Available online at www.sciencedirect.com

ScienceDirect

Journal of Parallel and Distributed Computing 00 (2018) 1–14

Procedia Computer Science

Multi-Attribute Combinatorial Reverse Auction Model for Resource Procurement in Fog Integrated Cloud Architecture

Anubha Aggrawal[a], Neetesh Kumar[b], Deo Prakash Vidyarthi[c], Rajkumar Buyya[d]

[a]*Department of Computer Science & Engineering, Shri Mata Vaishno Devi University, Kakaryal, Katra, Jammu and Kashmir, India-182320.*
[b] *Department of Information Technology, ABV-Indian Institute of Information Technology and Management Gwalior, India-474015.*
[c]*School of Computer and Systems Sciences, Jawaharlal Nehru University, New Delhi, India, 110067.*
[d]*Cloud Computing and Distributed Systems (CLOUDS) Lab, School of Computing and Information Systems, The University of Melbourne, Australia, 3010.*

Abstract

Fog computing is an emerging service-oriented market with cloud association to fulfill mobile users resource demands for real-time applications. This work aims to develop a Fog Integrated Cloud Auctioning Model (FICAM) for resource provisioning using multi-attribute combinatorial reverse auction mechanism. Due to the unique need of the data sources mobility and limited resources in fog, existing cloud computing reverse auction architecture and techniques cannot be directly applied for resource procurement in fog integrated cloud architecture. To overcome this architectural limitation, we have developed a reverse auction based model which includes the customer, broker, fog providers, cloud providers and fog and cloud providers as participants. A pricing scheme is proposed which divides prices into three types, depending on resource requirement, i.e., local fog, remote fog, and cloud. A truthful, robust and fair algorithm for resource allocation is proposed which considers response time, data source mobility requirements and fog resource limitation. To encourage providers to bid truthfully, an incentive scheme inspired by Vickery Auction is proposed. An algorithm for resource procurement is also proposed in which instead of all resources proposed by winning provider in bundle; algorithm takes only that much resources as required by customer i.e., acquire as per need. Experimental results show that the proposed model offers a lower resource procurement cost in polynomial time in comparison to other states of the art. Customer is given bundle discount based on certain threshold, and that discount will be in a ratio of amount of resources taken by amount of the proposed resources. To the best of authors knowledge, this is the first model which implements an auctioning model for resource procurement in fog integrated cloud architecture.

Keywords: Fog computing, resource procurement, reverse auction, fog service provider, Internet of Thing (IoT).

1. Introduction

Internet of Things (IoTs) era has led to customers in real-time computing requirements of several applications such as smart home, infrastructure, healthcare, retail, transportation, security, surveillance and other industrial necessities. Most of such applications have a rigid time-sensitive requirement as a need for speedy processing of streamed data so that action can be initiated in real-time. For example, in home security systems if IoT devices sense a certain type of mishap then call should be made immediately (in real-time) to the police and homeowner. Nowadays, smart wearable, smart vehicles (Internet of Vehicles (IoV)), mobile IoTs are in practice; to make computing decisions in real-time requires placement of computing resources near them.

Fog computing is an emerging and best-fitted solution considering necessary constraints of the current requisite real-time computing[6]. Fog computing provides computing resources close to hosts with high network bandwidth and offers smooth VM migration [24] as soon as the data source goes out of coverage area of local fog server. In this scenario, if objects are mobile then, mobile devices and fog resources can communicate efficiently. According to research studies [25], if fog acts as an intermediate layer between cloud and consumer then not only the load on the cloud, but the operational cost of cloud data centers also gets reduced. Even Cisco [3] stated that today's cloud model is not designed for volume, variety, and velocity of data that IoT generates. Thus, there is a high demand of computing resources for real-time computing (IoT applications) utilizing fog computing. The necessity of fog computing in real-time computation with enormous fog providers motivated us to propose a model for resource procurement considering fog integrated cloud computing architecture.

The reverse auction is an auctioning scheme in which customer lists his/her requirements and according to the customer's requirement, providers bid. In this scheme, there is one customer and many bidders. There are many other types of resource procurement schemes present like fixed price, dynamic

price, etc. Nevertheless, the reverse auction is an attractive scheme with the reason that it makes the market more competitive. It also helps in lowering resource procurement cost [10]. Due to reverse auction scheme, customers are able to acquire resources at the best price. In reverse auction, the multi-attribute combinatorial reverse auction is chosen as it helps in comparing providers not only by the price but also by other attributes. According to study [20], some non-pricing attributes also play a significant role in a successful auction.

Few reverse auction mechanisms are explored for resource procurement in cloud computing. Cloud resources reverse auction procurement involves three participants:- broker, customer and cloud providers. Customer submits the requirement to broker and then, broker invites cloud providers for bidding. Broker declares winning cloud providers based on some criterias, winning providers allocate resources to the customer. Cloud computing reverse auction architecture model is limited to cloud providers, customer, broker as its vital components. However, in fog computing other members like fog providers, both fog and cloud providers are also required to take part with varied needs. In general, the nature of fog providers and customer's resource requirements are quite different due to current IoT era (real-time streamed data processing), limited fog resources and dynamic fog pricing schemes. Thus, as a need of current computing era and to successfully carry-out handshake between IoT and fog computing technologies; a new architecture model for an efficient auction mechanism is required to develop which facilitates the consumer, fog providers, cloud providers and fog and cloud providers to participate. Customer should be flexible to use local or remote fog resources, on the contrary, fog and cloud pricing models are different. Thus, an effective pricing scheme is needed. Considering the above-mentioned necessities, canonical reverse auction model for cloud computing cannot be directly applied to the fog integrated cloud computing auctioning architecture as shown in Figure 1. Addressing above-mentioned limitations, an efficient auctioning model for fog integrated cloud computing architecture is proposed.

In order to conduct this work, we followed a baseline cloud auctioning model [1]. Novelties of fog integrated cloud resource reverse auctioning over baseline model are listed as follows:

1. It reduces resource procurement cost as resources are acquired as per needs of the customer (not as per resources proposed by the provider in the quotation). The proposed scheme overcomes the limitation of the baseline model whereas customer had to acquire the total resources in the bundle that may be more than the customer's requirement.
2. It serves time-sensitive requirements of the customer which is not considered in the baseline model. The customer can specify how much response time it requires from each machine in the requirement.
3. The proposed model supports the collaboration between fog and cloud providers/services/resources in order to enable IoT oriented services (in real-time) and improve the quality of experience.
4. This work proposed a framework to act on dynamic pricing scheme which is not considered in baseline model.
5. Customers experience reduced latency as they get resources at edge also, and hence, improves the quality of service.
6. It considers the case of data source mobility, best of authors knowledge, which is not handled by any cloud resource reverse auctioning model.

Significant contributions are listed as follows:

1. A multi-attribute combinatorial reverse auction architecture is proposed for resource procurement in fog integrated cloud architecture. The broker, customer and service providers of cloud and fog resources are the essential components as shown in Figure 1.
2. Customer list requirements to the broker who is at broker layer. Fog, cloud, fog and cloud providers give their quotation to the broker at broker layer. Broker determines winners, and respective winner providers offer their services to the customer.
3. The proposed pricing scheme (Figure 2) consists of three types of pricing: local, remote, and cloud pricing. The provider charge local and remote prices if allocated Virtual Machines (VMs) are on fog. The local price of VMs is charged when the data source is local; and as soon as the data source migrates from the local place (i.e., district/city) then the user is charged with the remote price of respective VM. The provider charges user with cloud price of VM if allocated VM is on the cloud.
4. A new scheme inspired from Vickery [22] [12] is also proposed to offer incentives to providers to motivate them to bid on truth value.
5. A multi-attribute combinatorial reverse auction algorithm (runs in polynomial time) for resource procurement is also developed; it determines winners and quantity of VM's to be taken from each provider's bundle and offers incentives to the truthful providers. It also calculates the final bill on the basis of threshold, bundle discount and the ratio of quantity of used resources by quantity of proposed resources, and it also applies a penalty if required.
6. The proposed Fog Integrated Cloud Auctioning Model (FICAM) is tested under eight different scenarios. Comparative results evidenced that the resource procurement cost of FICAM is significantly reduced over other states of the art.

The rest of this paper is organized as follows. Section 2 briefs about the related work on the reverse auction. Section 3 describes FICAM modules. Section 4 details about properties, comparison, and performance of the algorithm. Lastly, section 5 concludes the paper along with future work.

2. Related Work and Background

Laun el at. [9] discusses system architecture of fog computing and compares fog computing with cloud computing on

the basis of design and research issues. Yi el at. [24] demonstrated the concept of VM migration in fog computing; three-layer architecture; goals like latency, efficiency and generality; challenges like the choice of virtualization technology; resource provisioning; and applications like smart home, smart grid, vehicle of fog computing. From the study, it is inferred that fog computing has enormous scope in the future IoT applications. Particularly, it will be used for real-time applications.

