Personal and Ubiquitous Computing (2023) 27:2225–2241
https://doi.org/10.1007/s00779-023-01765-6

ORIGINAL ARTICLE

Setting up SLAs using a dynamic pricing model and behavior analytics in business and marketing strategies in cloud computing

Ehsan Gorjian Mehlabani[1] · Amir Javadpour[2,3] · Chongqi Zhang[1] · Forough Ja'fari[4] · Arun Kumar Sangaiah[5,6]

Received: 3 August 2022 / Accepted: 27 August 2023 / Published online: 19 September 2023

Abstract

Increasing amounts of data are being generated every year. Sustainable computing systems have become capable of extracting and learning information from the underlying data. Edge and AI (artificial intelligence) are expanding into industrial systems requiring new computing and networking infrastructure. Due to this, SLA computing is becoming increasingly challenging to handle in these emerging cloud environments. The cloud is a service that provides virtual resources to users. Qualitative and quantitative findings in market-oriented approaches are one of the most common methods for managing virtual and physical machines in a network. When allocating services, price is an important factor to consider. In this study, we aim to determine the initial price of VMs while considering the dynamic pricing model in a competitive, sustainable computing system. Besides negotiation-based trading, a multifactor architecture is used for trading in the marketplace. Based on the simulation results, it was found that the performance could be improved by categorizing the VMs based on regression. According to the simulation results, the cloud market system provides a better service-level agreement (SLA) and response time when assigning virtual machines to the market. Based on the results, we found that using the regression method for categorizing the VMs to manage the market improved the SLA.

Keywords VMs · Service-level agreement · Multi-factor architecture · Regression · Sustainable computing, Green sustainable economy, Economy computing

1 Introduction

Sustainability computing and cloud computing are models that enable users to access configurable resources, such as networks, servers, storage spaces, and web-based applications according to their needs and the SLAs they have with their service providers. Process resources can be provided by minimal management effort or minimal interaction with service providers. Virtualization is a valuable technology in cloud computing. It maximizes processing power through virtualization. Cloud computing services include the following three types: cloud platform as a service, cloud infrastructure as a service, and cloud software as a service [1, 2]. Cost-saving, scalability, easy management, etc., are among the main advantages of cloud computing, which enable service providers and different companies to use this model [3–5].

A cloud infrastructure provider offers their users virtual machines (VMs) as service resources. The management of resources for cloud infrastructure providers is crucial. A resource management process involves resource allocation, requirement mapping, trading, estimation, and modeling. It

✉ Amir Javadpour
a.javadpour87@gmail.com

Arun Kumar Sangaiah
aksangaiah@ieee.org

1 Department of Probability and Statistics, Guangzhou University, Guangzhou 510006, China

2 Department of Computer Science and Technology (Cyberspace Security), Harbin Institute of Technology, Shenzhen, China

3 ADiT-Lab, Electrotechnics and Telecommunications Department, Instituto Politécnico de Viana do Castelo, 4900-347 Porto, Portugal

4 Department of Computer Engineering, Sharif University of Technology, Tehran, Iran

5 International Graduate School of Artificial Intelligence, National Yunlin University of Science and Technology, Yunlin 64002, Taiwan

6 Department of Electrical and Computer Engineering, Lebanese American University, Byblos 13-5053, Lebanon

is crucial that a cloud provider implements an efficient allocation technique that reflects the market conditions. This is the process through which a client determines how much they paid for the service received, and the provider determines how much they received for the service delivered. This is a fundamental issue in resource allocation. Cloud-marketplace is an online storefront that operates by the provider of cloud service. Cloud-marketplace enables the providers to supply their products as well as enables the clients to raise their requirements. Among the advantages of the cloud, the marketplace is controlling and simplifying the buy and sell processes for both parties; besides, the exchange in the market can help the price control and prevents the market to become a monopoly [6–8]. AWS Marketplace, Oracle Marketplace, Marketplace Azure Windows Microsoft, and Salesforce AppExchange are examples of cloud-marketing [9–11]. In finding out the price at which a resource will initially enter the market, there is an important issue and challenge involved. Initial pricing is a part of determining the price at which a resource will initially enter the market. Initial pricing can be significantly affected by the choice of pricing method and model, for example, if the auction method is used, the initial price would be equal to the final bid price of the last auction [8, 12–14]. Studying the initial prices of cloud resources or services is the purpose of this paper. In addition to creating the initial pricing, we have focused on the overall pricing. Along with the fixed prices, the market conditions should also be considered when creating an initial price for a service. Therefore, to create the initial pricing, along with the parameters and factors that lead to the creation of fixed costs, the variables and parameters that lead to the creation of variable costs should also be determined. When determining the price, we will consider all parameters and factors. It is possible that an agreement will not be reached between the buyer and seller if the initial price and the negotiated price are significantly different. A closer distance between the initial and negotiated prices reduces productivity and, therefore, the starting point of the initial price and the negotiated price should not be too far or too close. Thus, determining an initial price that is appropriate for the market will result in higher convergence rates and eventually more productivity. Furthermore, pricing has a direct impact on the profit of the traders, so the better the pricing, the greater the profit both parties realize. This study aims to develop an initial price based on different factors and conditions, which will result in a suitable agreement that will benefit both parties. The following are the objectives of the research [15]:

- Obtaining the effective factors in determining the initial pricing of the VM type and checking the response time.
- Finding the weight and the impact of each effective factor in the initial pricing of the VM type.
- Combining the extracted factors properly and investigating the SL.

A novel dynamic initial pricing method has been presented in this study to set up a service-level agreement based on qualitative and quantitative findings. Managing cloud services according to the market-oriented approach is one of the most common methods. We are trying to construct the VMs' (VM (virtual machine): A virtual machine is an emulation or virtualization of a computer system. Virtual machines resemble physical computers in their architecture and provide the functionality of one. Specialized hardware or software may be used in their implementation SLA) initial price by considering market dynamics and regression. Statistical modeling, also known as regression analysis, is a method of estimating relationships between a dependent variable and independent variables in cloud computing, which we have used. A linear regression analysis is the most common form of regression analysis in this field, which identifies the line (or a complex combination of lines) that most closely fits the data according to a mathematical condition. Negotiations in the market are based on negotiation-based architecture, where multifactor architecture is used for the negotiations.

The research is notable for its contributions to:

- Dynamic initial pricing of VMs as a means of setting SLAs in a competitive market.
- Using a market-oriented approach to negotiating market deals.
- Improved SLAs (SLAs (service-level agreements) are agreements between cloud service providers and their customers that ensure minimum levels of service are maintained) are indicated by categorizing VMs using regression and using regression to categorize VMs.

2 Definitions and research background

The aim of negotiation of a cloud SLA (service-level agreement) is a common decision-making process between the cloud service providers and cloud server clients to solve their conflicting goals. If the negotiation between the provider and clients is done manually, it can become a bottleneck [16–19]. In [4], this issue has been studied as, if the parameters, such as deadline and the initial offer value, have changed, what is the effect on the outcome of the negotiation. Figure 1 shows the life cycle of a service that generally has five phases.

In the service discovery phase, the user's need is captured as an input for discovering the most appropriate service. In the SLA negotiation phase, both sides are negotiating the quality of service and an agreement achieves between them. In the monitoring phase, the negotiated service is

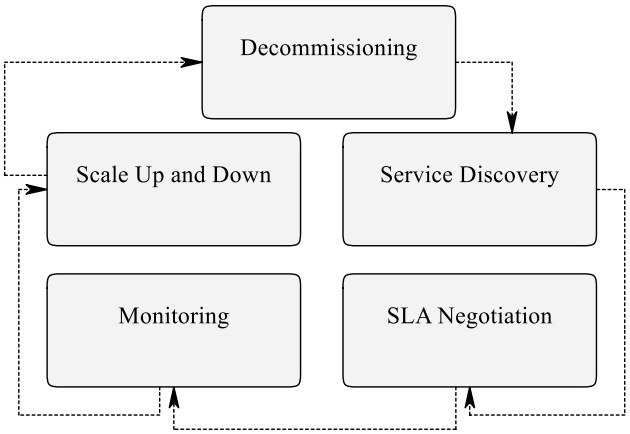

Fig. 1 The life cycle of a service

continuously monitored and if necessary, the service scale increases.

Two different scenarios have been considered to show the results of the test. In the first scenario, a third party is responsible for providing the negotiating strategy for both sides, where its main target is maximizing the number of transactions and fairness of the result for both sides. In the second scenario, there is no third party, and both sides are willing their profit. After describing the test's scenarios, some issues are addressed. First, the initial offer price and the performance of time-dependent techniques can affect social welfare and satisfaction. Second, the strategy provided for the negotiation can be effective in increasing the profit of the providers. The success or failure of the strategy is computed based on the number of allocated VMs [20]. The last issue, which is studied here, is determining how reactions to the different conditions of the market lead to an increase in the profit of the negotiation strategy. The negotiation strategy presented in this research has been tested with different load working, and the obtained results show that the time-dependent negotiation strategy can dynamically help providers increase their profits. In [9], the different pricing models have been investigated and compared.

Pay-as-you-go model pricing: This pricing model is one of the static pricing methods, and the price is determined by the provider and remains fixed. This method has some advantages, including that the customer is aware of the cost needed to pay, and the resources are reserved for the customer. However, one of the disadvantages of this method is that the service provider may allocate more resources for a customer to use. Moreover, the provider is not able to raise the price when demand increases, also the customers have to pay more in a case where the demand is low.

Subscription pricing model: This pricing model is another static pricing method wherein the price is defined based on the subscription period of the customer. In this method, sometimes the customer has to pay more than they must pay and vice versa. The advantage of this method is that the customer has to pay less for the resources if they use the resources widely, and the disadvantage of this model is that the customer has to pay more when the customer has less use of resources.

Pay-for-resources pricing model: This model is a static pricing model. In this method, productivity of resources is maximized. The difficulty of the implementation is the weakness of this model. In [21], it has been shown that if the provider uses fixed pricing, the welfare loses (Fig. 2).

When the demand is low, according to the fixed pricing method, the resource price will have a price more than the market price and cause the buyer to think of other options. Moreover, in a case where the demand is high, fixed pricing limits the welfare of the seller, because in such a case, the price of the resources can increase.

In [22], a negotiation-based resource allocation model has been presented and developed in the CloudSim environment.

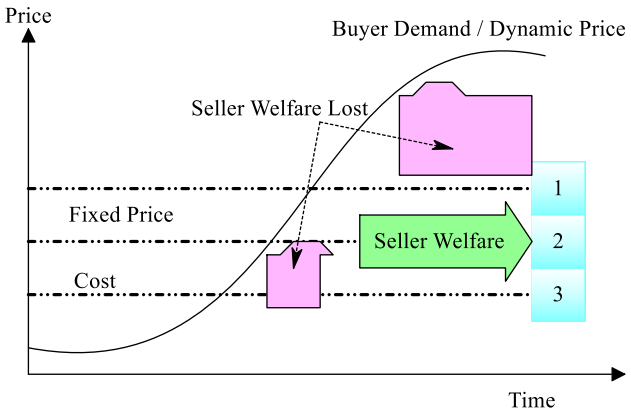

Fig. 2 The comparison of the seller profit in statistics and dynamic pricing

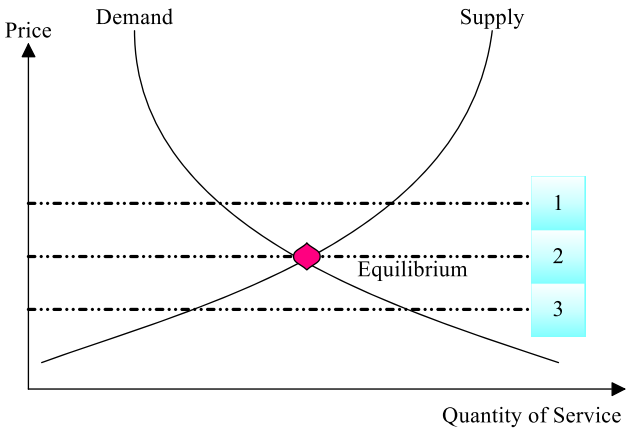

Fig. 3 Supply and demand in the market

The proposed mechanism is used so that the buyer and seller are in direct contact. Supply and demand, in a normal market model, are shown in Fig. 2.

Figure 3 shows three different cases denoted by 1, 2, and 3. Case 1 is a scenario wherein the demand is over-provisioning. Case 2 demonstrates the balance of supply and demand, and case 3 is a scenario of under-provisioning of the demand. Among these three scenarios, cases 1 and 3 are inefficient, because the traders are not able to execute the sell and buy tasks well.

Figure 4 shows the messaging process between the buyer and the seller. This means that at the time $t = 0$, an offer is sent to the buyer from the seller and usually the first message is called a template. Templates are the first offers that are sent, made by the seller, to establish a negotiation.

In [23], some software pricing factors have been introduced, including price formation, the structure of the flow assessment base, price discrimination, price bundling, and then the SBIFT (the SBIFT pricing model was introduced by Iveroth et al. as part of a comprehensive taxonomy of pricing models) model was investigated. SBIFT model is a category of different pricing methods based on the five dimensions that include scope, base, influence, formula, and temporal rights. The items of the SBIFT model and the software pricing parameters have a lot in common. According to this, by combining the items that exist in the SBIFT model and the software pricing parameters, a seven-dimension model can be obtained, and this new model can be used in the cloud concept.

In [24], a model has been presented for the providers of the cloud software service. By using the proposed model, the cloud service providers can use the resources they have access efficiently and maximize their profit. The system factors include the users, the SaaS providers, and the IaaS providers, who own the resources, supply the resources as VMs, and provide them to the SaaS providers, whereas SaaS usually rent the resources from them. In the proposed model, the system analyzes the requests sent from the users to the SaaS providers using two factors: admission control and scheduling, which decides whether their service requests

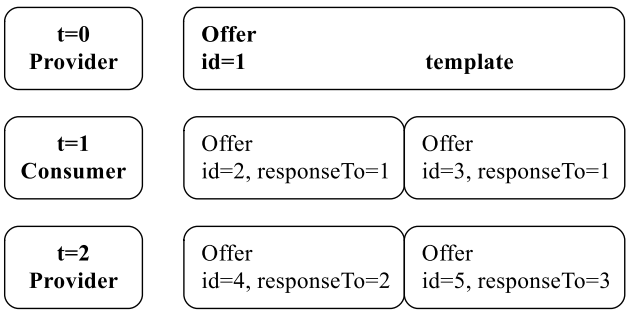

Fig. 4 Messages exchanged between the seller and the buyer in the market and in the time range $t = 0$ to $t = 2$ to reach an agreement

are accepted or not. The admission control parameter performs the analysis and makes a decision using four strategies. These strategies use the related parameters for making a decision which include inPrij (the price of the input data), outPrij (the price of the output data), iniTijl (time taken for installing the VM machine of l-type by the j provider), subT (time requested by a user), B (the maximum price that a user pays to a SaaS provider), β (penalty rate), profijlnew (profit of a SaaS), costijlnew (the sum of the costs that a SaaS provider pays to complete the process of the user request on the VM machine of l type which receives from the j provider), Pcijl (process cost), DTCjl (data transfer cost), ICijl (initial price; the initial price of the VMi machine depends on the type of the machine which creates in a data center of an IaaS provider), PDCijl (delay penalty), DTTijl (data transformation time), Tijl (new request-response time), and retijl (return of capital).

In [12], the auction pricing method is investigated. Each auction is done in a period, i.e., the time from 0 to *T*. Determining the initial price is one of the main parts of every pricing method. In the auction, the initial price of the services is determined in the following method: If, in the previous round, the average price of the auction proposed for the k3 machine is P, then the base price at the start of the next auction round for the k3 machine is considered P.

In [25], the BSM model was studied, which is a pricing model that was introduced by three economists. In this study, the cloud parameters were mapped into BSM to enable cloud pricing. The aim of [26] was to help customers select an appropriate pricing method. This research also investigated the factors that had positive and negative effects on the customer's profit. To determine the price of a service or a product, some factors, such as production and maintenance costs, competition in the market, and the recommendations of the customers, should be taken into account [27]. In [28], the tendency to use cloud process technology in companies and organizations was addressed. DOI theory is a fundamental theory and method to understand how new technology is broadcasted. Researchers obtained other factors that affect the broadcasting of a new technology, combined them with the DOI theory, and achieved the technology-organization-environment (TOE) framework. TOE is a framework for spreading innovation; this research determines how each factor in the TOE framework is effective in selecting a pricing method by the cloud providers. Moreover, this research explains that the pay-as-you-go pricing method in cloud technology is a distinction of this technology compared to other technologies. In [5], the basic question of whether to pursue cloud processing or not is addressed by providing different scenarios and available computer resources, such as the number of processors, memory, storage volume, and bandwidth, which helps the user to decide whether to buy or rent resources. The costs of the server, network, middleware

licenses, physical space requirements, costs, and expenses other than electricity costs are among the fixed costs for cloud implementation and the overhead cost of using electricity (such as cooling, idle time, the use of electronic means), internet costs, maintenance costs, hardware, data connectivity, and the cost of using servers, are the variable costs. In our research, these variable costs can be used as effective factors and parameters in the fixed part of the initial cost.

3 The proposed method

In this section, the effective parameters of the initial price are investigated and extracted [5]. The extracted factors are effective, fixed, and dynamic factors in the initial price of the VMs. Then the dynamic initial pricing model of the VMs is provided in a competitive market along with the market entities and architectures. Deals in the recommended market are executed using the relaxation of negotiation criteria. Moreover, to make complicated decisions in such a trading market, the software factors are designed.

3.1 Effective factors of the initial dynamic price section

3.1.1 Penalty rate (PR)

In a case where the cloud service providers are not able to fulfill their SLA commitments, they should pay to the consumers based on the PR selected method. In [29, 30], three effective items are defined in computing PR:

- Calculation method: There are three methods to calculate the penalty which are all based on the violation level (for example, unavailability or downtime). These three methods include (1) the ratio of the total charge which, based on the violation, a certain percentage of the amount paid by the customers will be returned; (2) fixed value, where a fixed amount will be returned to the customer; and (3) the ratio of downtime.
- Penalty capacity: The maximum value that a provider returns to the customer as a penalty.
- Payment method: Most of the providers do not pay credit or money directly to the customers as a penalty, but also the penalty will be paid from the total cost of the future purchases of the customers.

3.1.2 Market entities

In the following, the extracted entities from the market have been investigated and presented.

- VM type characteristics: The providers of the VMs categorized their resources into different VM types to provide to the demanders. Each type of VM has predefined characteristics <Core, Ram, Disk Space>.
- Market: The market entity helps the customers and providers interact with each other by creating a framework to allocate resources based on the negotiation.
- The reservoir of a record of the previous trading markets for buying the VM type (C_BehaviorDB): This includes a local database in which the transaction ID, the applicant ID of the virtual machine, the provider ID, and the result are determined.
- Resource demand file (VMDemand_File): This file includes the specifications of the VM type requested based on <Core, Ram, Disk Space>, reservation price, and the initial price, rent time of the VM, and deadline.
- Resource supply file (VMSupply_File): This file includes the characteristics of the resources supplied by the resource provider (Supply_VMType Characteristics), minimum offer price, maximum offer price, and the number of disposable resources by the provider, the quality level, and the deadline. Some of this information is confidential and some of it is not.
- BillboardSubMarControlAi: This includes the number of resource providers in the trading market, the number of applicants in the market, and the current round of the trading market.
- BillboardPrivateProj: This is used to share price information for the balancing protocol and only the children (to participate in the submarket to deliver their specific product, each of the PDA and CBA factors produce a child (sample) that can be represented in the submarket) of factors have access to its content. The recommended billboard is composed of the following:
- The maximum price that has been received from the customers for the VM with the quality X.
- The maximum price comparison between the final result and the potential for the type of virtual machine with quality X.

3.1.3 Trading scenario

The trading scenario in the recommended market includes five steps:

1. Demand submission: When an applicant requests to use the computational cloud VMs, the applicant submits the specifications of the requested VM along with the request conditions and the functions they want to use during the trading as a source demand file format (VMDemand_File), using a user interface. By submitting the request, the market coordinator (MarCoordina-

tor) orders the creation of a corresponding CA for the resource applicant to the entities' construction factors (TradingCA). The CA factor separates the Demand_VMTypeCharateristics from the VMDemand_File, delivers it to the market control factor, and the demand submit process ends.

2. Supply submission: When the resource provider wants to supply its resources in the computational cloud, the provider submits the characteristics of the supplied VM machine type (Supply_VMType Characteristics) along with the supply conditions and the functions they want to use during the deal as a resource supply file format (VMSupply_File) via the user interface in the cloud trading market environment. Once the resource supply file is submitted by the provider, the market coordinator (MarCoordinator) orders the entities' construction factor (TradingCA) to create a PA factor corresponding to the resource provider. Then, the PA factor submits the Supply_VMType Characteristics section of the reservoir related to maintaining the characteristics of the supplied resources by the resource provider (P_VMCHarDirectory) and the submission process of the supplied resource finishes.
3. Preparing the trading platform: By submitting supply and demand in the trading market and after creating the PA and CA factors, the market coordinator factor receives the Demand_VMTypeCharacteristics/VMSupply_file from every CA customer/PA provider factor. If the provider submits the supply file and it is not founded in the appropriate market, its characteristics are registered in the reservoir and wait until the creation of the submarket in accordance with the characteristics of the offered service. On the resource applicant side, in the case of a lack of finding a submarket in which the service, in accordance with the features of CA, provides the requested resources, the MarCoordinator factor searches for a market on the general blackboard wherein the requested resource is provided, regardless of quality level of the service. In the case of finding an appropriate market to create a submarket, the P_VMCHarDirectory reservoir should be searched first to find a resource provider(s) that is capable of servicing the request of the resource applicant (with the desired quality level).
4. Trading: Trade is done in discrete time. In the odd periods, the applicants of the VMs are allowed to send their recommendations, and the providers are not allowed to do anything, and in the even period, the providers of the VMs are allowed to send their recommendations, and the applicants are not allowed to do anything. In the first round of creating the market, all CBA factors announce their proposed prices (PConRoundT) for the VM providers in the submarkets in the Initial_Negotiation message format. In this round, the providers are not allowed to do anything.

Providing the transactions, a platform for the proposed algorithm and investigation of SLA in Algorithm 1 and its details are shown in Fig. 5.

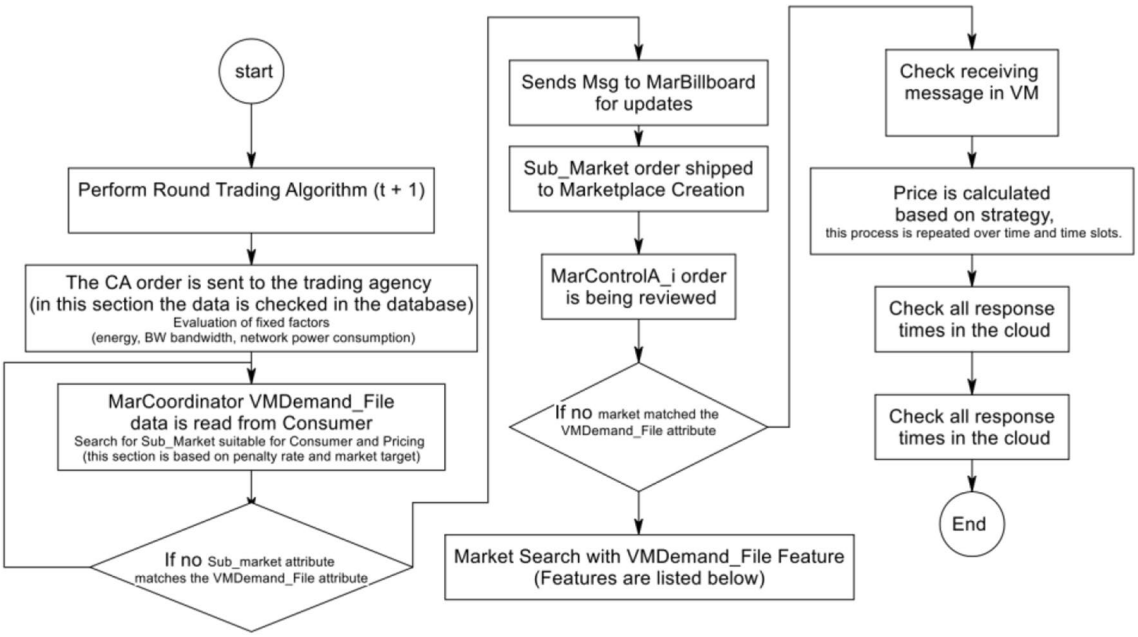

Fig. 5 Providing a trading platform for the consumer algorithm and SLA review

Table 1 Factors that work for the resource demanders [5]

Factors that work for the resource demanders [5]		
Consumer agent (CA)	Consumer broker agent (CBAi)	Consumer broker agent child (CBAi Child)
Receive VMDemand_File from demander (resource demander submit its conditions). Separation of the Supply_VMType Charateristics from the VMDemand_File and submit it in the MarCoordinator reservoir Resource providers informed from the results obtained in the transaction market using this factor Creating a PDA factor by receiving MarControlAi command	Receiving the protocol governs the transaction market from MarControlAi Checking the allowed period to maintain in the market Creating a child to locate in submarket, which is able to provide a type of the VM with X quality	Making negotiation suggestions on behalf of the resource demander in the market base on the selected algorithm Assess the performance of the resource provider offer price by the obtained price of strategy for VM in *t* period Announcing the result obtained from the negotiation to the CA factor
Factor which works for the resource providers		
Provider agent (PAi)	Provider dealer agent (PDAi)	Provider dealer agent child (PDAi Child)
Receiving VMSupply_File file from the resource provider (resource provider will submit its available resource characteristic using this factor in the cloud) Separating the Supply_VMType Charateristics part from VMSupply_File and submit it in the P_VMCHarDirectory reservoir Creating a PDA factor by receiving the command from MarControlAi	Receiving the protocol governs the market Checking the allowed period to maintain in the market Creating a child in order to locate in submarket, which able to provide a type of the VM with X quality	Making negotiate suggestions on behalf of the resource provider in the market Receiving the required information to calculate the virtual price from the MarketBillboardA and MarketBillboardAi factors Calculating the virtual price Performing the price coordination part from the negotiating protocol Sending the maximum value of the final and potential price comparison result for VM type with X quality to MarBillboadA factor for updating the private billboard
Factor which works for the resource applicants		
Coordinator (MarCoordinator)	Market control agent (MarControlAi)	Agent billboard market (MarBillboardA)
Providing a user interface to enter the data by the provider and applicant of the resource Providing a user interface to submit a resource demand file (VMDemand_File) Checking the global billboard to find a submarket in which the service in accordance with the resource features of the CA factor is selling Creating a market (MarketVM1) by receiving the resource demand file (VMDemand_File) from the VM applicant in the case of lacking an appropriate market	Announcing the command to create CBA and PDA factors to entities create factor (TradingCA) Sending the market billboard update message to the MarBillboadA factor once the CA and PA enter Controlling the market end time Announcing the submarket delete message to MarCoordinator and MarBillboadA Updating the market behavior data in the reservoir maintaining the previous transactions records of the market in buying the VM type (C_BehaviorDB)	Updating the private/submarket/global billboard by receiving the corresponding message MarCoordinator/Provider Dealer Agent/MarControlAi Sending the information contained on the billboard to PDA factor

3.1.4 Market factors

The factors involved in the market have been listed in Table 1 with their tasks. It consists of consumer agents, consumer broker agents, provider agents, etc. It also includes information on how resource demanders submit their conditions.

3.1.5 Trading transactions in the proposed market

The transaction in the proposed market has five stages that include (1) demand submission, (2) supply submission, (3) preparing the transaction platform, (4) performing the transaction, and (5) finalizing the results obtained from the transaction.

Demand submission When a resource applicant requests to use the computational cloud VMs, the requested VMs' characteristics, along with the demand conditions and functions that are willing to be used, are submitted during the transaction as a resource demand file format (VMDemand_File) via the user interface. By submitting the market demand coordinator (MarCoordinator), the command is issued to create the CA factor corresponding to the resource applicant for the entities' construction factor (TradingCA). The CA factor separates the Demand_VMTypeCharateristics from the VMDemand_File and delivers it to the market control factor and the demand submission process ends.

Supply submission When the resource provider wants to supply its resources in the computational cloud, submits the characteristics of the supplied VM machine type (Supply_VMType Characteristics), along with the supply conditions and the functions they want to use during the trade as a resource supply file format (VMSupply_File) via the user interface in the cloud trading market environment. Once the resource supply file has been submitted by the provider, the market coordinator (MarCoordinator) orders the entities' construction factor (TradingCA) to create a PA factor corresponding to the resource provider. Then the PA factor submits the Supply_VMType Characteristics section of the reservoir related to maintaining the characteristics of the supplied resources by the resource provider (P_VMCHarDirectory) and the submission process of the supplied resource finishes.

Preparing the trading platform By submitting the supply and demand in the trading market, and after creating the PA and CA factors, the market coordinator factor receives the Demand_VMTypeCharacteristics/VMSupply_File from every CA customer/PA provider factor and searches a submarket in the general blackboard, wherein a service matching the requested features of the CA factor and/or matching with the provided service by the PA factor is sold/supplied. If the provider submits the supply file and it is not founded in the appropriate market, its characteristics are registered in the reservoir and wait until the creation of the submarket in accordance with the characteristics of the offered service. On the resource applicant side, in the case of a lack of finding a submarket in which the service, in accordance with the features of CA, provides the requested resources, the MarCoordinator factor searches for a market on the general blackboard wherein the requested resource is provided, regardless of the quality level of the service. In a case of finding the appropriate market to create a submarket, the P_VMCHarDirectory reservoir should be searched first to find a resource provider(s) that is capable of servicing the request of the resource applicant (with the desired quality level). Finding an appropriate submarket means the existence of the market and the appropriate MarControlAi factor; therefore, the MarCoordinator does nothing to set up a similar market and puts the CA/PA factor into the available market, and the MarControlAi factor sends the establishing message of a negotiator PDA/CBA to the corresponding PA/CA. The PDA/CBA factor locates in the market and creates a child, then locates in the submarket. MarControlAi sends the updating command to update the information contained in the blackboard to the MarBillboardA factor.

If the provider submits the supply file and the submarket is not found, its characteristics are submitted in the reservoir and wait until the creation of a submarket matching the features of the provided service. On the source applicant side, in the case of lack of finding a submarket wherein a service matches with the features of the requested resource of the CA factor, the MarCoordinator factor in the general blackboard searches a market wherein the demanded resource is provided, regardless of the quality level of the service. In the case of finding the appropriate market to create a submarket, the P_VMCHarDirectory reservoir should be searched first to find a resource provider(s) that is capable of servicing the request of the resource applicant (with the desired quality level). In the case of a lack of finding a list of resource providers, no actions are not taken for the transaction (existence of at least one provider to start the transaction is necessary).

Finding an appropriate submarket means the existence of the market and the appropriate MarControlAi factor; therefore, the MarCoordinator does nothing to set up a similar market and puts the CA/PA factor into the available market, and the MarControlAi factor sends the establishing message of a negotiator PDA/CBA to the corresponding PA/CA. The PDA/CBA factor locates in the market and creates a child, then locates in the submarket. MarControlAi sent the updating command to update the information contained in the blackboard to the MarBillboardA factor. In the case of lack of finding an appropriate market, if the provider submits the supply file, its characteristics are recorded in the reservoir, and it should wait until the

creation of a market that matches the features of the provided service. For the resource applicant, in the case of lack of finding a market, it first searches for a P_VMCHar-Directory reservoir to find a resource provider(s) which is (are) able to service the resource applicant's demand (with the desired quality level). In a case of finding resource provider(s), it creates a market and submarket and issues a command for the market construction factor (MarplaceAC) to create them. Moreover, it sends the message to the entity construction factor (TradingAC) for creating the market control factor (MarControlAi), and the applicant/provider of VMs puts the PVs in the new markets. After locating the MarControlAi, PA, and CA in the market, the MarControlAi factor sends a message for creating the CBA and PDA trader factors to the entity construction factor (TradingAC). The PDA/CBA factor creates a child and then puts it in the submarket. MarCoordinator and MarControlA send the command to the MarBillboardA factor for updating the billboard to update the information contained in the general blackboard and market. By constructing the trader factors, these factors receive the negotiation protocol from (MarControlAi). The negotiation clue between the traders is created by the (MarControlAi) factor. Finally, the billboard updating command sends the MarBillboardA factor to update the information contained in the submarket billboard.

It should be noted that if CBA/PDA corresponding to the applicant/provider of VM type exists in the market and the applicant/provider again enters the same market to buy/supply the VM type, its corresponding CBA/PDA will not be constructed, and CBA/PDA will construct a child as its representative and put it into an appropriate market.

Trading The deal is done in discrete time. In the odd periods, the applicants of the VMs are allowed to send their recommendations, and the providers are not allowed to do anything, and in the even period, the providers of the VMs are allowed to send their recommendations, and the applicants are not allowed to do anything. In the first round of creating the market, all CBA factors announce their proposed prices (PConRoundT) for the VMs providers in the submarkets in the Initial_Negotiation message format. In this round, the providers are not allowed to do anything. In the second round, PDA factors act as follows [5]:

- Create a price based on the selected strategy (PProCtr, Round T)

Comparing the results of the applicant's offered price (PConRoundT-1) with the price obtained from the selected strategy (PProCtr, Round T), if the offered price by the applicant is more than the price obtained from the strategy, the offer of the VM applicant is submitted and enters the price coordination stage, 2-1. Otherwise, it enters the price virtualizing stage. If constructing a virtual price is not possible, the price obtained from the selected strategy (PProCtr, Round T) is submitted and enters the coordination stage. However, if constructing the virtual price is possible, it compares the virtual price and the price obtained from the selected strategy (PProCtr, Round T); if the virtual price is more, then it enters the coordination stage as the submitted price. Otherwise, the price obtained from the selected strategy enters the coordination stage.

Comparing the prices acquired in the coordination stage, and finally, the message for accepting the VM type, applicant price, and/or the new price will be announced by the New_Price message format.

The consumers receive the price from the VM type provider and compare the offered prices of the VM provider with the price obtained from the selected strategy. If the price offered by the VM provider is less than the price obtained from the selected strategy, the price will be accepted and will enter the finalizing step of the result obtained from the deal. Otherwise, the price obtained from the strategy will be announced to the VM providers as a new message format. The negotiations between the applicant and the provider of the VM type will continue in this way until the last round is reached, and in the case of failure to achieve an agreement in the negotiation, Resource_allocation_process_Termination messages must be sent to the CBA (PDA) and MarControlA.

Virtual price construction stage By reaching half of the rounds in which a VM provider can stay in the market, and in case of the failure of an agreement, to improve the negotiation trend, the virtual price will be constructed. Before constructing a virtual price, it is necessary to check whether there is a possibility or not; therefore, (1) the number of allowed rounds to keep itself in the market will be checked and if it reaches half of the rounds that a VM provider needs to stay in the market, it enters the next stage. (2) When comparing the offered price by the VM type applicant with the maximum price offered by the customer for the machines with lower quality, if the applicant's offer price is higher than or equal to the maximum offered price for the VM type with the lower quality, constructing the virtual price is possible; otherwise, given that the VM type provider is not allowed to sell the higher quality VM with the lower price with a lower price of lower quality VM, constructing the virtual price is not possible.

In the case of the possibility to construct the virtual price, PDA requested the following factors from the PDA, respectively [5, 31, 32] (Fig. 6):

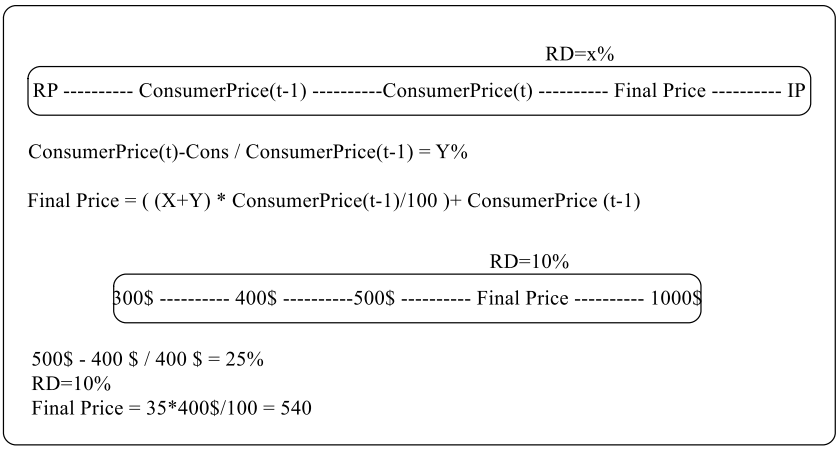

Fig. 6 The number of competitors and the number of applicants under the VM type (information on the private market blackboard)

- MarBillboardA: the number of competitors and the number of applicants under the VM type (information on the private market blackboard)
- Provider Dealer Agent: the number of successful instances of cooperation between the VM provider and buyer
- Market Control Agent: the number of instances of cooperation of a VM applicant with other providers
- Given that the calculated flexibility is a percentage value, the following method is used to convert it to a currency unit.

Coordination stage In the stage, it will be controlled that the VMs with any quality are not sold with a price lower than the maximum price of the VMs with the lower quality.

The input of this stage has the following three cases:

The offered price of the VM type applicant is higher than the price offered based on the selected strategy of the VM provider. In this case, a price that is entered to this stage to be checked will be the offered price by the VM applicant.

The offered price of the VM applicant is lower than the price offered based on the selected strategy of the VM provider, it is lower than the maximum price of the VMs with lower quality (MaxPvnwithSowerQ), and the construction is not possible. The price that is entered in the stage will be the price offered based on the selected strategy.

The offered price of the VM applicant is lower than the price offered based on the selected strategy of the VM provider; it is higher than the maximum price of the VMs with lower quality (MaxPvnwithSowerQ) and enters the virtual price construction stage. The output of the negotiation of the virtual price construction stage will enter the coordination stage.

In this stage, the price of every VM type compares with the maximum price of the VMs with lower quality, which is introduced as the submitted price. If the price of the VM type is higher than the maximum price of the VMs with lower quality, the price of the VM type (the VM type, applicant offered price, or price obtained from the strategy) is announced as the final price, and if the VM price is more than the maximum VM price with lower quality, the maximum VM price with lower quality is announced as the final price.

Finalizing the results obtained from the deal Finalizing the results obtained from the deal will be done by receiving two messages: accept, reject, and/or Resource_allocation_process_Termination.

If the "accept" message is received, the CBA (and corresponding PDA) factors announce the deal result to the MarControlA factor and CA (and corresponding PA) factor trading on its behalf. The information on the customer behavior and the results of the transaction are updated by the market control factor (MarControlA) in the record deal maintaining reservoir and the previous transactions in the purchase of VMs (C_BehaviorDB). Moreover, PA is responsible for updating the remaining capacity of the reservoir used for maintaining the specifications of supplied resources (P_VMCHarDirectory). Then, the MarControlA factor deletes the transaction clue between the PDA and CBA factors. By deleting the transaction clues, the MarControlA factor sends the billboard updating message to the MarBillboardA factor, and the MarBillboardA factor updates the submarket billboard. Finally, the PA factor schedules and runs negotiated demand for its physical resources and updates the remaining capacity in the reservoir (P_VMCHarDirector).

If the "reject" message is received from the provider to the applicant or vice versa, the CBA/PDA factor must announce the result to the MarControlA factor as well as the CA/PA factor that deals on its behalf. The MarControlA factor deletes the transaction clue between the PDA and CBA factors. By deleting the transaction clues, the MarControlA factor sends the billboard updating message to the MarBillboardA factor, and the MarBillboardA factor updates the submarket billboard.

If the "Resource_allocation_process_Termination message" is received, which means that a round has been reached where the provider is not capable of staying in the market (NProRound = Deadline), it is necessary that the PDA factor announces the MarControlA factor and the PA factor

Table 2 Example 1—the offer price of three different customers for three different virtual machines

Virtual machine type/client ID	VM_G	VM_S	VM_B
C1	400$	250$	130$
C2	540$	210$	170$
C3	600$	190$	200$
Maximum price of source applicants with quality grade *X*	600$	250$	200$

Table 3 Example 2—the offer price of three different customers for three different virtual machines

Virtual machine type/client ID	VM_G	VM_S	VM_B
C1	350$	250$	130$
C2	430$	450$	290$
C3	410$	380$	200$
Maximum price of source applicants with quality grade *X*	430$	450$	290$

that deals on its behalf, and if they exist, CBA/CBAs that are negotiating with it to pass the Deadline. The MarControlA factor deletes the transaction clue between the PDA and CBA/CBA factors. By deleting the transaction clues, the MarControlA factor sends the billboard updating message to the MarBillboardA factor, and the MarBillboardA factor updates the submarket billboard.

If the "Resource_allocation_process_Termination" is received, which means that a round has been reached where the VM applicant is not capable of staying in the market (NMarRoundk = NConRound), it is necessary that the CBA factor announces the MarControlA factor and the CA factor that deals on its behalf, and if they exist, PDA/PDAs that are negotiating with it to pass the Deadline. The MarControlA factor deletes the transaction clue between the CBA and PDA/PDA factors. By deleting the transaction clues, the MarControlA factor sends the billboard updating message to the MarBillboardA factor, and the MarBillboardA factor updates the submarket billboard.

Some samples of the described architecture are as follows: (examples 1 and 2 in Tables 2, 3, 4, and 5).

Table 2 shows the offer price of three different customers for three different virtual machines.

To pass the deadline, it is essential that the CBA factor announces both the MarControlA factor and the CA factor that deals on its behalf, as well as PDA/PDAs negotiating with it. When added to CBA and PDA/PDAs, the MarControlA factor deletes the transaction clue. To update the submarket billboard, the MarControlA factor deletes the transaction clues and then sends the billboard updating message to the MarBillboardA factor (Table 4).

4 Simulation and results

In designing a system for a large-scale cloud data center, it is necessary to perform various tests and experiments on the cloud infrastructure to evaluate the proposed algorithm, given that performing such rests on the cloud infrastructure is very difficult and costly, especially if it is necessary to repeat the tests (Fig. 7). For this reason, to evaluate the method proposed in this research, a simulator called CloudSim is used, which is an open-source library based on the Java language and the NetBeans programming environment is used [5]. Moreover, Docker has been used for information storage. The input of the system is a set of tasks that denote a working load (limited to the processor and/or input-output). The number of commands in terms of millions, the number of processing units, the size of the input and output file, the time entering the process, and the production model of the memory bandwidth and the processor, compose a task. To perform the simulation, it is necessary that the number of commands in terms of millions, as well as the time entering the process, be determined randomly. To obtain a random value for the number of commands, the normal distribution function with average value $\mu = 4000$ and standard deviation a2 = 10,000 was used. As a result, the entering time of about 95% of the generated values fall in the range of 20,000 to 60,000. The Poisson distribution function was also used to obtain the random value for the tasks' entering the time parameter. Since we want, on average, 100 tasks to enter the cloud network in each time unit, $h = 100$ has been considered. To prevent mathematical-related complexities, the Colt package was used. The size of the input and output files of every duty was considered a constant value equal to 3000 kb.

To evaluate the proposed algorithm, two scenarios were considered for the transaction in the market: using the regression for categorizing the supplied VMs and the lack of using a regression for categorizing the VMs. To implement the proposed algorithm, Docker and NetBeans were used as storage space and for algorithm development, respectively.

4.1 Market and dataset generation

Given the specifications mentioned for a proper dataset, to evaluate the efficiency of the proposed method, the design and simulation of the desired dataset have been done. In the following, the proper evaluating criteria for each part of the proposed method are defined and described to determine a definite criterion and compare the efficiency of the proposed method. Cloud as the only connector technology, processing and distributed

Table 4 Details of offer price for three different virtual machines in Table 2

Virtual machine type	VM_G	VM_S	VM_B	VM_G	VM_S	VM_B	VM_G	VM_S	VM_B
Get the offer price customer	400$	250$	130$	540$	210$	170$	600$	190$	200$
Step 1: build potential cross-price (by strategy)	420$	270$	110$	560$	230$	190$	580$	230$	210$
Sub-step 1–2: check the second condition	400$ ≤ 420$	250$ ≤ 270$	130$ ≤ 110$	540$ ≤ 560$	210$ ≤ 230$	170$ ≤ 190$	600$ ≤ 580$	190$ ≤ 230$	200$ ