FINANCIAL CRYPTOGRAPHY: DISCRIMINATORY PRICING MECHANISM

Sumit Chakraborty
Fellow, Indian Institute of Management Calcutta, BEE (Jadavpur University), India
E-mail: schakraborty2010@hotmail.com, surya20046@yahoo.co.in
Phone : 91-9940433441

Abstract: This work presents an adaptive profitable discriminatory pricing mechanism for cloud computing based on secure function decomposition, cryptographic commitments and zero knowledge proof. Cloud computing is an emerging trend of enterprise resource planning where a selling agent or service provider (S) wants to allocate a set of computational resources and related IT services optimally and fairly among many buying agents or service consumers (B) within its capacity constraint. Each service consumer discloses its demand plan for an IT portfolio within its budget constraint and rank of preference. An IT portfolio may include SaaS, PaaS, IaaS, CaaS, DaaS and dSaaS. The basic objective of the service provider is to optimize its expected revenue within target profit margin. It is basically a problem of secure function evaluation where the concept of decomposition of a function is considered. It is a constrained nonlinear optimization problem; the search is governed by a set of intelligent moves. The communication complexity of the pricing mechanism depends on the time constraint of the negotiating agents, their information state and the number of negotiation issues; it also depends on number of negotiation rounds and the complexity of IT portfolio. The computational cost depends on the complexity of function decomposition. The security and privacy of strategic data of the trading agents provides business intelligence to the pricing mechanism. The ultimate objective of the mechanism is to predict a profitable discriminatory pricing plan for each consumer.

[Categories and Subject Descriptors] Pricing algorithm
[General Terms] Algorithmic mechanism
Keywords: Nonlinear discriminatory pricing mechanism, Secure function decomposition, Cryptographic commitment, Zero knowledge proof, Computational intelligence, Cloud computing.

1. INTRODUCTION

Recently, the domain of algorithmic pricing has been emerged as a means of resource allocation in distributed setting for constrained optimization problems [5,6,7,20,21]. This area has important link with the domain of algorithmic mechanism design. The issue of optimal mechanism design and profit maximizing auction setting has been discussed in [15]. Algorithmic pricing is a specific type of computational problem where a selling agent tries to find out the price vector for a set of products or services to maximize its revenue within target profit margin in the presence of known demand of a set of buying agents. The selling agent distributes its resources to the buying agents through a fair allocation protocol. This motivates the search for fair and mutually beneficial agreements where algorithmic theory gives an improved understanding of pricing. In order to structure, analyze, and ultimately resolve the complexity of pricing, an important and useful concept is decomposition. Partitioning a large and complex system or decision problem into interrelated but independent subsystems or sub-problems can reduce dimensionality, increase flexibility and facilitate overall modeling, computational and decisional requirements. It is basically a problem of secure function evaluation where the concept of decomposition of a function with minimum disclosure is considered [12].

Strategic pricing and revenue optimization is increasingly becoming a core competency of trading in a smart market of information and communication technology. The basic objective of strategic pricing is to set and update the prices for each type of product, service and customer segment in a profitable way. Pricing is a crucial business decision since a minor adjustment of pricing parameters has significant impact on the revenue and profitability, the diffusion and life-cycle of the offered products and services [19]. In this connection, several decisions are important: what to price, when to price and how to price. It involves a complex set of decisions to be managed in an integrated way. Traditional pricing mechanisms are based on cost, market share and value driven approaches. Optimization plays a critical role in pricing and revenue management [9]. Pricing decisions are actually non-linear constrained optimization problems which can be solved with the support of financial economics, computer science and operations research [30]. This work considers the pricing problem of cloud computing.

With the significant advancement of information and communication technology, computing is perceived to be used as the next utility after water, electricity, gas and telecommunication. The concept can be extended to cloud computing and grid computing for a market oriented grid. Utility computing is associated with a parallel and distributed system that enables the sharing, selection and aggregation of geographically distributed autonomous computational resources dynamically at runtime depending on their availability, capability, performance, cost and quality through web service [3]. The computational resources include different types of sophisticated software applications such as data mining, scientific computing and image processing, data, CPU or processing power, servers, storage devices, scanners, UPS and network interfaces which can be shared through web service. The objective of utility computing is to provide computing power and storage capacity that can be used and reallocated for any application and billed on a pay-per-use basis. Utility computing consists of a virtualized pool of information systems and other IT resources that can be continually reallocated to meet changing business and service needs of the consumers. These resources can be located anywhere and managed internally or externally. The service provider tracks the usage of computational resources of the consumers and makes invoice based on predefined price setting and usage data. An efficient resource management system coordinates and monitors the complex operation.

Utility computing supports virtualization. Cloud computing is basically a distributed computing where dynamically scalable and virtualized resources are provided as a service over the internet to achieve cost saving, easy scalability and high availability. The services offered through cloud computing usually include Software-as-a-Service (SaaS), Infrastructure-as-a-service (IaaS), Platform-as-a-service (PaaS), data-Storage-as-a-Service (dSaaS) and database-as-a-service (DaaS) [11]. SaaS allows users to run applications remotely from the cloud. IaaS provides a set of computing resources as a service which includes virtualized computers with guaranteed processing power and reserved bandwidth for storage and Internet access. PaaS includes operating systems and required services for particular applications along with data security, backup and recovery, application hosting and scalable architecture. dSaaS provides data storage, data warehousing and data mining facilities. This is a cost effective, innovative IT infrastructure from which the consumers are able to access desired computational resources and from anywhere in the world on demand.

The key technologies that enable cloud computing are virtualization, web service, service oriented architecture, service flows and work flows. The trading in cloud computing depends on several technological issues such as high availability of service, business continuity, data lock-in, security and privacy of data, efficient data transfer, performance predictability, scalable storage, efficient bugs management in large distributed system, adaptive scaling of operation, innovative software licensing and reputation mechanisms [1]. Strategic pricing considers all these QoS factors to define optimal price setting for cloud computing. In fact, an intelligent, innovative competitive pricing mechanism and secured high QoS can make cloud computing an attractive IT business model as compared to traditional corporate computing model based on direct IT investment. Nowadays, pay-for-use or pay-as-you-go licensing are becoming popular in cloud computing market. Thus, the computing world is rapidly transforming towards developing information systems to be consumed as a service. Various service providers (e.g. Amazon, Google, IBM, Microsoft and Sun Microsystems) have started to build scalable data centers at various locations for hosting cloud computing.

The key players of the market of cloud computing are a set of service providers, service consumers and resource brokers. There are several challenges of trading in cloud computing : fair resource allocation protocols, optimal task scheduling, tendering, contract net protocols, auction, market clearing and negotiation mechanisms and pricing algorithms. The major threats are reduced contract duration, uncertainty, risk and variable duration of a portfolio of contracts, reduced switching costs and customer lock-in, uncertain customer demand, short life-cycle and high sunk cost. Cloud computing may require high development cost for instrumentation, provisioning and monitoring and start up costs in the face of uncertain demand.

The work is organized as follows. Section 2 defines the problem and reviews related works. Section 3 presents a pricing mechanism for cloud computing. Section 4 analyzes the mechanism from the perspectives of computational and communication complexity and information disclosure. Section 5 explains the computational intelligence of the pricing mechanism and section 6 concludes the work.

2. THE PROBLEM

Problem definition: We consider a cloud computing market with l types of IT services ($r_{j;\ j=1,\ldots,z}$), a selling agent or service provider (S) and n number of buying agents or service consumers ($B_i;\ _{i=1\ldots,n}$). The profile of S is indicated by (R, m, $\boldsymbol{C}$; $r_j,\ _{j=1,\ldots,z}$) where S can sell z types of IT services with a capacity constraint vector $\boldsymbol{C}$, revenue function R and target profit margin m. Each buying agent has a specific profile which is indicated by (d, b); d indicates the demand plan for an IT portfolio with a specific rank of preference and b is the budget constraint. An IT service portfolio include SaaS, PaaS, IaaS, CaaS, DaaS and dSaaS. The objective is to set a pricing plan (**P***) of the portfolio to maximize expected revenue R: $\mathbf{P^*} \in \max E[R(m, d, b)]$.

Assumptions: The pricing problem has a number of characteristics:

a. The trading agents act rationally. S follows a fair resource allocation protocol.

b. Privacy: Each service consumer discloses its profile to S. S does not disclose its revenue function, target profit margin, capacity constraint and reservation pricing plan to the buying agents. S discloses a *combinatorial price* for the desired IT portfolio to B. But, S does not disclose individual price settings for each portfolio sector such as SaaS, PaaS, IaaS and dSaaS precisely to B. The objective of S is not to disclose the competitive intelligence of IT portfolio. S can describe the information on IT portfolio indirectly by specifying pricing parameters with cryptographic commitments. Without de-committing the portfolio composition, the service provider can use zero knowledge proofs to reveal chosen features to the service consumers such as approximate allocation of IT investment, risk and future business opportunities to various sectors of IT portfolio.

c. The service consumers are assumed to be heterogeneous in their willingness to pay discriminately.

d. The market is assumed to be segmentable i.e. it is possible to identify different groups of buying agents in the market.

e. Arbitrage is limited in the market. The cost of strategic pricing is significantly low as compared to the profit.

f. The demand as a function of price is unknown a priori and is learned over time. Over time, a service provider is able to acquire knowledge regarding the demand pattern of the service consumers that can be utilized to improve the profitability of the provider. But, it is actually very complex to model the demand as a function of price precisely since the demand of the service consumers varies widely for different types of cloud computing services and the private and strategic data on the price-demand

functions of the competitors is not disclosed to the service provider.
g. The cloud computing services are priced *dynamically* over a finite time horizon since the demand and the data of the problem evolve dynamically. Most of the existing research works do not consider the profitable, adaptive, competitive and discriminatory aspects of the pricing problem jointly.
h. A smart market allows *competition* in an *oligopolistic* market, where the profit of a service provider not only depends on its price setting but also on the prices set by its competing providers.
i. The cloud computing services and resources are not *perishable;* there is no finite time limit to sell the resources and services after which any unused capacity is lost. The cost component has a significant impact on the decision-making process.
The problem of multi-product pricing has been solved by [20]. This work sets the prices of multiple products for the maximization of revenue of a seller; it presents a rank based price ladder for automotive business based on data mining. Dynamic pricing is the dynamic adjustment of the selling price to the buying agents depending upon various factors such as perceived value of the customers, market conditions, macro-economic factors and the financial health of the seller [9]. Price dispersion and price discrimination are two critical aspects of dynamic pricing [14]. Different selling agents offer a resource at different price in spatial price dispersion. The selling agent varies its price for a resource over time based on time of sale and supply-demand situation in temporal price dispersion. The other aspect of dynamic pricing is differential or discriminatory pricing where different prices are charged to different buyers for the same resource [2]. The selling agent sells different units of resource at different prices to different customers who value the resource most highly in first degree or perfect differentiation.
In cloud computing, the service providers can adopt different types of economic models such as autonomic metered pricing, posted pricing, auction, tendering, bargaining and contract net protocols [25,26,27,28]. A non-discriminatory risk based pricing mechanism was proposed in [16] for utility computing considering uncertainty in demand, high sunk cost and short product life-cycle. Another work [8] considers economic aspects of utility trading where high price is charged for improved quality of services. The works of [4,22] treat the pricing for utility computing as a problem of market clearing mechanism which adopts different types of strategies such as fixed, fixed time and flexible. Fixed mechanism charges a fixed price for different time slots. Fixed time mechanism charges different price for different time slots. The flexible pricing mechanism has two components: fixed and variable; it adjusts pricing parameters based on current distribution of workload across a set of nodes and the budget constraints of the consumers. It offers incentives to promote the demand of the users. Combinatorial auction is a type of set packing problem; the concept can be applicable to the trading of utility computing market. The issues of negotiation based pricing have been discussed in [3] through service level agreements. The focus of the pricing mechanism in [28] is to maximize the social welfare associated with the public cloud operations which is the aggregate service utility obtained by the cloud users less the cost of infrastructure and operation of the service provider. Studies on pricing in cloud computing are now at a preliminary stage. Existing works have not explored the scope of intelligent function decomposition with minimum disclosure.
Contributions: This work contributes to the literature of profit based pricing by identifying a discriminatory pricing mechanism for an adaptive business model of cloud computing. This is an extension of the works [7,37]. The mechanism solves the pricing problem through secure function decomposition with minimum disclosure. The basic objective is to improve the performance of pricing algorithm in terms of profit, stability and robustness. This is a nonlinear constrained optimization problem where the search is governed by a set of intelligent moves sequentially. The proposed pricing mechanism makes a trade-off between revenue and target profit margin and tries to optimize the revenue of the service provider within its target profit margin and capacity constraint. It not only predicts critical pricing parameters efficiently; but also mitigates risk. This is a dynamic pricing mechanism which is suitable for an adaptive business model. The pricing mechanism has been analyzed from the perspective of revelation principle, computational and communication complexity and algorithmic game theory. We develop a pricing mechanism while there is incomplete demand information. The mechanism assumes an approximate forecasted demand initially to compute a reference price ladder and finally adjusts the ladder based on the actual demand of the service consumers adaptively through intelligent negotiation. The modeling of demand learning is hard due to various factors such as stochasticity (in which the price elasticities are slowly varying functions of time) and non-availability of reliable data on the competitor's business.

3. PRICING MECHANISM

Mechanism design is the study of preference aggregation protocols that lead to good outcomes in spite of strategic behavior of the agents. The basic objective is to know how efficiently the preferences of the agents can be aggregated towards a social choice. The proposed pricing mechanism is an algorithmic mechanism which is defined by various types of elements: finite set of trading agents (i.e. service provider and service consumer), finite set of inputs given by each agent, finite state of outcome as defined by output function, utility function what each agent aims to optimize, objective functions and constraints, payments, a set of strategic moves and revelation principle. Absolute privacy or confidentiality may result an inefficient game. Therefore, the agents preserve the privacy of strategic data but share critical information. The mechanism is truthful since all the agents follow their strategies correctly. It should be a dominant strategy. A mechanism is strongly truthful if truth telling is the only dominant strategy. The mechanism ensures that individual interests of the trading agents are best served by their rational behavior. The mechanism allocates payments to the agents fairly. The payments are carefully selected to motivate all the agents to act rationally. In this sense, this is the dominant strategy of the service provider which maximizes its utility for all possible strategies of other agents involved in the mechanism. The following section presents an adaptive, profitable discriminatory pricing mechanism [APDPM] for cloud computing.

Agents: A service provider (S) and n number of service consumers (B_i, i =1,…,n) interact in a market for the allocation of l types IT services ($r_{j;\ j=1,\ldots,l}$). B_i may be single or multiple minded. S permits group buying. In case of cloud computing, S is the service provider and B is the service consumer. T is a trusted third party.

Input: A service consumer (B) has a specific input (**d**, b); **d** indicates the demand for an IT portfolio with a specific rank of preference and b is the budget constraint. An IT portfolio includes several sectors such as SaaS, PaaS, IaaS, dSaaS, DaaS and CaaS. S has a limited capacity of resources and services. S has an input (R, m, *C*, $r_{j,\ j=1,\ldots,l}$) where S can sell l types of IT services (e.g. SaaS, PaaS, IaaS, DaaS, dSaaS and CaaS) with a capacity constraint vector ***C***, revenue function R and target profit margin m. S can sell a single or a combination of IT services to B. S holds an intelligent dynamic pricing recommender system.

Strategic move :

a. S adopts a discriminatory service pricing strategy: pricing different cloud computing services differently for SaaS, PaaS, IaaS and DaaS.

b. S adopts a discriminatory traffic pricing strategy with *swing option* for dSaaS and CaaS: pricing data storage space consumed or the traffic transmitted for a specific period.

c. S computes a price ladder for each single and combinatorial or bundling services. S defines both simple and complex pricing schemes. The nonlinear pricing function is adjusted adaptively by applying intelligent analytics instead of following common sense or intuition based random heuristics.

d. A service consumer adopts public or private or hybrid cloud computing strategy for optimal utilization of resources.

Payment function:

1. Each service consumer (B) discloses its desired IT service portfolio to the service provider. The consumer can be single or multiple minded and can opt a fixed or negotiated pricing scheme.
2. Social choice: B and S jointly explore optimal service plan in terms of public, private or hybrid cloud computing and also the system architecture. They settle a combinatorial price through multi-party negotiation based on secure function decomposition with minimum disclosure. $f(x_1,x_2,\ldots,x_n) = f_n(\ldots(f_2(f_1(x_1),x_2)\ldots, x_n)$. S calls the price ladder; adjusts it adaptively by applying analytics and negotiates with B intelligently. B selects prepaid or postpaid option.
3. S *commits* price to each service of portfolio as desired by B; sends commitments to B and a trusted third party (T); also sends a de-commitment of the commitment to T.
4. S verifies the fulfillment of financial constraints for each service, computes zero *knowledge proofs* and sends the proofs to B. B verifies the completeness and correctness of proofs.
5. T reveals the commitments if there arises any dispute. The agents may have to pay penalty if T detects any malafide behavior.

Revelation principle*:* S does not disclose its capacity constraint, revenue function and target profit margin to B. B does not disclose its budget constraint to S. S also keeps the privacy of the intelligence of combinatorial pricing strategy. The agents behave rationally and fairly and share correct data (e.g. demand plan, QoS) during multi-party negotiation.

Output: Profitable discriminatory pricing plan (P_f) for each service consumer.

The basic objective of discriminatory pricing mechanism is to price different services differently for SaaS, PaaS, DaaS and IaaS and price the traffic transmitted for CaaS and dSaaS. This strategic move can optimize the revenue and the profit of the service provider and can increase the social welfare from the perspective of economic impacts. Technologically, it is not fair and rational to adopt a non-discriminatory pricing strategy for different types of cloud computing services. It is interesting to understand how discriminatory pricing impacts the service providers, service consumers and the society in cloud computing business. The next section analyzes the mechanism and explains the intelligence through various theorems.

Social choice is generally expected to ensure rationality, correctness and fairness in pricing of information and communication technology enabled services. But, the adoption of social choice is a debatable issue in today's business scenario. What should be the right model of pricing: is it a cooperative or non-cooperative game? For the last few decades, the service providers have been adopting social choice strategy in the form of free service or low price or random heuristics to test the price sensitivity of the service consumers. It is now a challenging task for the service providers to demand right price at higher rate since the consumers are habituated to enjoy the same service at free of cost or nominal rate. The social choice strategy has transformed the consumers extremely price sensitive. The providers may not be interested in innovation or improvement of QoS since they do not get fair price. They are also facing threats from the grey market. If the providers do not get right price for right service, the quality of service may be compromised and that will decline the demand of the consumers in long run. Both simple and complex types of discriminatory pricing mechanism are essential for the financially weak small and medium sized firms / agents and rich large complex enterprises according to the demand of complexity and quality of services.

4. MECHANISM ANALYSIS

This section analyzes the proposed pricing mechanism through various theorems.

Theorem 1: *The revelation principle of the pricing mechanism is based on secure function decomposition, cryptographic commitments and zero knowledge proof. The privacy of strategic data adds business intelligence to the pricing mechanism.*

The computational intelligence of the pricing mechanism is associated with a secure function evaluation which finds a good decomposition of a given function with minimum disclosure and devises a set of algorithms to compute the decomposition privately. In the proposed pricing mechanism, decomposition of a n-input function $f(x_1,x_2,\ldots,x_n)$ is a vector of functions $\{f_i(y_{i-1},x_i)$: i = 1,….,n$\}$ where for inputs $(x_1,x_2,..,x_n)$ $f(x_1,x_2,\ldots,x_n) = f_n(\ldots(f_2(f_1(x_1),x_2)\ldots,x_n)$, y_i is the intermediate result based on the inputs of the agents. The objective of the pricing mechanism is to find a good decomposition of a given function and devise a protocol to securely compute a given decomposition with minimum disclosure. It is based on secure multi-party computation (SMC) of approximation which may provide efficient solution while no efficient exact computation is known. SMC of approximation is useful in a distributed setting where all the inputs belong to different parties and the parties want to compute a function of their inputs securely without revealing more information than necessary.

The trading agents try to preserve the privacy of strategic data related to the competitive intelligence of the pricing mechanism. During multi-party negotiation, the trading agents do not disclose their reservation and expectation parameters to the opponents completely. For example, the service provider does not disclose its reference price ladder, revenue optimization model, target profit margin and capacity constraint to the service consumers. The provider also preserves the privacy of the business intelligence about the consumers mined by the analytic function and weights of the adaptive filter.

A cryptographic commitment is a piece of data which binds its creator to a unique value, yet appears random until it is decommitted. For example, a Pedersen commitment to w with randomness r is the group element $C_r(w) = g^w h^r$ and can be decommitted by revealing r and w [17]. Here, p is a large prime and q is a prime such that q divides [p-1]; $G = Z_p$ denotes the group of mod p integers; $g \in G$ and $h \in G$ be group elements of order q such that discrete log and $\log_g(h)$ are unknown. This commitment is computationally binding and unconditionally hiding. A zero knowledge proof of knowledge allows a prover to demonstrate knowledge of hidden values without actually revealing them. A proof of knowledge of a Pedersen committed integer w demonstrates knowledge of w and r such that $C_r(w) = g^w h^r$. One can also prove that a committed value w satisfies some condition $\phi(w)$ without revealing it. $POK(w,r \mid C = g^w h^r, \phi(w))$ denotes a zero knowledge proof of knowledge of (w,r) satisfying both $C = g^w h^r$ and the predicate $\phi(w)$. In the pricing mechanism, the service provider knows the actual allocation of price to each type of cloud computing service of IT portfolio. Each service consumer does not know the actual allocation but knows the commitment of the service provider to each sector of IT portfolio. The service provider computes zero knowledge proof $POK\ (w_i,r_i \mid C_i = g^{wi}h^{ri},\ w_i \in [(k_i + \alpha), (k_i - \alpha)])$ such that a committed integer satisfies an in equality or lies between an interval. In zero knowledge authentications, the claimant preserves the privacy of its secret data. The claimant proves to the verifier that it knows a secret without revealing it. The interactions between the claimant and the verifier can not disclose the secret. After exchanging a set of messages, the verifier only knows that the claimant does or does not have the secret.

For example, the trading agents can follow *Fiat-Shamir protocol* on zero knowledge proof [10]. A trusted third party chooses two large prime numbers p and q to calculate the value of n = p.q; n is a public data but p and q are kept secret. Alice, the claimant chooses a secret number s between 1 and (n-1). She calculates $v = s^2 \bmod n$; she keeps s as private key and registers v as her public key with the third party. Alice, the claimant chooses a random number r between 0 and (n-1); r is commitment. She then calculates $x = r^2 \bmod n$; x is the witness. s is Alice's private key. Alice sends x to Bob as the witness. Bob, the verifier, sends the challenge c to Alice. The value of c is either 0 or 1. Alice calculates the response $y = rs^c$. Alice sends y to Bob to show that she knows the value of her private key s. Bob calculates y^2 and xv^c. If these two values are congruent, then Alice either knows the value of s (she is honest) or she is dishonest. $y^2 = (rs^c)^2 = r^2 s^{2c} = r^2(s^2)^c = xv^c$.

Theorem 2: *The communication complexity of the pricing mechanism mainly depends on the time constraint of the negotiating agents, their information state and the number of negotiation issues. The computational complexity depends on the complexity of function decomposition.*

The communication complexity of the proposed pricing mechanism depends on number of negotiation rounds and the complexity of IT portfolio. The trading agents can bundle all the issues and negotiate them simultaneously as a complete package. Alternatively, they can negotiate the issues sequentially following some ordering constraints in preference. The information state of each trading agent is a private knowledge. The information that an agent has about its opponent is acquired through learning from previous encounters and this information is not known to its opponent.

The computational complexity of the pricing mechanism mainly depends on secure function evaluation and the complexity of function decomposition. The agents settle a combinatorial price through secure function evaluation. It is a decomposition of a n-input function $f(x_1,x_2,\ldots,x_n)$: a vector of functions $\{f_i(y_{i-1},x_i): i =1,\ldots,n\}$; inputs are $(x_1,x_2,..,x_n)$ and $f(x_1,x_2,\ldots,x_n) = f_n(\ldots(f_2(f_1(x_1),x_2)\ldots, x_n)$ where y_i represents the intermediate results based on the inputs of the agents. It is basically secure function evaluation wherein the agents preserve the privacy of critical data. The computational complexity is O(n). The cost of computation depends on the complexity of price ladder, analytics and adaptive filtering function. The computational burden of the price ladder is a function of the complexity of the optimization problem in terms of number of objective functions and constraints. The cost of computation of the adaptive filter is a function of number of triggers. The computational burden of the analytics is determined by learning algorithm, learning rate, number of input and output variables.

Theorem 3: *The discriminatory pricing mechanism is a profitable strategic move that improves the revenue and profit of the cloud service provider.*

Nondiscriminatory pricing strategy is not technologically feasible for different types of cloud computing services. Each service consumer tries to maximize its profit subject to individual rationality and incentive compatibility constraints. It tries to purchase a plan that offers nonnegative utility and should generate maximum surplus. On the other side, the service provider tries to maximize its profit subject to resource capacity constraints by adopting discriminatory and quasi bundling pricing strategy for different types of cloud computing services. The provider adopts various types of strategic moves to optimize its revenue and profit.

dSaaS / DaaS pricing strategy**:** The basic objective of DaaS is to avoid the complexity and cost of running a database with improved availability, performance, price and flexibility. It gives the access to various types business intelligence solutions (through web) which include distributed database, data warehousing, data mining, business and web analytics, data visualization and business performance measurement applications. The pricing of dSaaS is based on the cost of hardware (e.g. data warehouse, servers), the cost of software (e.g. business intelligence solutions) and system administration cost (e.g. data centre administration, data base security, backup, recovery and maintenance). A consumer can lease a data storage space where it is required to measure different system parameters such as stored data (GB/month) and number of processed queries (per 10k requests / month) to compute the price of dSaaS / DaaS. The provider can offer quantity discount in case of group buying of storage space. The prices of DaaS / dSaaS are also determined by various QoS parameters such as connection speed, data store delete time, data store read time, deployment latency (i.e. the amount of latency between when an application is posted and ready to use) and lag time (how slow the system is).

The pricing of dSaaS is also governed by the security and privacy of data and the related system architecture. A complex system architecture enhances the cost of computation and communication among the agents and also the cost of energy. There may be different types of system architecture possible in cloud computing. In a simple setting, the service consumer encrypts its data and stores in the data storage server of the service provider. Whenever required, the consumer gets access its data through proper access control schema and decrypts the data. In a slightly complex setting; the service consumer stores its encrypted data in the server of the service provider and wants to share data with a client. In this case, the service provider uses a data processor, data verifier and tocken generator for dSaaS service. The service consumer perform data indexing with the support of the data processor, encrypts the data and sends to the cloud server. The client requests the service consumer for a specific query on stored data. The consumer sends a credential and tocken to the client. The client sends the tocken to the service provider. The provider finds the appropriate encrypted data with the help of the tocken and returns the same to the client. The client and the consumer jointly check the integrity of data using verification mechanism. So, the costs of computation and communication are different in simple and complex cloud computing settings. The pricing of dSaaS should consider these issues intelligently.

Some applications (e.g. education sector) require low level of privacy of data. Some applications (e.g. financial service, healthcare etc.) need high level of security and privacy in data outsourcing and this involves high cost of computation and communication from the perspectives of statistical disclosure control, private data analysis, privacy preserving data mining, intelligent access control and query processing on encrypted data. The service provider should define a discriminatory pricing mechanism for dSaaS: high level of security and privacy of data demands high price and low level of security asks low price.

The price of dSaaS is a function of miscellaneous cost elements of a data center. A *data centre* or data bank is the collection of servers where the applications and data are stored. Data center consists of a set of servers and network architecture. The servers store the data from different organizations and network architecture facilitates the services to use, store, and update the data of the servers. The cost of administration of data centre includes several factors: initial development cost, operating cost, maintenance cost and cost associated with disaster recovery plan. The development cost includes the cost that requires making master plan, building infrastructure, buying hardware and software, making database and security schema. Operating cost includes the cost of energy, cooling system, system administrators, software license and network cost. Maintenance cost is the cost of maintaining the system which includes upgradation of hardware and software. One of the most challenging issues of data center management is the resource allocation strategy: how it is possible to cater the demand of the service consumers using minimum number of servers. It has an impact on the size, complexity and cost of data center. The data centre administrator can follow dedicated or shared server allocation strategy.

The price of dSaaS is also a function of *energy consumption* of cloud computing system in a data center. There are many open challenges of energy efficient design of computing systems and green IT covering the hardware, operating system, virtualization and data center levels [22]. The basic objective of the cloud computing system design has been shifted to power and energy efficiency to improve the profit of the service provider. Energy consumption is not only determined by hardware efficiency, but it is also dependent on the resource management system deployed on the infrastructure and the efficiency of applications running in the system. Solar power electronics is an interesting option of green IT. Higher power consumption results not only high energy cost but also increases the cost of cooling system and power delivery infrastructure including UPS and power distribution units / panels. The consolidation of IT infrastructure should be done intelligently to reduce both energy consumption and performance degradation through improved power management. Energy consumption can be reduced by increasing the resource utilization and use of energy efficient cloud computing system.

Software-as-a-Service (SaaS) pricing strategy: SaaS is an application hosted on a remote server and accessed through web; it can be business service or customer oriented service. The basic objective is to reduce software licensing cost and improve productivity by using sophisticated applications. The pricing strategy of SaaS is based on pay-as-you-go basis; not dependent on number of licensing period and licensing users as in case of direct software procurement. The service provider can configure the number of required features of a software as per the demand of a service consumer and price SaaS service charge accordingly based on basic, medium and mega package configuration. Another concept is *software plus service* where an enterprise uses a locally hosted software application and additionally uses SaaS through cloud for a specific type of application. Using the existing software paradigm, the consumer purchases a software package and license by paying a one-time fee. The software then becomes the property of the consumer. Support and updates are provided by the vendor under the terms of the license agreement. This can be costly if the user is installing a new application on hundreds or thousands of computers. SaaS, on the other hand, has no licensing. Rather than buying the application, the consumer pay for it through the use of a subscription based on number of concurrent users and only pay for what is used.

The computation of subscription fee can be *stochastic* pricing or simple *cost* based pricing. The price of SaaS depends on the specific business model of the service provider. Suppose, a service provider develops in-house software products. Another service provider buys COTS from third-party vendor based on number of licensed users and licensing period and provides SaaS to the consumers. There may be restriction of number of concurrent users and different subscription rate of SaaS in second case.

This pricing strategy should also consider cost of upgrading software application; the provider may offer incentive for upgrading applications. In case of security software pricing, there may be different alternative strategies to manage network security: (i) consumer self-patching where no external incentives are provided for patching or purchasing, (ii) mandatory patching, (ii) patching rebate and (iv) usage tax. For proprietary software, when the software security risk and the patching costs are high, a patching rebate dominates the other strategies. When the patching cost or the security risk is low, self-patching is the best option.

Stochastic risk based pricing mechanism considers several risk factors and optimizes the expected net present value of revenue subject to maximum acceptable risk of the provider. In this case, the service provider does not give much focus on cost accounting model or profit margin but tests the price sensitivity of the customers experimentally or through trial and error method. The provider does not have any precise perception about the demand of the new software products. But, it follows dynamic risk based pricing based on assessed risks and competitive intelligence. For in-house software development, software cost is a function of efforts on feasibility study, requirement analysis, system design, program design, coding, testing and modification following waterfall / v-process / spiral / proto-typing / incremental delivery model. The service provider estimates effort for a specific SDLC model and then selects an optimal profit margin.

Infrastructure-as-a-Service (IaaS) pricing strategy: A cloud computing infrastructure consists of different types of elements: clients (e.g. mobile, PDA, laptop, thin and thick), the data center and distributed servers. *Thin clients* are less costly than thick clients. A growing trend in the cloud computing is *virtualization* of servers. In a virtualized environment, applications run on a server and are displayed on the client. The server can be local or on the other side of the cloud. Software can be installed allowing multiple instances of virtual servers which run on a physical server. Full *virtualization* is a technique in which a complete installation of one machine is run on another. It allows the running of different and unique operating systems. *Hardware-as-a-Service (HaaS)* simply offers the hardware required by a consumer. Cloud computing is a business model of delivering IT resources and applications as services accessible remotely over the Internet rather than locally. IaaS supports remote access of computer infrastructure as a service.

Cloud computing supports elastically scaling computation to match time varying demand. But, the uncertainty of variable loads necessitate the use of margins i.e. the servers that must be kept active to absorb unpredictable potential load surges which can be a significant fraction of overall cost. [23] addresses the challenges of minimizing margin costs and true costs for IaaS. The provider should not adopt a fixed margin strategy; the margin should be load dependent. The margin required at low loads may be higher than the margin required at high loads. Secondly, the tolerance i.e. the fraction of time when the response time target may be violated need not be uniform across all load levels. It is really challenging to achieve optimal margin cost while guarantying desired response time for IaaS.

The pricing strategy of IaaS is based on the cost of servers, storage space, network equipment and system software like operating systems and database systems. The price of IaaS is basically a subscription fee for a specific timeline. Now the question is how to compute this subscription fee. The rate should be fixed based on the cost of hardware and software, target revenue and profit margin. The service provider may adopt a profit maximizing pricing strategy or revenue maximizing pricing strategy within reasonable, stable target profit margin. The profit margin is a dynamic variable; it should be set intelligently according to competitive intelligence and quality of service. The quality of service is measured in terms of computing time. For small firm or individual service consumer, the provider can set a fixed price per unit time; there may be SLA but there is no scope of negotiation of price. Large PSU can negotiate with the service provider to set a rational price for fixed timeline.

Incentive compatibility plays a significant role in IaaS pricing, it is important to analyze the significance of incentives for network infrastructure investment under different pricing strategies: *congestion based negative externality pricing* and the flat *rate pricing* [33]. A lack of proper infrastructure investment incentive may lead to an environment where network growth may not keep pace with the service requirements. It is really complex to compute maximum capacity that IaaS provider will be willing to invest under different pricing schemes. Optimal capacity of IaaS is determined by different factors: per unit cost of capacity of network resources, average value of the user's requests, average value of the user's tolerance for delay and the level of exogeneous demand for the services on the network. It is hard to determine whether time based pricing is more profitable than flat rate pricing. IaaS consumers always try to identify whether average stream of the net benefits realized under congestion based pricing is higher than the average net benefits under flat rate pricing. IaaS provider may adopt different types of pricing strategies at different points of time but the service consumers may control their demand of IaaS service adaptively to avoid the increase in cost.

***Platform-as-a-Service (PaaS) pricing strategy*:** PaaS supplies all the resources required to build applications and services completely from the web without any download or installation of any software in the clients. The price of PaaS can be negotiated for a specific project. There can be different types of project environments such as application-delivery-only-environment (e.g. security and on demand scalability), standalone environment and add-on-developmental-environment (e.g. subscriptions of add-on SaaS application are bought). The price of system software can be charged as a subscription fee based on number of concurrent users and usage period. The pricing of PaaS is also governed by the complexity of platform services which may include application design, development, testing, deployment, hosting, geographically dispersed team collaboration, web service integration, database integration, security, scalability, storage, state management and versioning. The developers, project managers, and testers can access the development and testing softwares of the service provider through web; but, lack of interoperability and portability may be a critical issue in PaaS. The price of PaaS is determined by the complexity of interoperability between the systems of the service provider and service consumer.

Communication-as-a-service (CaaS) pricing strategy **:** For CaaS, a service provider can adopt service or traffic pricing strategy and can offer several options to the consumers for voice calls, text messages, multi-media messages, mobile internet and smart phone applications. For CaaS, the service provider may adopt fixed up to pricing : a fixed fee p, a free call time allowance of n units and over limit rate r i.e. charge p for usage upto n units and bill r for usage over n units [36]. In case of traffic pricing, the service provider simply prices the traffic consumed and each consumer has the right to allocate the purchased traffic among various types of services according to individual preferences. The provider designs only a single price schedule to price the traffic consumption; each plan in the schedule provides a certain level of traffic usage for a specific price setting. Traffic pricing is a specific case of quasi bundling. Bundling of services may increase the profit of the service provider when the service consumers have different valuations for individual goods but similar valuations for a bundle of services. Though the service provider chooses the bundle composition, each consumer chooses individual traffic allocation in case of traffic pricing.

Rational and intelligent CaaS service package configuration is important for customer retention. Bad configuration and poor quality of content result the disappointment of the service consumers and subsequent loss of revenue of the service provider. For example, communication service providers may try to optimize their revenue and profit through various ways: rapid increase in service charge, no price protection and dynamic service package configuration. The service consumers may not be happy with this business strategy of the service providers. Moreover, they may not be satisfied with the quality of content of the communicated data (e.g. TV broadcast due to got-up games, match fixing, misleading bogus news, cock and bull stories and stupid TV serials). Some consumers may be switched off from one to other service provider; the price sensitive consumers may terminate CaaS service contract as they find no good value in the service,

The proposed pricing mechanism is able to improve revenue within target profit margin as compared to the works of [3,4,5]. Cloud computing itself is an innovative business model. This business model can be more attractive by adopting various intelligent strategies. The common issues of pricing for utility computing are discussed in [3,4] without indicating any business intelligence move which can improve the revenue and profit of the service provider. [3] uses the concept of utility theory for service level agreements between the trading agents. Service-level agreement (SLA) is a bilateral contract that governs the terms of the interaction between the trading parties such as their identities, valid timeline, quality of service, service level indicators, service level objectives, pricing issues, payment terms and penalty clauses. The price function is a multi-dimensional outcome. The trading agents try to optimize the utility function based on their preference. This work did not address the risks of strategic pricing for revenue optimization.

[5] considers a natural pricing problem wherein a seller has n items to sell to a buyer who is interested to buy a single item. The seller wants to maximize profit by using stochastic knowledge of the buyer. The seller has access to a distribution from which the values $(v_1,\ldots,v_n)$ of the buyer for n items are drawn. Given this information setting, the seller needs to compute prices $(p_1,\ldots,p_n)$ for the items to maximize her revenue. The buyer is quasi-linear i.e. buys the item i maximizing $(v_i - p_i)$. The seller's expected pay-off from a price vector $P = (p_1,..,p_n)$ is $R = \sum_{i=1}^{n} p_i.P_r\,[\, i = \text{argmax}\ \{v_j\text{-}p_j\} \wedge (v_i\text{-}p_i) \geq 0]$. In case of single dimensional setting, the buyer values all the items equally. In case of multi-dimensional setting, a buyer values differently to different items. This work recommends a polynomial time approximation scheme for multi-dimensional unit demand pricing problem. The algorithm is based on extreme value theorem. The computation of pricing algorithm has not considered the issue of combinatorial pricing though a buyer may have demand of multiple items. The buyer is quasi-linear i.e. buys the item i maximizing $(v_i - p_i)$; it may not be able to boost the demand and revenue of the seller for different items. The computation of expected pay-off has not considered the cost factors of the seller. The scope of secure function evaluation is not explored for intelligent negotiation. It is difficult for a seller to access the distribution of valuation of a buyer precisely during negotiation since this is private information. It is stochastic information; the trading agents do not want to disclose their expectation and reservation fully to the opponent parties.

The proposed discriminatory pricing mechanism is able to improve the performance of pricing algorithm in terms of profit, stability and robustness. The mechanism computes combinatorial price of a set of IT services related to cloud computing; this move boosts the demand and revenue of different types of interdependent IT services intelligently. It also results a simple computation. A service consumer may be interested in buying only a single IT service; this case discloses the price of a service. But, the price of a single service should not match with the combinatorial price. The computation of combinatorial price preserves the competitive intelligence of the service provider for different types of cloud computing services. The competitors of the service provider may capture an approximate idea of the price of the service provider for different IT services. This is a nonlinear constrained optimization problem where the search is governed by a set of business intelligence moves. Secure function decomposition adds suitable business intelligence to the pricing mechanism. The pricing mechanism offers adequate business intelligence through an optimal price ladder, adaptive filter, business analytics and negotiation in an integrated way by solving a non-linear constrained optimization problem. This is not only a recommender system that predicts critical pricing parameters efficiently; it is also a risk mitigation tool. This is a dynamic pricing mechanism which is suitable for an adaptive business model. This business model provides information services when required. It reduces the financial risk of the service consumers through short lead time in service provisioning, high reliability, customized service level agreements and reduced learning time in the adoption of new technology. The service consumers do not incur high fixed cost of procurement or do not commit to long term fixed price outsourcing contracts. They receive the service they need and pay as per usage. On the other side, the service provider can attain economies of scale and high capacity utilization. Therefore, cloud computing can be a dominant strategy for both the service consumer and service provider.

Is competitive intelligence an important factor in cloud computing pricing mechanism? The service provider must analyze how its offering is different from the offerings of its competitors; do the consumers give importance to the brand and reputation of the provider or do they value high QoS? Does a high price signal high quality of service? Can a consumer compare the price and performance of alternatives? Is a consumer free to switch from one provider to the other without incurring high cost? It is critical for the service provider to assess what value its service consumers perceive on its cloud computing service. The provider must understand how differently the consumers value its services. Do the service consumers vary in their intensity of use? Do they use the cloud services differently? Heavy users generally value a cloud service more than light users; some consumers may use a service differently from others with a different perceived value. The service provider should be able to analyze the perceived value of each consumer; should try to differentiate the profitable consumers from the average prospects, segment the consumers and customize its price settings to generate high profit and revenue. The provider should also analyze the reactions of its competitors in a price war. A competitor can turn a brilliant pricing move of the provider into an ordinary one if the pricing decisions are made without considering second and third order effects. The provider should analyze the overall impact of the new price on the profitability of the industry. It should estimate whether the returns are worth the cost to serve. It should avoid the strategic accounts zone where the price received is low and cost to serve is high.

The service provider analyzes the switching cost of the service consumers from one provider to the other. The current consumers may incur high costs if they switch to another provider. The switching costs include physical configuration and installation costs, contractual costs and cognitive costs of learning. The switching costs may enable a service provider to extract more revenue from the existing consumers by charging them higher prices. But, higher prices may deter new consumers from buying cloud computing services and they may become more price sensitive and value to the competitive intelligence in pricing. It raises an important question of analytics: how best to balance the tradeoff between short-term revenue gain and long-run account growth [29]. Lower prices attract new consumers because new customers are price sensitive. On one side, lower and competitive prices ensure that the customer base of the service provider grows over time. But, lowering prices generates an immediate revenue loss from the base of existing locked-in consumers who can tolerate higher prices due to the switching costs. The service provider estimates switching costs and computes upper and lower bound of price intelligently by making a trade-off between revenue from existing consumers at higher prices versus growing market share with lower prices in account based services. The provider should price different cloud computing services according to target market share: [$Move_1$] : Price at the upper bound when the market share is above the target; [$Move_2$]: Price at the lower bound if the market share is below the target; and [$Move_3$]: Price at equilibrium target price when the market share is at the target.

The proposed pricing mechanism works effectively if it is possible to explore all possible *leakage paths* of the current revenue and profit management streams of the service provider. Revenue and profit can be leaked in various ways like loss of potential quotes, incorrect assessment of the credit risk and price sensitivity of the consumers, loss of cross-selling opportunities and incorrect computation of optimal price ladder and adjustment factors of the adaptive filter. Revenue and profit may be reduced due to poor quality of service of the service provider.

Theorem 4: *The pricing mechanism is associated with a sequential move game where the service consumer can select public or private or hybrid cloud computing strategy.*

In the game tree, each service consumer can opt one of two alternative options for IT portfolio management: either adopt cloud computing strategy or go for direct IT investment. Cloud computing is a unique, cost effective, differentiated business model. It makes the service consumer agile and flexible to the needs of an enterprise. Pay-for-what-happens is a flexible IT pricing strategy; the consumer can pay per user per month on any application as a service. It should not throw anything away; rather it should build on existing IT assets and choose a hybrid model of on-premises and off-premises resources. At the next level of the game tree, there are different options of cloud computing such as public, private and hybrid cloud. The computing resources are dynamically provisioned over web via web applications or web services from an off-site third party service provider in public cloud computing. In case of private cloud computing, private networks are used to provide full control over data, security and quality of service by a cloud provider or a company's own IT division. A hybrid cloud environment combines private and public cloud models.

The service consumer can reduce IT maintenance cost significantly and can always use the latest software applications with the cloud without worrying about upgrades and patches. It can reduce the training cost using the skill, product knowledge and experience of the cloud service providers. It can build a real-time enterprise model with the help of an enterprise ready cloud computing infrastructure through a well-defined SLA and 24/7 support. Each service consumer uses resources more effectively; a service consumer can identify areas that can be moved to the cloud and quickly free up skilled IT staff on high value initiatives. It can lower operating costs in terms of IT infrastructure, maintenance and operational costs. It can stay secure and store sensitive data on a global network of sophisticated data centers. Further, it can develop IT solutions faster with the support of the service provider's platform and interoperability support of third party solutions.

Cloud computing is particularly a desirable option for small to medium sized business wherein in-house development and operations of IT applications may be time consuming and expensive. Small or medium sized enterprises are best served by cloud computing within small IT budget. Large enterprises may select cloud computing as a suitable option when they want to experiment with new information technologies without high initial investment. It helps improve productivity by providing new machines and instant access to new resources and software; it builds an adaptive enterprise model in a changing business environment. The overall business competitiveness can be strengthened by reducing the time of deployment and enhancing the ability to adapt to changing market conditions. The new paradigm of cloud computing provides different types of benefits but there are still a number of challenges such as performance for intensive transaction and data oriented applications, security and privacy, control over IT platform, bandwidth costs and reliability of service.

Equilibrium analysis: It is basically a sequential-move game where the decision making agents consider the future consequences of their current moves before choosing the actions. This sequential game requires a strict order of play. Each player analyzes the current move of its opponent and takes decisions regarding its next move accordingly. It must consider that if it makes a specific move, how its opponent will respond. The agents thus decide their current moves based on the estimation of future consequences.

The outcome of this game is driven by the information states and time constraints of the decision making agents and the number of negotiation issues. In case of multi-issue negotiation, the agents may negotiate multiple issues sequentially or simultaneously. The sequential approach is less complex than the simultaneous approach from computational perspectives. Each agent defines its information state in terms of negotiation strategies, business objectives, aspiration and reservation levels. It may adopt different types of time-dependent strategic moves such as linear, boulware or conceder. An agent may go to its reservation level from initial level linearly. It may maintain its initial offer till the timeline is almost exhausted and then offers its reservation value. Alternatively, it may offer its reservation value very quickly and maintains the same offer till the deadline. Each agent's information state is a private knowledge; it has both complete and incomplete information. An agent does not disclose its negotiation strategy to the other players; it has only probabilistic information about the strategic moves of the other players. But, the agents have complete information about the rule of the game, negotiation protocol, negotiation issues and time constraints.

The trading agents alternately propose offers and counteroffers. The negotiation starts when an agent makes its offer to the other agent; the other agent receives the offer; evaluates the offer using its utility function; either accepts the offer or makes counter-offer; the agents try to reach an agreement. The pricing mechanism tries to explore a set of stable solutions so that the strategic moves of the agents constitute equilibrium. A set of strategic moves are in Nash equilibrium if each agent's strategic move is a best response to its opponent's move. An outcome is pareto-efficient if there is no other outcome that improves the payment of an agent without making another agent worse off.

5. INFORMATION SYSTEM SCHEMA

Let us discuss the information system schema which should be associated with the proposed pricing mechanism and provides necessary business intelligence. An information system schema has different system components such as computing, data, communication networking, application and security schema. The application schema is basically an integrated system with a mix of a price ladder, adaptive filter, analytics and negotiation support system. The trading agents call a set of intelligent applications to compute the combinatorial price. First, the service provider computes reference optimal price ladder (P_o) using a pricing algorithm. Each service consumer informs its demand plan, budget constraints and rank of preference to the provider. Secondly, the provider calls analytic function and estimates credit risk, price sensitivity and cross selling opportunities of each consumer. Thirdly, the provider calls an adaptive filter; adjusts P_o dynamically as per external triggers and estimated risk and opportunities and generates adjusted price ladder (P_a). Finally, the provider and consumer settle a combinatorial price for a set of cloud computing services using a negotiation support system.

The data schema requires a simple data structure for a price ladder: a set of cloud computing services, system performance measurement parameters, time duration, demand band, unit price, discount, penalty and payment terms. Additionally, the data schema requires the support of a set of data mining algorithms for intelligent business analysis. The basic building block of the communication schema is an web enabled distributed computing system. The security schema is expected to ensure authentication, authorization, correct identification, privacy and audit in each business transaction. The following sub-section is mainly focused on the computational intelligence of the pricing information system.

5.1. Computational Intelligence

A smart market requires *computational intelligence*. From the perspective of the design of a smart market, platform and the associated decision support tools, there are two broad categories of intelligence [31]: (a) real-time intelligence that is used primarily by individual or atomic entities in dynamic markets such as combinatorial auction and (b) collective intelligence where several atomic entities across different tiers of a supply chain interact with each other. The *collective intelligence* refers to developing an understanding of complex relationships in any multi-echelon ecosystem of codependent and coevolving markets such as end-to-end supply chains in a competitive market. The enhanced computational powers, open programming paradigms and advances in information system research have opened up unprecedented opportunities to study the whole ecosystems. The study of these ecosystems provides opportunities to understand the role of information and communication technology in the evolution of cloud computing business.

Figure 1 in the appendix shows the computational intelligence of the pricing mechanism. It is basically a constrained nonlinear optimization problem. The search is governed by a set of intelligent moves. In the pricing mechanism, the service provider and service consumer jointly compute a combinatorial price through secure function evaluation. It is basically an approach of function decomposition $f(x_1,x_2,x_3,x_4) = f_4(f_3(f_2(f_1(x_1),x_2)x_3),x_4)$ where f_1 : price ladder function, f_2 : analytic function, f_3 : adaptive filtering function and f_4 is negotiation function. The pricing function is associated with a dynamic system where different moves occur at different points of time.

The concept of secure function decomposition is approximately similar to *dynamic programming* (DP) which determines the optimum solution of a multivariable problem by decomposing it into stages, each stage comprising a single variable sub-problem. The advantage of the decomposition is that the optimization process at each stage involves one variable only, a simple task computationally than dealing with all the variables simultaneously. A DP model is basically a recursive equation linking the different stages of the problem in a manner that guarantees that each stage's optimal feasible solution is also optimal and feasible for the entire problem. The basic elements of DP are as follows: (a) define a set of *stages*; (b) define a set of *alternatives* at each stage and (c) define the *states* for each stage. The definition of the states varies depending on the situation being modeled. It is also important to know what relationship binds the stages together and what information is needed to make feasible decisions at the current stage without reexamining the decisions made at the previous stage. Unlike DP, secure function decomposition defines a set of stages where each stage may involve more than one variable. Secondly, it is difficult to define recursive equation in secure function decomposition.

Many useful algorithms like DP are recursive in nature; to solve a given problem, they call themselves recursively one or more time to deal with closely related sub-problems. These algorithms typically follow a *divide-and- conquer* approach: they break the problem into several sub-problems that are similar to the original problem but smaller in size, solve the problems recursively and then combine these solutions to create a solution to the original problem. The divide and conquer paradigm involves three steps at each level of the recursion: (Step 1) *Divide* the problem into a number of sub-problems. (Step 2) *Conquer* the sub-problems by solving them recursively. If the sub-problems are small enough, just solve the problems in a straightforward manner. (Step 3) *Combine* the solutions to the sub-problem into the solution to the original problem. Secure function decomposition divides the pricing problem into a set of sub-problems based on different intelligent moves; conquers each sub-problem in a straight forward way (may not recursively) and finally combines or integrates the solutions to the sub-problems into the solution of the original problem. Each sub-problem is not similar type of problem such as optimization.

5.1.1 Price ladder

Price discrimination is an interesting move of the service provider to improve its revenue and profit. There are different tactics of price discrimination: charging different prices to different service consumers for exactly the same product or service, charging different prices for different versions of the same products, charging different prices for different zones or different channels. In this connection, the critical task is to segment the market into different clusters such that high price can be charged to the premium consumers and low price can be offered to the price sensitive consumers. The core problem of strategic pricing can be formulated as a constrained nonlinear optimization problem where the price ladder is a function of target revenue and profit margin. The constraints originate from various businesses rules, regulatory constraints, lower and upper bounds of critical pricing parameters and resource capacity of the service provider. Let us consider an example of price ladder function.

input: A set of cloud computing services; a set of feasible combinations of services, forecasted demand plan;
algorithm [PL]:
1. Define a set of services $(s_1,..,s_z)$ and a set of feasible combinations of services $\{(s_1, s_2), (s_1, s_2, s_3), ..,(s_m, s_q)\}$.
2. Compute initial optimal price for each service: $\{(s_1 \rightarrow p_1),..,(s_z \rightarrow p_z)\}$ for $T_1 \le t \le T_2$;

Select move:

Move 1 [*Cost based pricing*]: $p_i = [1+m_i].\, c_i(d_i)/d_i$ for $i = 1,\ldots,z$; $c_i = c_h + c_s + c_n + c_y + c_m + c_a + c_r + c_a$;
Move 2 [*Profit maximizing pricing*]: Maximize $[p_i(d_i).d_i - c_i(d_i)]$ s.t. $0 < d_i \le D_i$; for $i = 1,\ldots,z$;
Move 3 [*Stochastic pricing*]: Maximize expected $NPV(p_i,t)$ s.t. probability $[(m_i(p_i, t) \ge m_i^{'})] \ge \Theta$ for $i = 1,\ldots,z$;
Move 4 [*Revenue - profit trade-off*]: maximize $\sum_{j=1}^{z} p_j L_j$;
s.t. $\sum_{j=1}^{z} c_j \le c^{'}$; $L_j \le z_j$; $L_j \ge 0$; $p_j \ge 0$; $m_{j,min} \le m \le m_{j,max}$; $q_j \le q_{th}$;

3. Do *what-if analysis* → Compute *discount function*, *penalty function* and *payment terms*; Compute *combinatorial price*.

output: *Reference price ladder* $[P_o]$.

In this example, the reference price ladder can be computed by calling different moves such as cost based pricing, profit maximizing, stochastic risk based pricing and revenue maximizing pricing within target profit margin. But, the decision of a specific move depends on the business objectives and strategies of the provider.

***Move 1*:** Any pricing mechanism is associated with three critical parameters: demand (d), cost structure (c) and gross profit margin (m). The cost structure includes sunk cost, fixed cost and variable cost. Sunk cost is independent of capacity; fixed cost is a function of capacity and the variable cost is proportional to the demand served by the provider. The simple *cost based pricing mechanism* assumes that the demand is controlled by price and the decision making agents act rationally in the trading process. The cost of a service (c_i) has various components : hardware cost (c_h), software cost (c_s), networking cost (c_n), security cost (c_y), maintenance cost (c_m), advertising and promotion cost (c_p), space rental cost (c_r) and system administration cost (c_a). The operational cost includes the cost of workload planning, service discovery and load balancing, resource allocation, workflow management, program management, database administration, system integration, operation, up-gradation and maintenance of IT system. Software cost is a function of efforts on requirement analysis, system design, coding and testing or direct procurement cost of COTS. It is not a risk based pricing but considers approximate demand forecast. Cost based pricing is particularly suitable for IaaS and dSaaS where it is relatively simple to define cost structure; it can be used during initial phase of cloud computing business cycle.

Move 2**:** *Profit maximizing pricing mechanism* tries to optimize the profit of the service provider subject to its capacity constraint. In this case, maximization of the profit is the primary objective and revenue maximization is not a critical issue. It can be used at matured phase of cloud computing service.

Move 3 **:** *Stochastic risk based pricing mechanism* considers several risk factors such as uncertainty in demand, estimated market size, expected capacity utilization factor, high sunk cost, short product life-cycle, rate of adoption and demand elasticity. This move optimizes the expected net present value of revenue subject to maximum acceptable risk of the provider. This move is suitable for SaaS and PaaS.

Move 4*:* The objective of move 1 is to trade-off revenue and profit margin of the provider. This move is applicable to all types of cloud computing services; the provider tries to maximize its revenue and QoS index of each service subject to cost and capacity constraint within target profit margin. It is basically a multi-objective combinatorial optimization problem which involves a set of objective functions such as maximization of the revenue of the provider and QoS index subject to a set of constraints; these constraints include the limit on total cost, load capacity for different types of services and profit margin of the provider. It is not simple to compute optimal price ladder due to various reasons [19]. Target profit margin should be determined based on economies of scale, transaction cost, learning curve and business dynamics. The target profit margin should be set such a way that the cost of cloud computing is moderately less than the cost of direct IT investment. Each pricing parameter may have two components: fixed and variable. There are other limitations such as cannibalization and arbitrage. The provider should be able to assess the price sensitivity of the service consumers precisely. Otherwise, it is not possible to compute a discriminatory price properly.

In move 1, the optimization problem may be multi-objective combinatorial problem and it can be solved by weighting or ϵ-constraint method [34]. In case of weighting method, each objective function is associated with a weighting co-efficient and the provider maximizes the weighted sum of the objectives. In this method, multiple objective functions are combined into a single objective function through weighting co-efficients. The basic problem can be expressed as : maximize $\sum_{i=1}^{k} w_i f_i(\mathbf{x})$ s.t. $\mathbf{x} \in S$, where $w_i \geq 0$ for all $i = 1,\ldots,k$ and $\sum_{i=1}^{k} w_i = 1$. Here, w_i are weighting co-efficients, $f_i(\mathbf{x})$ is the i^{th} objective functions, x is the decision vector belonging to a feasible region of decision variable space forming constraint functions. Alternatively, in case of ϵ-constraint method, one of the objective functions is selected to be optimized and all the other objective functions are converted into constraints by setting an upper bound to each of them. The problem can be expressed as maximize $f_l(\mathbf{x})$ s.t. $f_j(\mathbf{x}) \leq \epsilon_j$ for all $j = 1,\ldots,k$, $j \neq l$ and $\mathbf{x} \in S$.

This move uses following notations : p - price, d - demand, D - maximum resource capacity, c - cost, m - gross profit margin, m' - target profit margin, NPV - Net present value of expected revenue for a specific period t, Θ - maximum acceptable risk; j = 1,.., z: A set of cloud computing services; $p_j \rightarrow$ price of j^{th} service; $q_j \rightarrow$ QoS index of j^{th} service ; $L_j \rightarrow$ load of j^{th} service; c_j - cost associated with j^{th} service; $c_j = f(q_j)$; $z_j \rightarrow$ load capacity limit of j^{th} service; c' $\rightarrow$ cumulative cost i.e. budget constraint; m $\rightarrow$ Profit margin of service provider, k – discount factor, k_p – penalty factor, n – no. of concurrent users, $R_{j,QoS}$ – Rating of quality of j^{th} service, t – timeline.

Combinatorial pricing: First, it is required to compute the price for each type of IT service. Next, the provider defines a set of feasible combinations of IT services; estimates incentive or discount and penalty factors and computes combinatorial price ladder. It is really difficult to find out optimal discount factors; the service provider performs what-if-analysis on revenue and profitability; it includes goal seeking and sensitivity analysis. Goal seeking finds out the price for a target profit margin; sensitivity analysis is done to understand the impact of discount or incentive on profit and revenue for different services. For a feasible combination of service, the combinatorial price is as follows : $[\{(s_1,s_2) \rightarrow (k_1.p_1 + k_2.p_2)\},..,\{(s_1,s_2,s_3) \rightarrow (k'_1.p_1 + k'_2.p_2 + k'_3.p_3)\},...,\{(s_m, s_q) \rightarrow (k_m.p_m + k_n.p_q)\}]$; for *single service*: $\{(s_j \rightarrow p_j \text{ if } n_1 < n \leq n_2), (s_j \rightarrow k'_j.p_j \text{ if } n_2 < n \leq n_3), (s_j \rightarrow k''_j.p_j \text{ if } n > n_3)\}_{j=1,..,b}$; $\{(s_j \rightarrow p_j \text{ if } d_1 < d_j \leq d_2), (s_j \rightarrow k'_i.p_i \text{ if } d_j > d_3)\}_{j = g,h}$. The *penalty function*: $(s_j \rightarrow k_{pj} \text{ if } q_j < q_{th})_{j=1,..,c}$; *payment terms* for pre-paid and post-paid service : ($T_{pre\text{-}paid}$, $T_{post\text{-}paid}$, penalty clauses).

A sample price ladder is shown in Table 1. The price ladder defines the price settings of a set of IT services based on computational resources, time slots, demand band, discount, penalty and payment terms. The ladder is computed for different types of IT services such as SaaS, PaaS, IaaS, dSaaS, DaaS, HaaS and a set of feasible combinations of service packages. The pricing for cloud computing services is a multidimensional problem; it is defined based on several parameters such as stored data, storage transactions, incoming and outgoing bandwidth and computing time [11]. Storage is measured as average daily amount of data stored in GB over monthly period. Bandwidth is measured by computing total data transferred in and out of platform service through transaction and batch processing. Computing time is measured as the time units required to run an instance or application or machine to servicing requests.

5.1.2 Adaptive Filter

The dynamic pricing mechanism requires the support of an adaptive filter which can adjust optimal price ladder dynamically as per external triggers, assessed business risk and opportunities. In the proposed pricing mechanism, the service provider uses an adaptive filter which gets reference price ladder and various external triggers as inputs. The filter computes *adjustment factors* based on competitive intelligence, capacity utilization, adverse selection effect, regulatory constraints, change of market conditions and various macro-economic factors, corporate financial health and local and global business dynamics. It also computes *adjustment factors* based on revolving credit policy and refinancing strategy. The filter generates adjusted price ladder (P_a) based on reference price ladder and adjustment factors. Each service consumer pays fees to the service provider periodically; it includes payment for the application software, hardware, service and support, maintenance and upgrades. The service provider can vary the unit rate as per time of usage, low rate for high timeline and high rate for low timeline. It can be fixed or based on utilization. The filter measures and controls *adverse selection effect*. If the provider increases the price, the number of service consumers may be decreased and the consumers with high risk will start taking the resources. The filter should understand *local and global dynamics of competition*. It applies competitive intelligence to analyze the offers of the competitors of the provider. It differentiates the demand and risk characteristics of local and global customers effectively. It configures the price ladder differently for different zones. The filter analyzes the capacity *utilization* of the provider. It can be configured based on a revolving credit policy. A specific price point may be optimal to acquire a new consumer; but the same may not be optimal for the retention of the consumer on long term basis. It requires a customer life-cycle view rather than single point price setting.

Input: Reference price ladder (P_o), Triggers, (λ,r,o), filtering function (F);

Algorithm [AF]: compute weights (**w**); $y(t) = \sum_{j=0}^{n-1} w_j(t).u_j(t)$; $e(t) = v(t) - y(t)$; $y(t) / u(t) = G.(1+H) / (1+G)$; $F(P_o,\mathbf{w}) \rightarrow P_a$. *Output*: Adjusted price ladder (P_a).

In a simple adaptive filter, u(t): input applied to the adaptive filter, y(t) = output of the filter = $\sum_{j=0}^{n-1} w_j(t).u_j(t)$; v(t): desired response and e(t) = v(t) - y(t) = estimation error. If G and H are the constant gains of the adaptive filter and pricing system respectively; then the transfer function is y(t) / u(t) = G.(1+H) / (1+G). Here, the pricing system and the adaptive filter are driven by the same input. The output of the pricing system is the desired response of the filter. The error (e) is used to control the values of a set of adjustable coefficients or weights (w_j). A filter performs three basic information processing tasks: filtering, smoothing and prediction. The filters can be classified into linear and nonlinear. A filter is linear if the filtered, smoothed or predicted output parameters are a linear function of the inputs to the filter. Otherwise the filter is nonlinear. In this case, the service provider preserves the privacy of all input data (P_o,λ,r,o,F) and adjustment factors (**w**).

The adaptive filter ensures the stability and robustness of the pricing mechanism. It should provide good performance characteristics and should be robust enough against the disturbances, uncertainties of parameters and unmodelled structural properties of the system under control. The other critical task is to keep a predefined error between desired state and current state so that the system stays at an efficient point of the state space. If the service provider tries to maximize its profit unlimitedly, it may lose its stability in achieving target revenue. On the other side, if it tries to focus only on revenue optimization, it may lose its target profit margin. The adaptive filter tries to maintain the stability and the robustness of the performance of the proposed pricing system.

5.1.3 Negotiation Support System

Negotiation is a means for the trading agents to reach mutually beneficial agreements through communication and compromise. This is a value driven approach by which a joint decision is made by the agents. They exchange information in the form of offers, counter-offers and arguments and search for a fair and optimal value based price setting. The service provider tries to understand when it should make an offer to a consumer, when a consumer decides to accept or reject an offer and how the quoted pricing plan can be adjusted to reach an agreement. On the other side, each service consumer tries to explore optimal value in desired services. In case of SaaS, the consumer can negotiate the pricing plan with the provider on maximum number of concurrent users and usage period. For dSaaS, they can go for swing option contract; For PaaS and IaaS, they can negotiate the pricing plan based on miscellaneous issues such as interoperability, standard or nonstandard schema and simple or complex computation. An web enabled negotiation support system is a critical component of pricing system.

A smart market needs real-time intelligence due to significant uncertainty about demand and supply; the market mechanisms focus on *equilibrium*, *incentive compatibility* and also *computational aspects*. The trading agents can jointly settle different types of intelligent contracts such as option, auction, quantity discount and delivery flexibility contract. They need a negotiation support system for different purposes. Firstly, the bidding agents may face difficulties to *formulate* and *express bidding preferences* appropriately: what information should be revealed; what should be the format and specific rules of engagement in the negotiation mechanism. Secondly, NSS should provide necessary *intelligence for the bidders' decision making*: how to express multidimensional preferences as bids and how to reasonably and accurately predict future state of the negotiation to understand the quality of their own bids.

Thirdly, the bidders should be able to *express complex preference*. NSS may be built on web service oriented computing to express and exchange information about their preferences in multiple dimensions (e.g. multiple items, multiple units and multiple issues) and in real time. Developing more expressive and yet simple and compact bidding languages can be implemented by using functions or logic formalisms. It supports an intelligent negotiation protocol by creating higher allocative efficiency. The bidders should understand *bid quality* properly. Traditionally, game theory has been used to derive optimal strategies for the bidders in interactive environments. However, a negotiation mechanism generally does not have a dominant strategy equilibria; the bidders can at best speculate on other bidders' valuations or strategies rationally. Understanding own bid quality is essential for a bidder to place a good quality bid. Intelligent discount policy or incentive compatibility is an essential component of an efficient negotiation protocol. The service provider should be able to select a rational incentive strategy based on its capacity utilization, target revenue and profit.

In case of auction contract, a service provider may call *combinatorial auction* (CA) where instead of auctioning multiple items in a sequence or in parallel, CAs allow for bids on bundles of services. It is an efficient resource allocation mechanism. However, the bidders may face problems to express intelligent bids and need real-time intelligence. The bidders may face *strategic valuation complexity*: what should be the valuation of a bid; when and how much to bid. Thus, the service provider solves a non-linear constrained optimization problem through a set of intelligent moves and generates a profitable pricing plan. The provider adopts an optimal set of business intelligence moves and conduct private search to reach near the optimal point at the profit zone approximately.

Let us consider a specific scenario of multi-party negotiation in cloud computing; the concept of swing option is applicable to the trading of dSaaS. *Swing option* is a specific type of supply contract in trading of stochastic demand of a resource. It gives the owner of the swing option the right to change the required delivery of computational resources through short time notice [18]. It gives the owner of the swing option multiple exercise rights at many different time horizons with exercise amounts on a continuous scale. A typical swing option is defined by a set of characteristics and constraints [13]. There are predefined exercise times t_i, $i \in [1,2,..,n]$, $1 \leq t_1 < t_2 < \ldots < t_n \leq T$ at which a fixed volume of d_0 units of computational resources may be obtained. With a notice of specific short period, the owner of the option may use swing right to receive more (up-swing) or less (down-swing) than d_0 at any of n moments. The scheme permits swing only at g out of possible n time moments where $g \leq n$ is swing number constraint. A freeze time constraint forbids swings within short interval of the moments. The local constraints up-swing [α] and down-swing limits [β] define how much the requested demand d_i at time t_i may differ from d_0. There are two global constraints which restrict the total requested volume D within the contract period by maximum total demand (γ) and minimum total demand (λ). The option holder must pay penalty determined by a function ρ for violating local or global constraints.

***Swing option protocol**:*

Agents: B and S;

Inputs: S holds adjusted price ladder (P_a); B holds approximate demand, budget;

Negotiation issues: primary - combinatorial pricing plan; secondary - [$d,\alpha,\beta,\gamma,\lambda,\rho,g$];

S bids its optimal pricing plan P_o to B.

Set i = 0. Reference plan = P_o;

Repeat until the stopping criteria is satisfied:

  Set i = i + 1;

  B counter bids P_i^B to S or S counter bids P_i^S to B;

  $N^s(t, P^t_{i,B\rightarrow S})$ = *quit* if $t > T^S$ or

  accept offer if $u^S(t, P^t_{i,B\rightarrow S}) \geq u^S(t', P^{t'}_{i,S\rightarrow B})$ or

  counter offer $P^{t'}_{i,S\rightarrow B;}$

If both parties agree, output plan $P_f = P_i$.

Output: Final pricing plan [P_f];

In this negotiation protocol, the primary negotiation issue is a combinatorial pricing plan which depends on the negotiation of a set of secondary issues. For DaaS / dSaaS, the secondary negotiation issues may be up-swing limit [α], down-swing limit [β], maximum total demand [γ], minimum total demand [λ], penalty function [ρ] and number of swings [g] for a specific period. Similarly; the trading agents may negotiate different issues for other services. For SaaS, the issues may be number of concurrent users and usage period; for PaaS and IaaS, the issues may be the complexity of infrastructure. The service provider negotiates with a service consumer based on multiple parameters concurrently and finally settles the combinatorial pricing plan through the selection of optimal discount or incentive. The utility $u^a(t,P) = (RP_B - P)$ for service consumer and $(P - RP_S)$ for service provider; u: utility, P : pricing plan, t - time, T: deadline, RP : reservation price, N : Negotiation function..

5.1.4 Analytics

Analytics is another critical component of the pricing information system. It may not be useful at the initial phase of business life-cycle due to the non-availability of sufficient data. But, at the later phase of the business cycle, the service provider should analyze the past transactional data of different consumers and should try to discover business intelligence in terms of price-demand function, cross-selling opportunities and credit risk of the consumers. This intelligence is required to adjust the price ladder adaptively over a business cycle.

Price-demand function: This is a function approximation problem; it is the task of learning or constructing a function based on time series data. For a particular cloud computing service, the service provider holds a limited capacity over a finite period T. In the beginning of each period *t*, the provider knows the previous price and demand realizations, that is, $d_1, \ldots, d_{t-1}$ and $p_1, \ldots, p_{t-1}$. The price-demand function of the provider may be approximated as $d_t = \beta_0 + \beta_1 p_t + \epsilon_t$, that is, it depends on the current period prices p_t, unknown parameters β_0, β_1 and a random noise $\epsilon_t \sim N(0, \sigma 2)$. The firm's objectives are to estimate its demand dynamically and set prices in order to maximize its total expected revenue.

Input: Time series data $(d_t, p_t)_{t=1,\ldots,n}$;

Output: $d_t = \beta_0 + \beta_1 p_t + \epsilon_t$;

Algorithm: Recurrent or backpropogation learning algorithm for a feedforward neural network

Cross-selling opportunities: Each service consumer approaches the service provider with an approximate demand plan and budget; the provider analyzes the profile of the consumers and tries to assess the credit risk, price sensitivity and cross-selling opportunities. Price sensitivity is basically the percentage change in revenue given one percent change in price. It depends on the economics, search and usage of the service consumers and the competitive situation. An efficient pricing mechanism should be able to compute the price sensitivity, cross selling opportunities and credit risk of the service consumers by mining trading data. The provider can store strategic data of the consumers from various transactions and apply association mining algorithms on the stored filtered data. This is a data driven price discovery mechanism. The provider should analyze lost quote data. Initially, many consumers show their interests to buy a service. Later, they decide not to buy the same for various reasons. Analytics can analyze lost quote data and can predict the price sensitivity of the consumers correctly. The service provider can leverage individual transactions into long lasting customer relationship across multiple types of IT services. The pricing mechanism should not only consider the profitability of a single service but it should also analyze the interrelationship among various cross services through *association rule mining*. An association rule mining algorithm generates all frequent service set having support above minimum support. It also generates all confident association rules from the frequent service set having confidence above minimum confidence.

Credit risk: The objective of analytics is also to build an efficient classifier that can predict credit risk of the service consumer by mining financial data. The prediction system requires a robust training algorithm such *as support vector machine*. This is a linear learning system that finds the maximal margin decision boundary to separate positive and negative training data. The provider can estimate the credit risk of each service consumer based on financial data such as financial ratios, revenue and credit history of the consumers.

6. CONCLUSION

The following section concludes the work by highlighting some critical observations on cloud computing pricing mechanism. Firstly, the basic objective of the proposed pricing mechanism is to support a smart market of cloud computing trading efficiently. The growth of computational power, communication networks and user interface design can implement such complex algorithm mechanisms efficiently. Today's research in computer and management science is trying to design market structure, organization, trading mechanisms and DSS in a complex and integrated dynamic business environment. A Smart market of cloud computing should focus on rational pricing mechanism, optimization, interactive market design, DSS and computational tools to improve the allocative efficiency of different types of services. Efficient algorithmic mechanisms are expected to provide necessary computational intelligence for improved decision-making in complex and dynamic business environment.

Secondly, cloud computing is perceived to be used as the next utility. Traditional software pricing mechanism may not be a suitable option for utility or cloud computing. Till now, software and other IT enabled enterprise solutions are not treated as a utility. The cloud computing market requires a new paradigm and reliable pricing algorithm. Cloud computing is the future solution of e-governance; the basic objective is to get high quality of service at low reasonable cost; it is an emerging trend of enterprise resource planning strategy. The backbone of e-governance is service oriented architecture. Two main trends are seen in e-governance: constant development of IT architecture and rapid increase of users' skills and knowledge of IT. Public sector enterprises should take advantage of these improved conditions for the development and deployment of e-government solutions.

Thirdly, the pricing of cloud computing (e.g. SaaS) should not be as same as that of traditional experimental trial and error based software pricing. Random heuristic driven pricing (e.g. free software) has resulted significant loss of revenue and profit in ICT industry; it has encouraged piracy, arbitrage and cannibalization problems. Once, a large chunk of consumers are habituated to use pirated software products, it is difficult for them to buy original products in future. The innovative solutions were not perceived and utilized properly by many consumers. Therefore, the rate of penetration of IT based enterprise solutions is not as high as compared to the expectation. It is really complex to know the valuation of the service consumers precisely by trial and error method. Another important aspect is that the price of cloud computing is a time variable function. At first, the price ladder is computed at time t_1 and the final price is negotiated at time t_2. Therefore, the pricing mechanism should be adaptive.

Fourthly, the pricing of cloud computing is a complex combinatorial issue since there may be different types of feasible combinations possible: SaaS, DaaS, dSaaS, PaaS, IaaS, public, private and hybrid cloud computing. The pricing mechanism of cloud computing should consider miscellaneous types of cost elements: cost of hardware and software solutions, promotion, operational costs and cost of communication. The provider must charge its operational and promotional cost to the consumers precisely otherwise it may result significant loss of revenue. The pricing mechanism should define discount, penalty and payments terms intelligently since these factors have direct impact on the revenue of the provider. The discount factor should be set such a way that it boosts the demand of the consumer and provides desired competitive intelligence.

Finally, should a service provider adjust its price dynamically according to the demand of the consumers? Uncertainty in demand suggests that a service provider can benefit from dynamic pricing. The provider may adjust its price adaptively: when demand is strong, set a high price, and when demand is weak, set a low price. Through dynamic pricing, the provider can exploit high demand by charging a high price or low demand by charging a low price for optimal capacity utilization. Strategic consumers may sense the intelligence of dynamic pricing strategy and may take the buying decision cautiously. But, a static pricing strategy may result significant loss of revenue and profit of the service provider. The dynamic pricing may be effective as compared to static pricing if the service provider sets a modest base price and then adjusts it according to the complexity of cloud computing service; it should not raise the price randomly in response to strong demand of the consumers. An efficient adaptive pricing mechanism tries to strategize: where do the trading agents want to go in future? plan: how do they get there? monitor: how are they doing? act and adjust: what do they need to do differently to improve? Today's digital economy is ready for complex dynamic discriminatory value based pricing. It is possible to compute right price to the right service consumers at right time through business intelligence.

REFERENCE

[1] Armburst, M. et al. (2010). A view of cloud computing, Communications of the ACM, 53(4), 50-58.

[2] Anderson,E.T. and Dana,J.D. (2009). When is price discrimination profitable? Management Science,55(6), pp. 980-989.

[3] Buyya, R. and Bubendorfer, K. (2010). Market-oriented grid and utility computing, John Wiley & Sons, Inc.

[4] Buyya, R., Yeo, C.S., Venugopal, S., Broberg,J. and Brandic, I. (2009). Cloud computing and emerging IT platforms: Vision, hype and reality for delivering computing as the 5th utility. Future generation computer systems.

[5] Cai, Y. and Daskalakis, C. (2011). Extreme-value theorems for optimal multi-dimensional pricing, SAGT'2011.

[6] Chawla,S., Hartline,J.D. and Kleinberg,R. (2007). Algorithmic pricing vs. virtual valuations. Proceedings of ACM conference on Electronic Commerce, California, USA.
[7] Chakraborty, S. (2007). A study of several privacy-preserving multi-party negotiation problems with applications to supply chain management. Doctoral dissertation (unpublished), Indian Institute of Management Calcutta.
[8] Degabricle,J.P. and Pym, D. (2007). Economic aspects of a utility computing service. Technical report HPL-2007-101, HP Labs, Bristol.
[9] Elmarghraby,W. and Keskinocal, P. (2003). Dynamic pricing: Research overview, current practices and future directions. Management Science, 49(10), pp. 1287-1309.
[10] Forouzan, B.A. (2007). Cryptography & network security. McGraw Hill.
[11] Furht,B. and Escalante, A. (2010). Handbook of Cloud Computing. Springer.
[12] Goldreich, O. (2004). Foundations of Cryptography, Basic Applications. Volume 2. Cambridge University Press.
[13] Keppo,J.(2004). Pricing of electricity swing options. The Journal of Derivatives, 11(2), pp. 26-43.
[14] Narahari,Y., Raju,C.V.L., Ravikumar,K. and Shah,S. (2005). Dynamic pricing for electronic business. Sadhana, 30(2,3), pp. 231-256.
[15] Nisan,N., Roghgarden,T, Tardos, E. and Vazirani,V.V. (2007). Algorithmic game theory. Cambridge University Press. Chapter 13, pp. 331-361.
[16] Paleologo, G.A. (2004). A methodology for pricing utility computing services. IBM System Journal,43(1), pp. 20-31.
[17] Pedersen, T.P. (1991). A threshold cryptosystem without a trusted party. In D.W.Davies, editor, Advances in cryptology – Eurocrypt'91, pp. 522-526, Berlin,1991, LNCS volume 547, Springer-Verlag.
[18] Pflug,G.C. and Broussev, N. (2009). Electricity swing options: behavioral models and pricing. European Journal of Operational Research.197, pp. 1041-1050.
[19] Phillips,R.L. (2005). Pricing and revenue optimization. Stanford University Press.
[20] Rusmevichientong,P., Roy, B.V. and Glynn,P.W. (2006). A nonparametric approach to multiproduct pricing. Operations Research, 54 (1), pp. 82-98.
[21] Wilson, R.B. (1997). Nonlinear pricing. Oxford University Press.
[22] Yeo,C.S. and Buyya,R. (2007). Pricing for utility driven resource management and allocation in clusters. International Journal of High Performance Computing Application,21(4), pp. 405-418.
[23] Hong, Y-J, Thottethodi, M., Xue, Jiachen (2011). Dynamic server provisioning to minimize cost in an IaaS cloud, SIGMETRICS Performance evaluation review, volume 39, issue 1.
[24] Beloglazov, A., Buyya, R., Lee, Y.C and Zomaya, A. (2011). A taxonomy and survey of energy efficient data centers and cloud computing systems. Advances in computers, volume 82.
[25] Wang, H., Jing, Q., Chen, R., He, B., Qian, Z and Zhou, L. (2010). Distributed systems meet economics: pricing in the cloud. Hotcloud'10.
[26] Daoud, A-A, Agarwal, S. and Alpcan, T. (2009). Brief announcements: Cloud computing games: Pricing services of large data centers. DISC'09.
[27] Macias, M. and Guitart, J. (2011). A genetic model for pricing in cloud computing markets. SAC'11.
[28] Menache, I., Ozdaglar, A. and Shimkin, N. (2011). Socially optimal pricing for cloud computing resources. Valuetools'11
[29] Klemperer, P. (1987). The competitiveness of markets with switching costs. The RAND Journal of Economics, 18:138–150.
[30] Sundararajan, A. (2003). Nonlinear pricing of information goods. Management Science.
[31] Bichler, M., Gupta, A. and Ketter, W. (2010): Research Commentary: Designing Smart Markets. Information Systems Research 21(4), pp. 688 – 699.
[32] Bertsimas, D. and Perakis, G. (2001). Dynamic pricing, a learning approach, MIT.
[33] Gupta, A., Jukic, B., Stahl, D.O. and Whinston, A.B. (2010). An analysis of incentives for network architecture investment for different pricing strategies. Information Systems Research, pp 1-21. Published online.
[34] Miettinen,K. (1999). Nonlinear multiobjective optimization. Kluwer Academic Publishers, Boston.
[35] Ganesh, A.,Laevens, K. and Steinberg, R. (2006). Congestion pricing and non-cooperative games in communication networks.
[36] Lahiri, A., Dewan,R.M. and Freimer,M. Pricing of wireless services : service pricing vs. traffic pricing. Information Systems Research, pp. 1-18.
[37] Chakraborty, S., Sehgal S.K. and Pal A.K. (2005), Privacy-preserving discriminatory pricing protocol for supply chain management, 4th Security Conference, Las Vegas, USA.

Appendix :

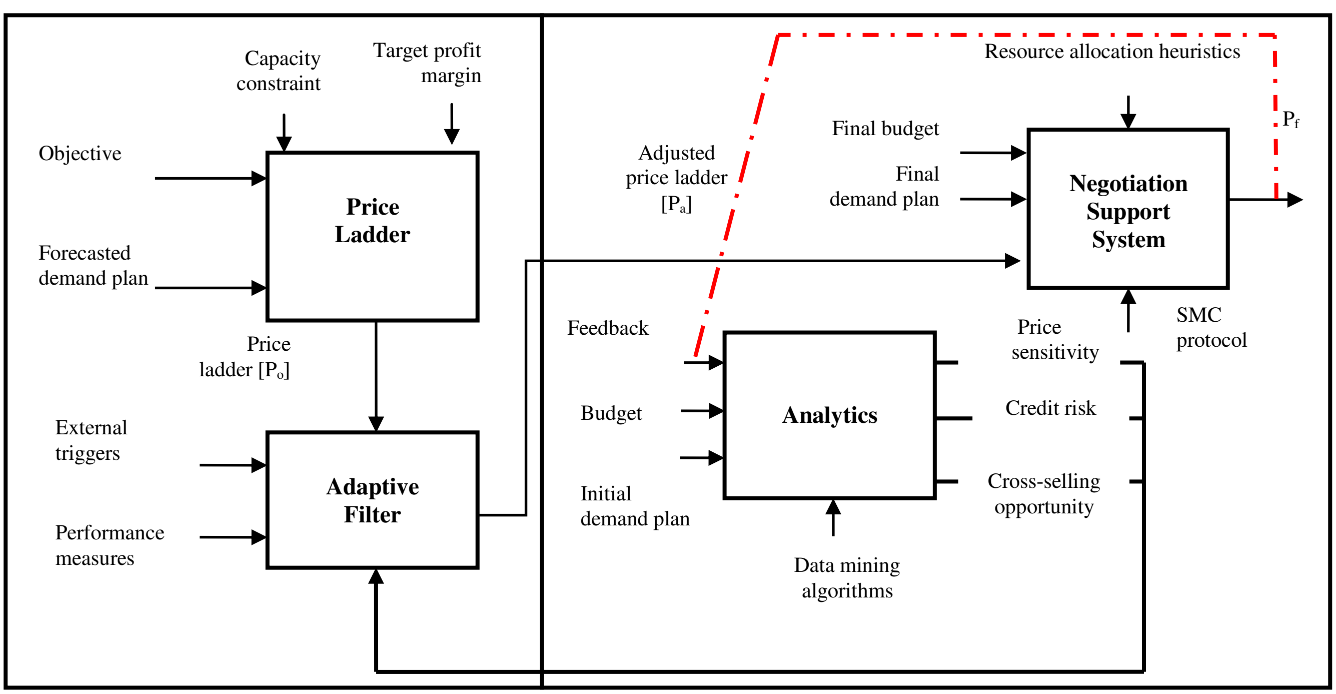

Figure 1: Information System Schema for Discriminatory Pricing Mechanism

SL No.	Cloud computing services	System parameters	Time duration	Demand band	Unit price	Discount factor	Penalty factor	Pay terms
	Postpaid							
1	SaaS : Application 1	No. of concurrent users	$T_1<T\leq T_2$	$n_1<n\leq n_2$	P_1	-	-	T_3
2	SaaS : Application 1	No. of concurrent users	$T_1<T\leq T_2$	$n_2<n\leq n_3$	P_2	-	-	T_3
3	SaaS : Application 2	No. of concurrent users	$T_1<T\leq T_2$	$n_3<n\leq n_4$	P_3	-	-	T_3
4	IaaS	Incoming & outgoing bandwidth (GB)	$T_1<T\leq T_2$	-	P_4	-	-	T_3
5	IaaS	Incoming & outgoing bandwidth (GB)	$T_3<T\leq T_4$	-	P_5	-	-	T_3
6	PaaS (Standard)	Computing time (instance hours)	$T_1<T\leq T_2$	-	P_6	-	k'_6	T_3
7	PaaS (Non-standard)	Computing time (instance hours)	$T_1<T\leq T_2$	-	P_7	-	k'_7	T_3
8	dSaaS	GB /month	$T_1<T\leq T_2$	$d_1<d\leq d_2$	P_8	k_8	-	T_4
9	dSaaS	GB /month	$T_1<T\leq T_2$	$d_2<d\leq d_3$	P_9	k_9	-	T_4
10	DaaS	Per 10K requests	$T_1<T\leq T_2$		P_{10}	-	-	T_3
11	SaaS+IaaS+PaaS+dSaaS		$T_1<T\leq T_2$	-	P_{12}	-	-	T_3
12	dSaaS + DaaS		$T_1<T\leq T_2$	-	P_{11}	-	-	T_3
13	SaaS + PaaS		$T_1<T\leq T_2$	-	P_{13}	-	-	T_3
14	IaaS + dSaaS		$T_1<T\leq T_2$	-	P_{14}	-	-	T_3
15	CaaS	voice calls, text messages, internet	$T_1<T\leq T_2$	-	P_{15}	-	-	T_3
	Prepaid	Define price for SL no. 1-15						T_5

Table 1: A sample price ladder for cloud computing