Zaman el at. [26] proposed two combinatorial auction mechanisms: CA-LP and CA-GREEDY. They proposed dynamic pricing schemes and showed that these mechanisms are better than fixed price resource allocation schemes in cloud computing. Experimental results confirmed that their mechanisms could improve the revenue of the provider.

Modica el at. [11] stated that from the provider perspective, using an effective dynamic pricing strategy can make a difference between making a profit or a loss: in an open cloud marketplace, the strategy is even more important due to its dynamic and volatile nature. He used reverse cloud auctioning to sell scanty computing resources. Through experimental results [11] showed that accurate modeling and tuning of auction parameters can lead to maximize the utilization of resources.

Manoochehri el at. [10] discussed advantages, risks of reverse auction mechanism and inferred that it benefits customers beside helping to cut down the price. It also takes decision for the best provider from the large set of providers. Song el at. [21] introduced a combinatorial reverse auction scheme in which providers tie up with each other to fulfill customer's demand at best price. This enhances their chance of winning. Vries el at. [23] state that if instead of a single item, bidders bid on a group of items (bundle) then resource procurement cost gets reduced. Prasad el at. [16] addressed the issue of procuring multiple resources from several cloud vendors. He used auction for this. Experimental results showed that combinatorial acution is superior to the sequential auction. Prasad el at. [15] presented a cloud resource procurement approach which not only automates the selection of an appropriate cloud vendor but also implements dynamic pricing. He proposed three possible mechanisms based on properties of auction such as individually rational, budget balance, incentive compatible : cloud-dominant strategy incentive compatible (C-DSIC), cloud-Bayesian incentive compatible (C-BIC), and cloud optimal (C-OPT). C-DSIC uses VCG mechanism and is incentive compatible, individual rational but not budget balanced. C-BIC is Bayesian incentive compatible, budget balance but not individual rational. C-OPT is not only Bayesian incentive compatible, but also individually rational.

The methods of auction discussed above, consider the price as an attribute for winning. However, Quality of Service (QoS) parameters i.e., non-price attributes are equally important factors while determining winners. Pla el at. [14] introduced a multi-attribute reverse auction mechanism, VMA2, and classified attributes in three types: verifiable attributes, auctioneer provided attributes, and non-verifiable attributes. He stated that verifiable attributes and auctioneer provided attributes ensures truthfulness and trust. The mechanism gave incentive to truthful bidders. VMA2 was tested against unattributed auctions (auctions which consider the price as the one and only attribute) and it was found that its resource procurement cost was less than unattributed auctions. VMA2 encouraged bidders to bid their true value. But if only those providers who offer a cheaper price and high-quality service win, then other providers lose interest and leave auction market. Due to this, remaining providers may start controlling the market. This is bidder drop out problem [7]. Thus, auction mechanism should be fair to all providers.

Gaurav el at. [1] introduced a fair reverse auction mechanism for resource procurement in cloud computing, i.e., TFMCRA. It is a multi-attribute, combinatorial and truthful reverse auction mechanism. For encouraging providers to bid truthfully, TFMCRA uses Vickery payment. To maintain fairness, if a provider p_x wins auction then it reduces p_x chances of winning and increases chances of winning of non-winning providers in next round . It considers price as well as non-price attributes for winner determination. It is truthful, fair, multi-attribute, combinatorial reverse auction model for resource procurement in cloud computing and executes in polynomial time. Thus, this work is considered a baseline for resource procurement in fog integrated cloud computing architecture. However, this model is limited to cloud computing as a two-tier architecture model. Fog computing architecture is different from cloud computing model; it is a three-tier architecture model in general. Pricing schemes of fog computing are also different as the type and demand of fog resources are different. Further, data source mobility issue [2] has been unconsidered in baseline model [1]. Consequently, there is a need of designing a new resource procurement mechanism for the three-tier architecture: consumer, fog, and cloud.

3. Proposed Model : FICAM

This section details the proposed FICAM that involves architecture, pricing model, and resource procurement algorithm for fog integrated cloud computing three-tier architecture. FICAM consider differences between fog and cloud computing especially data source mobility, location awareness and dynamic pricing schemes based on real-time workloads. Further, in base model [1] quantity of resources that customer gets after auction could be more or same that the customer has demanded. But then, using FICAM the customer gets the same quantity of resources that the customer requested for.

3.1. Fog Integrated Reverse Auction Architecture

Figure 1 shows the three-tier architectural model of the proposed FICAM. FICAM allows customers to acquire resources at the best prices following pay per use model according to their necessity along with multiple QoS. The participants in this model are Customer, Broker, Fog Providers, Cloud Providers and Fog&Cloud Providers.

Broker :- The broker is a middleman between the customer and providers (fog, cloud, fog and cloud) [1], the broker upon getting requirements from the customer invites providers to submit their quotation. Based on history, quotation and customer's requirement broker selects winners. The broker also imposes

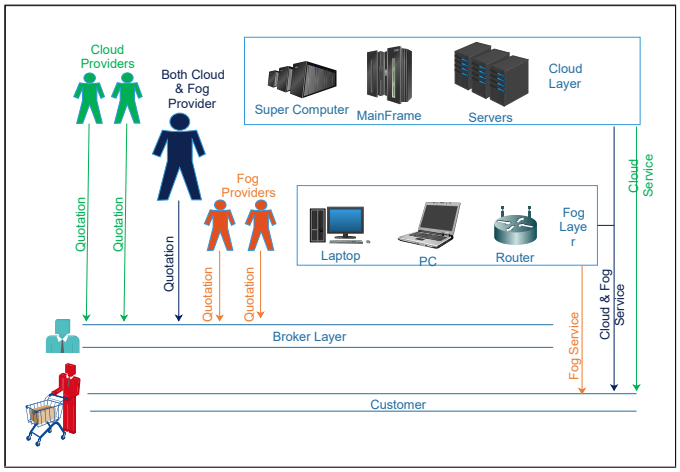


Figure 1: Fog Integrated Cloud Reverse Auctioning Architecture

the penalty on those providers who cheat and urges them to grant compensation to customers. The broker provides an incentive to winners for biding on truth value.

Customer :- The customer can be a person or a company. The customer today has two types of requirements: 1. Real-time/time-sensitive requirement 2. Normal requirement. Various IoTs, gaming applications, smart vehicles, etc. demand time-sensitive requirement. The customer submits configuration of all types of machines to the broker and specifies response time requirements from each configuration machine.

Cloud Provider (CP) :- CP provides cloud computing resources for rent at the cloud layer. The customer's requirement is verified; if it is normal and CP's can offer all configuration machines, then they surely take part. But if the customer's requirement is time sensitive, then CP's may or may not participate. They can only participate if they can satisfy specified response time specifications of the customer.

Fog Provider (FP) :- FP provides fog services/resources for rent at the fog layer. If the customer's requirement is time sensitive, and FP can provide all required configuration machines, then they surely take part as fog is devised for time-sensitive applications. On the contrary, if the customer's requirement is normal, then FP's may or may not participate. It depends upon the workload on fog resources and how much profit they may earn or how much are their chances of winning.

Fog and Cloud Provider (FCP) :- These providers offer services of both fog and cloud resources. FCPs are the owner of both fog and cloud resources. Sometimes the customer wants large configuration machines that cannot be provided by the fog providers (as fog provides small configuration machines). Sometimes the customer wants a quick response time that cannot be provided by cloud providers. The customer could need both these types of machines. For such a scenario to serve the customer with better QoS, FCPs are needed which can give both fog and cloud resources. These providers may either own both fog and cloud resources or may have tied up with some other provider to provide the customer both fog and cloud resources. As they provide both fog and cloud resources, they can participate in any type of customer requirement.

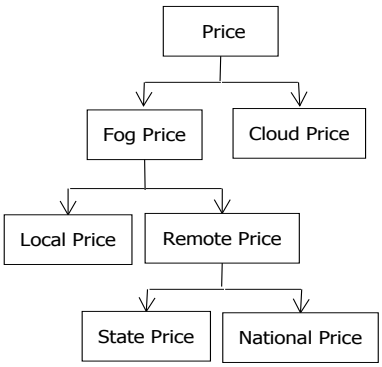


Figure 2: Pricing Schemes used by FICAM

3.2. Fog Pricing Schemes

Figure 2 shows the possible pricing types based on computing requirements. To deal with both fog and cloud resources, pricing schemes are categorized into fog price and cloud price. Technically, fog resources are meant for time-sensitive applications; therefore, meeting response time deadline is a necessity. In order to meet the deadline for computation when data source is mobile, VM migration [24] occurs in fog computing. As IoT moves, its resources are migrated to the nearest fog center (i.e., nearest to its current location). In this transfer, VM migration cost is involved. The load is different on different fog centers so because of dynamic pricing, pricing for same VM in different fog centers of the same provider will vary. Keeping VM migration and dynamic fog pricing in mind, fog prices are divided into two types: local and remote. In FICAM, providers indicate three types of prices for a VM in their quotation.

