items By command Go) |
| 3 | UserInfo → ViewCustomers | AccountInfo→ViewBalance AccountInfo→Deposite |
| 4 | Security→ChangePassword | SearchInfo→GetMySearchInfo(Search records with optimal cost pay by client) |
| 5 | Logout | Profile→ViewMyProfile |
| 6 | | Security→ChangePassword |
| 7 | | Logout |

### 6.2 Optimal cost estimation method for Cloud Clients

1. Cloud parameter is inbuilt apply methods for sorting the sequence of searching content or items, keeping information on his search status wise, that is known as queue list, This queue is keep information in order of FIFO(First in first out).
2. Every registered client is automatically come on queue list and Inbuilt Optimal Cost estimation method is automatically sorting this queue list by using following procedure:

a. Every Purchased items is sorted by client wise.
b. Every client is sorted by according to Purchased items, for sorting we are using some unique key parameter constraint Date, Time, User ID and DOB and Content Name;

3. This queue list is keep information only a) Content Name, b) User Name, c) User ID, Minimum Reduction Tax.
4. Every purchased item of Reduction Tax (Sorted Optimal Reduction Tax) is divided by average number of registered client is

searching that item and by this we are getting a new balanced profit (Optimal profit).

5. When optimal method is obtained balanced profit then these profits (Optimal profit) will is automatically posting on client accounts according to queue list and by this every client will getting optimal price of purchased items.

Suppose uploaded data 'School tifin', Client want to purchase, Firstly client will type content name on text box (explain on above client section like **CloudData→RetrieveCloudData**) and using 'Go' Command, This query information is retrieved either Cache or Backend. when first time client is poses query for purchasing item on cloud services market then query is passes to decision maker(cloud server) & decision maker is firstly execute query related information on cache file, If no any information is available on cache file, then this are executing on backend(as shown in below table 6.1).

**Table 6.1: Service Tax is retrieved from cloud database**

| SITN | ST | RST | PAMT | TPAMT | PDate | Data From |
|---|---|---|---|---|---|---|
| STI | 5.0 | 0.0 | 5.0 | 5.0 | 2012-05-20 09:13:05.0 | CloudDatabase |

As show on below optimal profit table (table 6.2) purchased item of practical output ('School tifin'), First time query content item is retrieved from Backend because defined item is previously not processed by any client, But the second time for same content, information is retrieve from the cache data file, because first time processed item is saved on cache file.

**Table 6.2: Optimal profit information of purchased items**

| SITN | ST | RST | PAMT | TPA | PDate | Data From |
|---|---|---|---|---|---|---|
| STI | 5.0 | 2.0 | 3.0 | 8.0 | 2012-05-20 09:13:31.0 | CacheDatabase |
| STI | 3.0 | 1.2 | 1.8 | 6.8 | 2012-05-20 09:14:10.0 | CacheDatabase |
| STI | 1.8 | 0.72 | 1.08 | 6.08 | 2012-05-20 09:14:35.0 | CacheDatabase |
| STI | 1.08 | 0.432000000000000005 | 0.648 | 5.648 | 2012-05-20 09:15:00.0 | CacheDatabase |
| STI | 0.648 | 0.259200000000000004 | 0.3888 | 5.3888 | 2012-05-20 09:15:17.0 | CacheDatabase |
| STI | 0.3888 | 0.155552 | 0.23328 | 5.23328 | 2012-05-20 09:15:28.0 | CacheDatabase |
| STI | 0.23328 | 0.093311999999999999 | 0.139967999999999998 | 5.139968 | 2012-05-20 09:15:38.0 | CacheDatabase |
| STI | 0.13996799999999998 | 0.05598719999999999994 | 0.08398808 | 5.08398808 | 2012-05-20 09:15:46.0 | CacheDatabase |

**6.3 Table (6.1, 6.2) Attribute (Column Name) Description:**

1. SITN - SearchItemName
2. ST - ServiceTax,

3. RST - ReductionServiceTax
4. PAMT - PayingAmount
5. TPA - TotalPayAmount
6. PDate - PayDate
7. DataFrom- CACHE(Optimal) / Backend ( Cloud Owner Decide Price ).

### 6.4 Table Tuple (Row Name) Description:

**1. STI – 'SchoolTiffin'.**

Cloud user will get optimal price because cloud cache is to offer cheap efficient multiuser querying on the back-end data, as shown on table 6.3, in the form of increasing order (RT – Reduction Tax).

**Table 6.3 Reduction Service Tax**

| | | | | | |
|---|---|---|---|---|---|
| RT | 0.0 | 2.0 | 1.2 | 0.72 | 0.43200000000000005 |

## 7. Conclusion

One of the best advantages is that first time retrieving information is came from Backend, so cloud is charged actual Service Tax on first client but same process is done by same or another client is then Service tax is come optimal, So Result output for first user will pay actual defined Service Tax and another pay optimal cost, Solving this cost estimation for first user or same for next user; Cloud owner is here apply Parameter cost estimation method for optimal price to every registered cloud clients, 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 handler). The cloud caches regulates which arrangements (cached columns, views, indexes, previous demand price, offered price, future price reduction probability, offered price correlation.

## References

[1]. John W. Rittinghouse, James F. Ransome, "Cloud Computing Implementation Management and Security", CRC Press Taylor & Francis Group, ISBN: 978-1-4398-0680-7;

[2]. Verena Kantere, Debabrata Dash, Gregory Francois, Sofia Kyriakopoulou and Anastasia Ailamaki, "Optimal Service Pricing for a Cloud Cache", IEEE Transaction On Knowledge & Data Engineering, VOL. 23, NO.9, SEPTEMBER 2011.

[3]. Mike Hogan, "Cloud Computing & Databases", November 14, 2008.

[4]. Kevin Loney, "Oracle Database 10g", Tata McGraw-Hill Publishing company Ltd, New Delhi.

Received the degree BE (CSE) in 2007 and doing regular MTECH (IT) from the LNCT Bhopal, After completion of my bachelor's degree, I am selected for the post of programmer on State Unit Raipur Mantralay, NIC Headquarter (CG) & here my work period is from 2007 to 2009, Again Join Shwetark software solution as a system administrator designation and here duration of my work is from July-2009 to August-2010.