Local Price :- The allocated fog resources are registered in a fog center nearest to the data source. The base area is an area in which that fog center is located. The customer is charged according to local price rate of fog machine as long as data source remains in the base local area like in registered city or district.

Remote Price :- Remote prices are charged only when the data source moves from base area to another area. This way, resources are transferred from fog center/server in the base area to a fog center in another area. From that point of time, the customer is charged according to remote price rates of the fog machine. Remote price depends on the range of the mobility of data source. Based on the range of data source mobility, the remote price is classified into two types: State Remote Price, National Remote Price. This division is inspired by types of roaming in telecommunication [18].

a) State Remote Price :-In this scheme, the limit of mobility of the data source is in the state of the registered base fog center. Consider a scenario; a customer is operating the smart vehicle and that vehicle uses fog resources. Eventually, smart vehicle crosses the district range and moves within state boundaries then, provider charges the customer as per the state remote price of acquired fog machines.

b) National Remote Price :- In this case, the data source's mobility limit is the nation. The data source can move anywhere in the nation, and VM is migrated to the nearest and fog center according to availability.

Cloud Price :- If VMs are allocated on the cloud then, the price from the customer is charged at cloud price rate of VMs.

If a provider offers only cloud resources then, local price and remote price of each VM is zero; else if only fog resources then, cloud price of each VM is zero. If a provider offers both fog and cloud resources, then for a random VM if it is in the cloud then, it's local and remote price is zero; else its cloud price is zero. This zero helps broker and customer to know about resource allocation whether it is in the cloud or fog.

3.3. Algorithmic Design

We present an algorithm for resource procurement based on fog integrated cloud reverse auction architecture. For making algorithm incentive compatible, an incentive approach is designed similar to the one used in [1] Vickery auction [22] considering the special properties of FICAM model like data source mobility and limited resources.

The methodology of the algorithm design is summarized in a sequence of steps as follows:

1. Requirement vector is given to broker by the customer.
2. Providers are invited by the broker to submit their quotation.
3. Bidding is completed.
4. Total bundle price, i.e., $tlp + trp + tcp$ of each provider is estimated by the broker.
5. Since the history of providers also matters, hence, past history attributes are added in the quotation by the broker. This version of the quotation is referred as extended Quotation.
6. Winner and Final quantity of VMs taken (i.e., fqt vector of each VM type from each winner provider) is determined by the broker.
7. The incentive is given to winners by the broker for biding truth value.
8. Resources are allocated to the customer.
9. The broker is asked by the customer to generate final bill and customer is asked to transmit a report on QoS delivered by the provider based on the customer's feedback.
10. Providers are asked by broker to submit flt, frt, fct matrix.
11. Finally, the bill is generated by the broker for each winning provider and is conveyed to the customer.

Rest of this section details about the algorithm.

3.3.1. Input

Algorithm takes the following inputs from customer.

Machine Configuration Vector (mcf):- It contains configurations of machines that customer wants on rent. This vector is denoted by $(mcf^1, mcf^2, ..., mcf^n)$.

Quantity Required Vector (qr):- It contains the quantity of each type of machine that customer wants. It is denoted by$(qr^1, qr^2, ..., qr^n)$, where qr^i corresponds to quantity of the instance of mcf^i required by customer.

Table 1: Notation

Notation	Description	Notation	Description
mcf	mach configuration	*qr*	quantity required
qo	quantity offered	*qt*	quantity taken
et	expected time	*resp_time*	response time
pb_l	local probability	pb_r	remote probability
lm	data source limit mobility	*quot_vec*	Quotation vector
lp	initial local price matrix	*rp*	initial remote price matrix
cp	initial cloud price matrix	*tlp*	total local price
trp	total remote price	*tcp*	total cloud price
repu	reputation	*pri*	priority
ctq	current taken quantity	*req*	requirement
discob	discount on bundle	*ebp*	estimated bundle price vector
flt	final local time matrix	*frt*	final remote time matrix
fct	final cloud time matrix	*fqt*	final quantity taken matrix
fb	final bill Vector	*extquot_vec*	extended Quotation vector
P	providers vector	*sorted_provider*	sorted provider list
rw	runner up winner	*w*	winner

Expected Time Vector (et) :- It contains estimated time usage of each machine. It is denoted by$(et^1, et^2, ..., et^n)$, where et^i is the expected time for which customer will use mcf^i.

Response Time Vector (res_time):-The response time is the expected response time from each machine for real-time applications. It is denoted by $(res\_time^1, res\_time^2, ..., res\_time^n)$, where res_time^i corresponds to the response time that customer wants from mcf^i.

Local and Remote probability (pb_l, pb_r):- The response time is also dependent on distance, especially if the customer is utilizing resources for time-sensitive applications. Consequently, the local and remote probability of the data source is required to be known. Let the probability of locally used resources is denoted by pb_l and probability of remotely used resources is denoted by pb_r.

Limit of Mobility (lm):- It denotes to what extent the data source can be expectedly remote. The limit can be state or national, if the limit is national it means the data source can move within the national boundaries. It is denoted by lm =state / national. Nevertheless, it is to make sure that despite mobility of data source, response time (deadline) for real-time applications are required to be met. If an application is not real-time response time does not matter, therefore, VM migration does not take place even if resources are allocated to fog and data source moves. In this case pb_l=1 , pb_r=0 and lm=local.

Quality of Service Vector and Weight Vector (Qos_vec and $weight\_vec$):- These vectors consist of the value of non-price attributes. Let the length of this vector be denoted by Qos_len. $weight\_vec$ holds the weight given by the customer to these QoS.

From a provider say p_x quotation is taken as input that consists of following:

Initial Local Price Vector(lp_x) :- This vector contains local price of each configuration machine. It is represented by $(lp_x^1, lp_x^2, .., lp_x^n)$ where lp_x^i corresponds to the local price of mcf^i provided by p_x. Value of lp_x^i is zero if p_x is providing mcf^i on cloud layer.

Initial Remote Price Vector(rp_x) :- This vector contains remote price of each configuration machine according to the lm specified in the customer's requirement. Let it be denoted by $(rp_x^1, rp_x^2, ..., rp_x^n)$ where rp_x^i corresponds to the remote price

of mcf^i provided by p_x. Value of rp^i_x is zero if p_x is providing mcf^i on cloud layer or if the data source is local i.e., pb_r=0.

Initial Cloud Price Vector(cp_x):-This vector contains cloud price of each configuration machine. It is denoted by ($cp^1_x, cp^2_x, .. , cp^n_x$), where cp^i_x corresponds to the cloud price of mcf^i provided by p_x. The value of cp^i_x is zero if p_x is providing mcf^i on fog layer.

Quantity Offered Vector (qo_x):- It is the quantity of each machine configuration that provider is able to provide. It is denoted by ($qo^1_x, qo\ ^2_x, ..., qo^n_x$), where qo^i_x is the quantity of i_{th} machine configuration that provider p_x is offering.

Discount on bundle ($discob_x$) :- As obvious, the customer is eligible to get the discount on bundle; $discob_x$ is the discount which provider p_x offers to the customer on the final bill.

QoS Vector:- It contains threshold value of non-price attributes like delay time that customer wants from cloud and fog services. If an attribute is high-value desirable attribute then, threshold value is minimum else if an attribute is low desirable attribute it is maximum.

Straightforwardly, the algorithm could have taken the price of the bundle from providers. But it is more advantageous to consider the price of each VM machine because the proposed algorithm first tries to avail resources from cheap providers before providing the opportunity to other providers. Therefore, quantity to be taken of each VM from each provider may not be equal to quantity offered by a provider in the quotation. Though the algorithm takes into account the probability of data source being local and remote, however, the actual time for which data source will be local or remote can be determined only at the end, i.e., after the customer releases all the consumed resources. If the algorithm takes bundle price from the provider then, only the provider can calculate the final bill and there is no way for the customer to verify the bill. From bundle price, the final bill according to final quantity taken & actual time taken matrices can't be estimated in a combinatorial auction. So, to prevent the customer from getting cheated it is required to have price rates and actual execution time of each VM type.

Broker not only consider provider's quotation but also considers attributes based on past experience. The attributes considered by broker are reputation and priority. Calculation of these attributes is not dependent on whether resource procument is for fog & cloud integrated architecture or for cloud architecture. Therefore, we have calculated reputation and priority in the same way as discussed in [1].

3.3.2. Algorithms of FICAM Model

There are five helper modules (Algorithm 2-6) namely Estimated Bundle Price Algorithm to estimate the bundle price of each provider, Sort Providers Algorithm to sort the providers on basis of attributes; Winner & Quantity Determination Algorithm to find winners and quantity of each VM type to take from each provider; Incentivize Winner Providers to make the model truthful; and Price Settlement algorithm to calculate final bills and impose a penalty on fraud providers. Notations used in algorithms are listed in Table 1. Time complexity of the algorithm is evaluated in Appendix A.

Algorithm 1: FICAM

1: Customer submits requirement to the broker
2: Broker starts auction
▷ Each provider submits quotation to broker
3: **for** each $p_x \in$ P **do**
4: quot_vec$_x$=(lp$_x$, rp$_x$, cp$_x$, qo$_x$, discob$_x$)
5: **end for**
▷ Broker extends Quotation and calculate each provider's estimated bundle price
6: **for** all $p_x \in$ to P **do**
7: extquot_vec$_x$ = (quot_vec$_x$, repu, pri)
8: **end for**
9: ebp = EstimatedBundlePriceCalculator (quot_vec, req)
▷ Broker decides winners
10: (Winners, fqt, point list, score list) = Winner & Quantity Determination (extquot_vec, ebp, req, false)
11: (Incentive)=Incentivize Winning Providers (Winners, extquot_vec, req, point list, score list)
12: Customer use resources
▷ Each winning provider p_x submits final time taken matrices
13: **for** all $p_x \in$ Winners **do**
14: Submit flt$_x$, frt$_x$, fct$_x$
15: **end for**
▷ Broker calculates final bill and apply penalty to cheater providers
16: (fb)=PriceSettlement(Winners, Incentive, fqt, flt, frt, fct, extquot_vec)
17: **Output:** fb

First, inputs are taken from the customer and providers then, broker extends each provider's quotation with attributes like reputation and priority. The broker estimates each provider's estimated bundle price using Estimated Bundle Price Calculator (Algorithm 2).

Estimated Bundle Price Calculator : It calculates total price of the bundle of each provider. It also takes care of local and remote probability and also mobility limit of the data source. For calculating the estimated bundle price of provider p_x , it calculates tlp, trp, tcp. Broker calculates this estimated bundle price so that providers can be compared. $tlp + trp$ is the total price per minute of resources that provider offers on the fog, and tcp is the total price per minute of resources that provider offers on the cloud. Total estimated bundle price of a provider p_x is given by Eq.(1) where $discob_x$ is discount on the bundle which provider is offering.

$$ebp_x = (tlp + trp + tcp) \times \frac{(100 - discob_x)}{100} \quad (1)$$

Broker now determines winners and also determines the quantity of each VM type to be taken from each winner.

Algorithm 2: Estimated Bundle Price Calculator

1: **Inputs:** quot_vec , req
2: **for** all $p_x \in$ P **do**
3: localPriceVec= quot_vec$_x$.lp$_x$
4: remotePriceVec = quot_vec$_x$.rp$_x$
5: cloudPriceVec= quot_vec$_x$.cp$_x$
6: tlp =$\sum_{i=1}^{n}(localPriceVec(i) \times pb_l \times qo^i{}_x \times et^i)$
7: trp =$\sum_{i=1}^{n}(remotePriceVec(i) \times pb_r \times qo^i{}_x \times et^i)$
8: tcp = $\sum_{i=1}^{n}(cloudPriceVec(i) \times qo^i{}_x \times et^i$
9: ebp$_x$=(tlp + trp + tcp) × (100 - discob$_x$)/100
10: **end for**
11: **Output:** ebp

Algorithm 3: Sort Providers

Input: extquot_vec, req, ebp, for_incentive
for all $p_x \in$ P **do**
  M(x) = $\sum_{i=1}^{n} mcf^{i} \times qo^{i}{}_{x}$
  BidDensity(x) = $ebp_x / \sqrt{M(x)}$
  $$score(x) = \sum_{i=1}^{Qos\_len} Qos(i) \times weight\_vec(i) + repu \times weight\_repu + priority \times weight\_pri \quad (2)$$
  $$points(x) = BidDensity(x) \times 1/score(x) \quad (3)$$
end for
Sort providers on basis of points (ascending order)
if for_incentive == false **then**
  Output: point list, score list, sorted_provider
else
  Output: sorted_provider
end if

Winner and Quantity Taken Matrix Determination: For determining winners, first sorting of providers is done. Used sorting technique is same as the one used in baseline model [1] as it uses $\sqrt{M}$ approximation. Lehmann et.al. [8] suggest that using $\sqrt{M}$ approximation changes time complexity from exponential to polynomial. However, in baseline model, the quantity of resources of a VM type taken from provider is the quantity of resources provided by the provider. FICAM overcomes this issue as it considers the quantity of resources of VMs type according to customer's requirement.

Those providers are declared as winners from whom at least one unit of any VM type is taken. The unit of resources of each VM type taken from winner providers is stored in fqt. To begin with, resources are taken from the first provider in the sorted list of providers, if the provider offers desired quantity of all VM types that customer has demanded then the first provider is the sole winner. Else remaining quantity is taken from the second provider in the sorted list of providers, and this continues until requirement gets fulfilled. Stopping condition is satisfied if there is no VM type taken from a service provider p_x, i.e.,
$\sum_{i=1}^{n} qt^{i}{}_{x}$ =0, here, i signifies VM type

Incentivize Winning Providers: This module proposes new incentive scheme for winning providers, to promote them for biding truth value. This incentive mechanism makes FICAM incentive compatible. It is inspired by payment scheme used in [1] i.e., Vickery auction [22]. In Vickery auction, the customer need not pay to the winner according to what winner has bid. Alternatively, the customer pays according to the provider (rw) who wins, if the winner did not take part in the auction. When there are two or more candidates for rw then the runner-up winner is the one who has bid higher.

But, the winner cannot be paid directly the estimated bundle price of rw as estimated bundle price is based on the quantity of VMs offered by providers and not on basis of the final quantity of VMs taken from the provider. Additionally, the winner cannot be directly paid the final price of the runner-up winner because the final price of the runner-up winner is zero (as final quantity taken from rw is zero). Therefore, instead of directly applying Vickery (paying the winner the prices which are bid by rw) as in base model TFMCRA, FICAM gives incentive to the winner based on the quotation of the w and rw.

Algorithm 4: Winner & Quantity Determination

Input: extquot_vec, req, ebp, for_incentive
if for_incentive == false **then**
  sorted_provider, point list, score list = Sort_Providers (extquot_vec, req, ebp, for_incentive)
else
  sorted_provider = Sort_Providers(extquot_vec, req, ebp, for_incentive)
end if
▷ calculate winners and how much quantity of each machine to take from them
for each $p_x \in$ to sorted_provider **do**
  qo_x = quot_vec$_x$.qo_x
  for i=1 to n **do**
    if $ctq^i < qr^i$ **then**
      if $qo^i{}_x >= qr^i$ - ctq^i **then**
        $qt^i{}_x = qr^i$ - ctq^i
      else
        $qt^i{}_x = qo^i{}_x$
      end if
    end if
  end for
  ▷ once all resources in desired quantity have been taken winner determination is complete
  if $\sum_{i=1}^{n} qt^i{}_x$ =0 **then**
    break
  else
    if for_incentive==false **then**
      $fqt_x = qt_x$
    end if
    Winners=Winners ∪ provider
  end if
end for
if for_incentive == false **then**
  Output: Winners , fqt , point list, score list
else
  Output: Winners
end if

Algorithm 5: Incentivize Winning Providers

Input: Winners, extquot_vec, req, point list, score list
for all $p_x \in$ Winners **do**
  M(x) = $\sum_{i=1}^{n} mcf^{i} \times qt^{i}{}_{x}$
  rw=-1
  (Winners′) = WinnerDetermination(req, extquot_vec $_{-x}$, ebp$_{-x}$,true)
  ▷ Here Winners′ vec is traversed from last to first
  for all $p_y \in$ Winners′ **do**
    if $p_y \notin$ Winners **then**
      $rw \leftarrow p_y$
      break
    end if
  end for
  if $rw > 0$ **then**
    Incentive_point(x)= (point(rw) - point(x))
  else
    Incentive_point(x)=0
  end if
  Incentive(x) = Incentive_point(x)× $\sqrt{(M(x) \times score(x)}$
end for
for $p_x \notin$ Winners **do**
  Incentive(x)=0
end for
Output: Incentive

According to the proposed incentive scheme, the incentive of the winner in case of an uni-attribute auction is the price

difference between runner-up winner and winner. Therefore, incentive of the winner becomes as shown in Eq.(4).

$$Incentive(w) = (price(rw) - price(w)) \quad (4)$$

In case of multi-attribute auction(the case here) payment according to Vickery auction means to pay a value to winner according to evaluated value(i.e., point) of runner-up winner considering overall attributes, not just only price. For obtaining that value incentive point is calculated. It is the difference between point of runner-up winner and winner as shown in Eq.(5).

$$Incentive\_point(w) = (point(rw) - point(w)) \quad (5)$$

After calculating incentive point, incentive price needs to be extracted from it. Since, point is calculated by Eq.(3), therefore, price can be extracted from the point using Eq.(6).

$$Incentive(w) = Incentive\_point(w) \times \sqrt{(M(w) \times score(w)} \quad (6)$$

Score is calculated using Eq.(2) and M(w) is calculated by Eq.(7)

$$M(w) = \sum_{i=1}^{n} mcf^i \times fqt_w^i \quad (7)$$

Algorithm 6: Price Settlement

Inputs: Winners, Incentive, fqt, flt , frt, fct, extquote_vec
Customer tells broker delivered Qos of winning providers i
Broker recalculates score (final_score) of winning providers
for each $p_x \in$ Winners **do**
  for i=1 to n **do**
    bill += ($lp^i_x \times flt^i_x + rp^i_x \times frt^i_x + cp^i_x \times fct^i_x$) $\times$ fqt^i_x
  end for
  ▷ if customer has taken at least 50% resources of what provider has offered then only customer gets discount on bundle and that too in proportion
  ratio= $\frac{\sum_{i=1}^{n} fqt^i_x}{\sum_{i=1}^{n} qo^i_x}$
  if ratio= >0.5 **then**
    bill = bill $\times$ (100- ($disco_x \times$ ratio))/100
  end if
  if final_score (x) is equal to score(x) **then**
    fb_x = bill + Incentive(x)
  else
    fb_x = bill $\times$ final_score / initial_score
  end if
end for
Output: fb

Price Settlement Algorithm In this algorithm, for each winning provider final bill vector is generated. Customer reports broker about the delivered quality. High value of QoS is desired thus, if delivered QoS is higher than proposed QoS then no problem; but if it is less than proposed QoS then, provider cheated the customer. Considering this logic the score is recalculated. If the recalculated score is equal to initial score then, an incentive is given to provider, but if provider cheats then no incentive is given to provider. However, if provider cheats, then as a compensation to the customer and as a penalty to the provider final bill is in proportion of $\frac{final\_score}{initial\_score}$. The reason is, if provider cheats then final_score is less than *initial_score*.

Table 2: Comparitative Study of Various State of Arts

Features	Reverse Auctioning Model				
	FICAM	TFMCRA	Prasad	C-DISC	Modica
Combinatorial Auction	✓	✓	✓	✗	✓
Multi-attribute Auction	✓	✓	✓	✓	✗
Budget Balanced	✓	✓	✓	✓	-
Individual Rational	✓	✓	✓	✓	✓
Robust	✓	✓	✗	✗	✗
Bidder's Optimality	✗	✗	✓	✗	✓
Non-Dominant	✓	✓	-	✗	✓
Incentive Compatibile	✓	✓	✗	✓	✗
Egalitarian Social Welfare	✓	✓	✗	✗	✗
Cloud Architecture Applicable	✓	✓	✓	✓	✓
Fog Integrated Cloud Architecture Applicable	✓	✗	✗	✗	✗
Satisfy Mobile Data Source / IOT Need	✓	✗	✗	✗	✗

TFMCRA - [1] Prasad - [16] C-DISC - [15] Modica - [11]

Here, *initial_ score* is the score calculated at the time of sorting providers and *final_score* is the score calculated after the customer reports delivered QoS.

Bill is calculated based on initial local, remote, cloud price matrices; final quantity taken matrix; and final local, remote, cloud time taken matrices. Before providing the discount, the ratio of quantity is calculated using Eq.(8) which is the ratio of quantity taken from and quantity offered by the provider. If $ratio > 0.5$, based on the ratio the discount is offered; otherwise, the discount is not rewarded.

$$\text{ratio} = \frac{\sum_{i=1}^{n} fqt^i_x}{\sum_{i=1}^{n} qo^i_x} \quad (8)$$

FICAM satisfy certain auction properties like incentive compatible/truthful, budget balanced, individually rational, etc. Proof of properties, i.e., truthfulness, monotone, non-dominant, robust, egalitarian social welfare(fairness), budget balanced, individually rational is given in Appendix B. We compared FICAM with other models, and a summary of comparison is given in Table 2. Comparision of FICAM with other reverse auctioning models is based on reverse auctioning properties like whether model supports combinatorial auction, multi-attribute auction, model is budget balanced, individual rational, robust, optimal with respect to bidders, non-dominant, incentive compatible, Egalitarian socially welfare, cloud architecture applicable, fog and integrated cloud architecture applicable and can it satisfy mobile data source and IOT needs.

4. Performance Evaluation

FICAM behavior is evaluated by simulation under various scenarios. As obvious, developing real environment is very costly and difficult due to several external entities like service providers, brokers, and consumers, etc. Therefore, to verify research outcomes, simulation is a better choice. Fog computing introduces its own emerging challenges, i.e., fog integrated

cloud architecture, dynamic pricing model, location wise fog services; therefore, simulators like CloudAuction[19], Cloudsim [4] for cloud computing environment cannot be used for the proposed work. Simulation experiments are designed in Matlab by following the same approach as in [1]. According to [1], no simulator is available which takes into consideration the requirements of multi-attributed reverse auction considering emerging fog integrated cloud architecture. Only one baseline model is considered because according to [1] TFMCRA is the only truthful resource allocation model in cloud computing with multi-attribute combinatorial reverse auction mechanism.

For simulating FICAM in Matlab, dataset for 400 providers is generated. For each provider p_x a random number between 0 and 15 is generated. p_x will provide VM Ids up to this random number on fog and after this random number, on the cloud. This random number is generated so that different providers offer a different number of VM Ids on fog and cloud. For generating final local, remote, cloud time matrices following pseudocode is used:

```
if lm = local then
  for each VM Id i
    if i is on fog then
      // final remote time and cloud time will be zero
      flt^i_x = et^i
    else
      // final local time and remote time will be zero
      fct^i_x = et^i
    end
  end
else
  for each VM Id i
    if i is on fog then
      // final cloud time will be zero
      flt^i_x = et^i * p_l
      frt^i_x = et^i * p_r
    else
      // final local time and remote time will be zero
      fct^i_x = et^i
    end
  end
end
```

where p_x is any random winner provider

4.1. Dataset

In order to simulate the proposed model and to analyze its behavior under various scenarios, a dataset is constructed from various publically available sources [5]. However, to generate the dataset, standard pricing schemes of authenticated services providers are followed [5]. The dataset contains those providers who are providing both fog and cloud resources or those fog providers who have tied up with cloud providers and vice versa. However, only fog and only cloud providers can participate as long as they can provide each type of VM configuration machine as required by the customer.

Thirty different types of VMs are taken into consideration [5], and the configuration of all 30 types of VMs is briefed in Table 3. It is assumed that very high configuration VMs (indexed from 15 to 30 in Table 3) are provided only on cloud whereas small configurations can be provided on cloud or fog by different providers. In the dataset, different types of VMs are indexed from 1 to 30 in increasing pricing order based on their configuration. As an example, 1 refers to VM with RAM: - 512 MB, Storage 1GB, CPU Power: - 1x. We used $/minute price model of VMs as suggested in [13]. Dataset is constructed for 400 providers. Cloud price of each provider's VM Id 1 is randomly taken between [0.00748 - 0.0150]. Cloud price of other 29 VMs say VM Id v_x is randomly taken between [1.1125 - 1.2375] $\times$ (v_{x-1}). These VM Ids and their cloud price are generated corresponding to various service providers and their standard pricing rates from the publically available sources [5]. Technically, because of the limited number of fog resources and high resource demand [27], it is observed that fog prices are dynamic based on the customer requirement and load at the fog providers site. In experiments, the fluctuation in fog and cloud prices are considered from -15% to 15%. Limit of mobility is local, state and national. 33% difference is taken between local fog prices & state price and 25% difference is taken between state & national fog prices are taken into account by following the current telephone utility service schemes from telephone company rates [17].

Table 3: VM Types and Configuration Attributes

VM ID	RAM	Disk	CPU	VM ID	RAM	Disk	CPU
1	512MB	1GB	1	16	4GB	100GB	5
2	1GB	5GB	1	17	4GB	200GB	6
3	1.5GB	10 GB	1	18	8GB	50GB	6
4	2GB	20GB	1	19	8GB	1TB	5
5	512MB	5GB	2	20	4GB	500GB	6
6	1GB	10GB	2	21	2GB	500GB	5
7	2GB	50GB	2	22	16GB	200GB	4
8	1GB	20GB	3	23	4GB	1TB	4
9	1.5GB	50GB	3	24	16GB	200GB	8
10	4GB	50GB	2	25	8GB	500GB	6
11	4GB	10GB	3	26	32GB	500GB	6
12	2GB	100GB	3	27	16GB	1TB	6
13	2GB	200GB	3	28	32GB	1TB	7
14	8GB	200GB	1	29	32GB	500 GB	8
15	8GB	100GB	4	30	64GB	1TB	8

4.2. TFMCRA as Baseline Model

We used TFMCRA as a baseline model and resource allocation in TFMCRA [1] is as follows: TFMCRA sorts providers on the basis of price and non-price attributes. Sorted provider list is traversed until constraint in Eq.(9)

$$\sum_{x=1}^{Winners.len} \sum_{i=1}^{n} q^{i}{}_{\text{Winners(x)}} >= q_c \tag{9}$$

is unsatisfied. All providers traversed before and the Winners(x) (Winners(x) is p_x where constraint in Eq.(9) is satisfied) are declared as winners. Winner providers resources are allocated to the customer.

Pricing in TFMCRA is as follows: In TFMCRA, providers bid bundle price. Let, p_X is a winner in TFMCRA and has bid

Table 4: Common Simulation Parameters

Types of VM(VM Id's)	1,3,4,6,7,9,12,15,17,19,22,23,25,28,30
Quantity of VM type	5,6,7,4,3,10,11,12,13,50,60,70,30,35,40
Time for VM type	12,14,16,17,18,19,20,40,60,120,90,50,100,70,80
Quality of Service	1.0
Weight Vector	0.25,0.25,0.25,0.25
Response Time Vector	0.3,0.35,0.4,0.5,0.6,0.7,0.9,1,1.5,2,2.5,3,3.5,4,5
Iterations	600
lm	local,state,national
Local Probability	0.5
Remote Probability	0.5
Reputation Factor	2
Decrease Factor	10

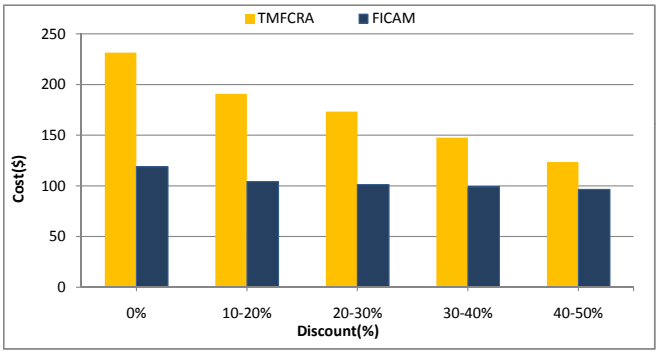


Figure 3: Comparative Study of Resource Procurement Cost of FICAM & TFMCRA on Varied Discount

price$_X$. TFMCRA uses Vickery auction pricing scheme. After application of Vickery pricing scheme price$_X$ becomes the price of provider p'_x who would have been the winner if p_x won't have participated. The final price vector is calculated after the customer uses resources and submits QoS. Broker imposes a penalty on the provider if provider cheats with the customer.

4.3. Results and Discussion

The FICAM is evaluated on 8 different scenarios. The experiments are run 600 times, i.e., 200 times with *lm*= local, 200 times with state and 200 times with national. Results of these 600 iterations are averaged so that the effect of *lm* and non-price attributes can be observed in results. In order to compare the results with baseline, same simulation parameters like quantity, type, prices, etc. are used. Common simulation parameters (as default parameters) are listed in Table 4, and the configurations of used VM type is listed in Table 3. Scenarios 1 to 5 are listed here, and scenarios 6 to 8, i.e., Effect of Imposition of Penalty on Cheater Providers, Effect of Priority Attribute on Providers, and Effect of inclusion of Reputation on Providers are listed in Appendix C.

4.3.1. Scenario1 : Resource Procurement Cost vs Discount

The customer receives some discount when the customer acquires a bundle of resources. The effect of variation of discount on resource procurement cost of FICAM and TFMCRA is analyzed. In this scenario, the discount is varied from 0% to 50% on 50 providers. Simulation parameters are same as in Table 4. From the Figure 3, following observations are made:

I. FICAM resource procurement cost is less than TFMCRA. This is because FICAM satisfies constraint in Eq.(10) whereas TFMCRA satisfies constraints in Eq.(9).

$$\sum_{x=1}^{Winners.len} \sum_{i=1}^{n} q^{i}_{\text{Winners(x)}} = q_c \tag{10}$$

According to constraints (shown in Eqs. (9) and (10)), FICAM considers only that much quantity as required by the customer, but TFMCRA considers quantity equals to or more than the required by the customer.

II. Markedly, it can also be observed that as discount increases, resource procurement cost of TFMCRA decreases at a more elevated rate than FICAM. The reason is that using FICAM, the customer is able to enjoy discount only if he takes at least 50% of resources offered in provider's quotation. Additionally, the discount is in the ratio of $\frac{qt}{qo}$ due to which discount ratio decreases; hence, the number of winner providers offering a discount to the customer also decreases.

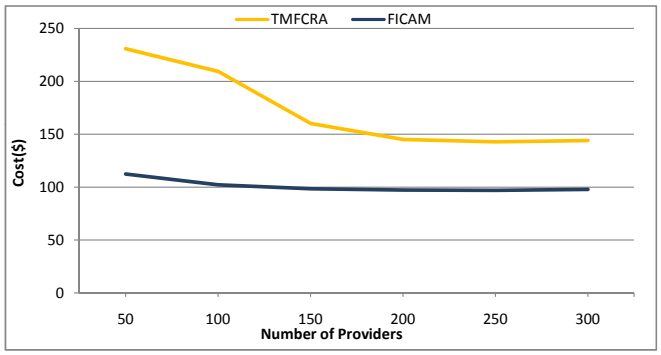


Figure 4: Comparative Cost of Resource Procurement Cost of FICAM & TFMCRA on Varied Number of Providers

4.3.2. Scenario 2: Resource Procurement Cost vs Number of Providers

In this scenario, the behavior of FICAM(procurement cost) is tested on the number of providers. The algorithms are run 600 times (same as in scenario 1 experiments), and results are averaged. Simulation parameters except VM IDs [2,3,5,7,9,11,13, 15,18,20,21, 23,26,28,29] are same so that each VM type in the dataset can be utilized. From results as shown in Figure 4, it is observed that if the number of providers is increased then, it affects resource procurement cost of both FICAM and TFMCRA. However, the cost of FICAM is better in comparison to TFMCRA. It is also observed that the resource procurement cost decreases as the number of providers increases and becomes constant. This is because, as the number of providers increases, the competition among providers increases and cost decreases. However, resource procurement cost of FICAM is lesser than TFMCRA because of the same reason as in scenario 1.

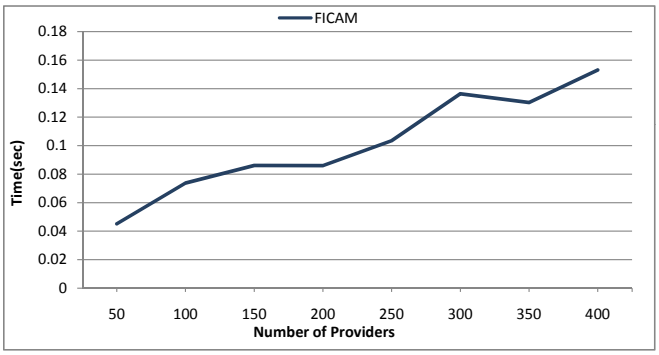


Figure 5: Computation Time of FICAM on Varied Number of Providers

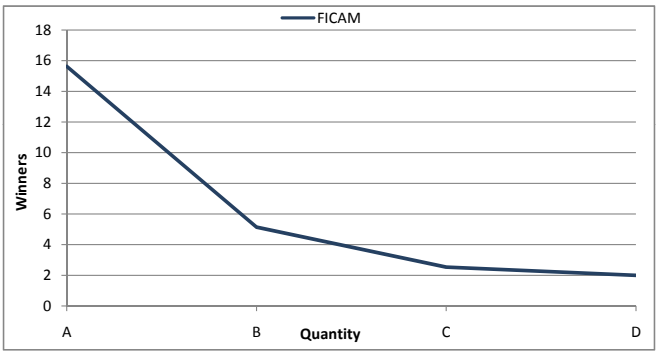


Figure 7: Number of Winners of FICAM on Varied Quantity of resources

4.3.3. Scenario 3: Computation Time vs Number of Providers

In scenario 3, the effect of the varied number of providers on computation time is evaluated. The simulation parameters of this experiment setup are same as in Table 4 except VM Ids [1,2,4,6,7,9,12,14,16,19,22,25,27,28,30]. The machine used to execute this set of experiments is core i5, 4 GB RAM, 500 GB disk. It is straightforward to infer from the output (Figure 5) that when the number of providers increases then, computation time also increases. Figure 5 shows that even when the number of providers is 400 even then, computation time is less than one second. Nevertheless, we can conclude from time complexity of FICAM discussed in Appendix A, that the computation time of the proposed algorithm is polynomial. Therefore, FICAM also fits the computation time requirements.

4.3.4. Scenario 4: Resource Procurement Cost and Number of Winners vs Quantity

This scenario is dedicated to evaluate the behavior of the FICAM in terms of resource procurement cost and the number of winners, when resource offered quantity by providers is varied. Resource quantity offered by providers is increased by an amount, and its effect is taken into account. Simulation parameters are same as in Table 4 expect VM Ids considered here are [2,5,8,10,11,13,14,16,18,20,21,24,26, 27,29]. In this scenario, 50 providers are accounted for bidding. Since fog resources are limited and their quantity is usually less than cloud quantity[3], therefore, different quantity increment is considered for fog and cloud resources. In the first case, every provider is providing every fog resource in the range from 0 to 5 and every cloud resource in the range from 0 to 10. In the second case, the quantity of every fog resource is between 5 and10 and every cloud resource is between 10 and 20, i.e., an increment of 5 is considered for fog resources and 10 for cloud resources. Similar is the third and the fourth case. Results are shown in Figures 6 and 7. On quantity X-axis, 4 points are shown i.e., A, B, C, D, where A point corresponds to first case i.e., where fog resource range is [0-5] and cloud resource range is [0-10], B corresponds to the second case and so on. From the results, as in Figures 6 and 7, it can be observed that when the quantity of each VM type provided by the provider is increased, FICAM's resource procurement cost and average number of providers decreases. The reason is that the lowered price providers are now giving more resources at the same rate as before, thus, FICAM acquires fewer resources from high-cost providers resulting in a decrease of cost and the number of winners.

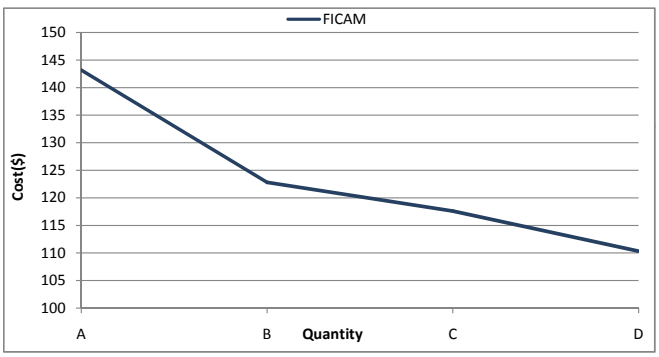
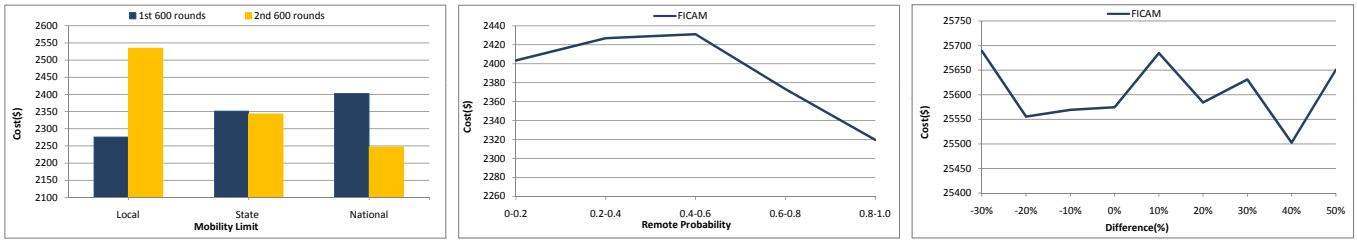


Figure 6: Resource Procurement Cost of FICAM on Varied Quantity of resources

4.3.5. Scenario 5: Effect of Limit_Mobility, Probability, Dynamic Fog Prices on Resource Procurement Cost

a) **Resource Procurement Cost vs Limit_Mobility**: In this, FICAM behavior is observed on the limit of data source mobility. Mobility limit can be local, state, national. When mobility limit changes then remote prices of VM changes and hence fog price changes. To visualize change when mobility limit changes, the share of fog price in total amount should be considerable. The share of fog price in total amount according to common simulation parameters was coming out less. Therefore, testing parameters are the same as in Table 4 except quantity vector [25,30,24,27,21,26,23 ,32,35, 37,41,39,42,38,33] and expected time for which each VM acquired is 10 days. The experiment is conducted twice; each time consisting of 600 rounds of which 200 rounds are with the local limit, 200 rounds with the state limit, and 200 rounds with the national limit. From a run of first and second 600 rounds resource procurement cost with local, state, and national mobility are shown in Figure 8 (a) as 1st and 2nd 600 rounds; where local cost is the average cost of 200 rounds with *lm* = 'local'. State cost and national cost are similarly defined.

b) **Resource Procurement Cost vs Remote Probability**: This scenario estimates the effect of change of data source remote probability on resource procurement cost. Simulation parameters are same as in scenario 5(a) and each time algorithm is run for 600 rounds - 200 for local, 200 for state and 200 for national. The average cost of 600 rounds is taken into account. Remote probability is varied from 0 to 1 to observe the behavior



(a) Cost vs Limit_Mobility (b) Cost vs Remote Probability (c) Cost vs Dynamic Fog Price

Figure 8: Resource Procurement Cost of FICAM on Varied Limit_Mobility, Probality, Dynamic Fog Price

of FICAM. The results are shown in Figure 8 (b).

c) **Effect of Dynamic Fog Price Change on Resource Procurement Cost**: As load changes on fog resources their prices are varied (increase or decrease) due to dynamic pricing[27]. This scenario estimates the effect of dynamic fog pricing on total resource procurement cost. For this set of experiments, the percentage difference between local fog price and cloud price of each VM of each provider is varied from -30% to 50%. Other parameters are same as in scenario 5(a). Each experiment is conducted 600 times as in scenario 5(b). The effect of the cost of dynamic fog pricing is shown in Figure 8 (c).

From the results in Figure 8 (a), (b) and (c) it is inferred that the model's working behavior is dynamic. Providers in FICAM are sorted on basis of total estimated bundle cost. Total estimated bundle cost of a provider is given by Eq.(1). Whenever *lm* or remote probability or fog prices increase, *tlp* and *trp* is multiplied by some factor and hence, increases but, *tcp* remains constant. Total estimated bundle price increases and this changes the sorted list. From the theorem 1, it can be proved that place of providers in this new sorted list may not be same as in previous sorted list(i.e., list before *lm* or remote probability or fog prices change). Hence, winner changes and this changes resource procurement cost. Resource procurement cost can increase or decrease, i.e., no relation can be established between resource procurement cost and mobility limit or remote probability or dynamic fog prices. An example can demonstrate this phenomenon.

Suppose there are six providers a, b, c, d, e, f; the sorted list of providers before the increase is [a, b, c, d, e, f] and winners are [a, b, c, d]. After increase let, the sorted list of providers is [a, b, c, e, f, d] and winners are [a, b, c, e]. Total resource procurement cost before the increase is given by Eq.(11).

$$fb_a + fb_b + fb_c + fb_d \tag{11}$$

After increase, resource procurement cost is given by Eqn. (12)

$$fb'_a + fb'_b + fb'_c + fb'_e \tag{12}$$

The final bill of provider *a* after the increase may be greater than or equal to before the increase (equal to in case when the provider is offering cloud resources only) i.e. $fb'_a >= fb_a$. This is valid for providers *b and c* as well. Hence, we can say

$$fb_a + fb_b + fb_c <= fb'_a + fb'_b + fb'_c \tag{13}$$

But $fb'_e >= fb_d$ cannot be directly said for providers d and e because providers are sorted not only by price, but on basis of non- price attributes. fb'_e can be less than fb_d. Hence, total resource procurement cost before the increase can be less than total resource procurement after the increase, i.e., Eq.(14) or can be greater than after the increase, i.e., Eq.(15).

$$fb_a + fb_b + fb_c + fb_d >= fb'_a + fb'_b + fb'_c + fb'_e \tag{14}$$

$$fb_a + fb_b + fb_c + fb_d <= fb'_a + fb'_b + fb'_c + fb'_e \tag{15}$$

Theorem1: If *tlp* is increased by factor a and *trp* is increased by factor b then, place of providers in new sorted list changes.

Proof: Let, p_x & p_y be two providers and before price increases place of p_x appears before p_y in sorted list of providers. Therefore,

$$tlp_x + trp_x + tcp_x < tlp_y + trp_y + tcp_y \tag{16}$$

When *tlp* increases by factor d1 and *trp* increases by a factor d2 then, sorted list changes. It is indeterminate that place of provider p_x will remain before p_y in the new sorted list of providers because Eq.(17) can't be derived from Eq.(16).

$$tlp_x \times d1 + trp_x \times d2 + tcp_x < tlp_y \times d1 + trp_y \times d2 + tcp_y \tag{17}$$

Hence, the place of providers in new sorted list changes.

5. Conclusions and Future Work

In this paper, a novel and multi-attribute mechanism for resource procurement utilizing fog integrated cloud architecture (FICAM) is proposed. Combinatorial reverse auction approach is chosen for resource procurement as it has more benefits than its counterparts. Considering fog and cloud service providers along with their respective attributes, an architecture model is also proposed along with pricing design and resource procurement algorithm. Pricing design is based on local, remote fog resources and cloud resources. The FICAM algorithm takes care of data source mobility and limited fog resource issues in fog computing. It estimates the bundle price by each provider on the basis of customer's local and remote probability, limit

of mobility; determine winners according to estimated bundle price and other attributes. As a fact, fog doesn't have unlimited resources like cloud; therefore, the algorithm also determines and takes into account quantity of resources to be acquired from each winner. An incentive scheme for winner providers is also proposed. The algorithm calculates final bill based on local, remote, cloud time matrices and quality of service delivered. For simulation, we prepared our own dataset because of unavailability of a dataset consisting of fog and cloud reverse auction prices addressing the data source mobility issue in fog computing. Eight different scenarios are explored for examining the behavior of FICAM considering various consequences. Results are compared with the base auctioning model based on cloud architecture. From the results, FICAM works effectively in real-time and highly dynamic scenarios; it can be a candidate for auctioning in future as meeting the requirements of customer and service providers of fog integrated cloud computing architecture.

In future, machine learning based techniques can be utilized to make FICAM more truthful, realistic, memorable auctioning mechanism by predicting the behavior of the service providers based on past experiences.

References

[1] G. Baranwal and D.P. Vidyarthi. A truthful and fair multi-attribute combinatorial reverse auction for resource procurement in cloud computing. *IEEE Transactions on Services Computing*, pages 77–97, 2016.

[2] L. F. Bittencourt, J. D., R. Buyya, O. F. Rana, and M. Parashar. Mobility-aware application scheduling in fog computing. *IEEE Cloud Computing*, pages 26 – 35, 2017.

[3] Fog computing and the internet of things: Extend the cloud to where the things are. `https://www.cisco.com/c/dam/en_us/solutions/trends/iot/docs/computing-overview.pdf`, 2015.

[4] R.N. Calheiros, R. Ranjan, A. Beloglazov, C.A. De Rose, and R.Buyya. Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Software: Practice and Experience*, pages 23 – 50, 2011.

[5] `https://www.cloudorado.com/`.

[6] A.V. Dastjerdi and R. Buyya. Fog computing: Helping the internet of things realize its potential. *IEEE Computer,IEEE CS Press, USA*, pages 40–44, 2016.

[7] J. S. Lee and B. K. Szymanski. A novel auction mechanism for selling time-sensitive e-services, e-commerce technology. *Seventh IEEE International Conference on IEEE*, pages 75 – 82, 2005.

[8] D. Lehmann, L.I. Ocallaghan, and Y. Shoham. Truth revelation in approximately efficient combinatorial auctions. *Journal of the ACM(JACM)*, pages 577 – 602, 2002.

[9] T. H. Luan, L. Gao, Z. Li, Y. Xian, G. We, and L. Sun. Fog computing: Focusing on mobile users at the edge. *Engineering with Computers*, pages 1 – 11, 2016.

[10] G. Manoochehri and C.Lindsy. Reverse auction: Benefits, challenges, best practices. *California Journal of Operations Management*, pages 123 – 130, 2008.

[11] G.D. Modica, G. Petralia, and O. Tomarchio. Procurement auctions to trade computing capacity in the cloud. *P2P, Parallel, Grid, Cloud and Internet Computing(3PGCIC), 2013 Eighth International Conference on, IEEE*, pages 298 – 305, 2013.

[12] Nisan, N., Ronen, and A. Algorithmic mechanism design. *Proceeding STOC '99 Proceedings of the thirty-first annual ACM symposium on Theory of computing*, pages 129 – 140, 1999.

[13] `http://omerio.com/2016/03/16/saving-hundreds-of-hours-with-google-compute-engine-per-minute-billing/`.

[14] A. Pla, B. Lopez, and J. Murillo. Multi-attribute auctions with different types of attributes: Enacting properties in multi-attribute auctions. *Expert Systems with Applicaions*, pages 4829 – 4843, 2014.

[15] A.S. Prasad and S. Rao. A mechanism design approach to resource procurement in cloud computing. *Computers, IEEE Transactions*, pages 17 – 30, 2014.

[16] G.V. Prasad and S. Rao. A combinatorial auction mechanism for multiple resource procurement in cloud computing. *Intelligent Syaytems Design and Applications(ISDA), 2012 12th Inter-national Conference on IEEE 2012*, pages 337 – 344, 2012.

[17] `http://www.rcom.co.in/Rcom/personal/home-phone/landline-phone.html`.

[18] `https://en.wikipedia.org/wiki/Roaming/`.

[19] P. Samimi, Y. Teimouri, and M. Mukhtar. A combinatorial double auction resource allocation model for cloud market. *Information Sciences*, 2014.

[20] T Schoenherr and V. A. Mabert. Online reverse auction: common myths vs evolving reality. *Business horizons*, pages 373 – 384, 2007.

[21] B. Song, M.M. Hassan, and E.N. Huh. A novel cloud market infrastructure for trading service. *Computational Science and Its Applications,2009,ICCSA'09, International Conference on, IEEE*, pages 44 – 50, 2009.

[22] `https://en.wikipedia.org/wiki/Vickrey_auction`.

[23] S.D. Vries and R.V. Vohra. Combinatorial auctions: A survey. *INFORMS Journal on computing*, pages 284 – 309, 2003.

[24] Shanhe Yi, Zijiang Hao, Zhengrui Qin, and Qun Li. Fog computing: Platform and applications. *Third IEEE Workshop on Hot Topics in Web Systems and Technologies (HotWeb)*, pages 73 – 78, 2015.

[25] Liang Yu, Tao Jiang, and Yulong Zou. Fog-assisted operational cost reduction for cloud data centers. *IEEE Access*, pages 13578 – 13586, 2017.

[26] S. Zaman and D.Grosu. Combinatorial auction-based allocation of virtual machine instances in clouds. *Journal of Parallel and Distributed Computing*, pages 495 – 508, 2013.

[27] H. Zhang, Y. Xiao, S. Bu, D. Niyato, R. Yu, and Z. Han. Computing resource allocation in three-tier iot fog networks: a joint optimization approach combining stackelberg game and matching. *IEEE Internet of Things Journal*, pages 1204 – 1215, 2017.

Anubha Aggrawal received B.Tech. degree in Computer Science & Engineering from Shri Mata Vaishno Devi University, Katra, J&K, India in 2017. Currently, she is working TechAspect as software engineer. Her research interests include resource provisioning and management in Fog Computing, Cloud Computing, Question Answering Systems and Word Alignment.

Neetesh Kumar did his PhD on the topic of Designing Scheduling Models for Multi-Core Many-Core Machines in the School of Computer and Systems Sciences, Jawaharlal Nehru University, New Delhi, India. Currently, he is working as Assistant Professor in ABV Indian Institute of Information Technology and Management, Gwalior, India. His research interests include parallel and distributed computing, high performance computing, soft computing etc.

Deo Prakash Vidyarthi is working as Professor in the School of Computer and Systems Sciences, Jawaharlal Nehru University, New Delhi. Dr. Vidyarthi has published around 90 research papers in various International Journals and Transactions (including IEEE, Elsevier, Springer, Wiley, World Scientific etc.) and around 45 research papers in proceedings of various peer-reviewed conferences in India and abroad. He has contributed chapters in many edited books. He is in the editorial board of many International Journals and also in the reviewers panel of many International Journals. Dr. Vidyarthi has co-authored a book (research monograph) entitled Scheduling in Distributed Computing Systems: Design, Analysis and Models published by Springer, USA released in 2009. Another book (edited) by Dr. Vidyarthi is Technologies and Protocols for the Future Internet Design: Reinventing the Web, by IGI-Global (USA) released in the year 2012. Dr. Vidyarthi is the senior member of the IEEE, International Society of Research in Science and Technology (ISRST), USA, International Association of Computer Science and Information Technology (IACSIT), Singapore and International Association of Engineers (IAENG). His research interest includes Parallel and Distributed System, Grid and Cloud Computing, Mobile Computing, Evolutionary Computing etc. Email- dpv@mail.jnu.ac.in, dpvidyarthi2002@yahoo.com

Rajkumar Buyya is a Redmond Barry Distinguished Professor and Director of the Cloud Computing and Distributed Systems (CLOUDS) Laboratory at the University of Melbourne, Australia. He served as a Future Fellow of the Australian Research Council during 2012-2016. He has authored over 625 publications and seven text books. He is one of the highly cited authors in computer science and software engineering worldwide. Dr. Buyya has led the establishment and development of key community activities, including serving as foundation Chair of the IEEE Technical Committee on Scalable Computing and five IEEE/ACM conferences. He served as the founding Editor-in-Chief of the IEEE Transactions on Cloud Computing. He is currently serving as Co-Editor-in-Chief of Journal of Software: Practice and Experience. Dr. Buyya is recognized as a 2016 Web of Science Highly Cited Researcher by Thomson Reuters, a Scopus Researcher of the Year 2017 with Excellence in Innovative Research Award by Elsevier, and a Fellow of IEEE for his outstanding contributions to Cloud computing.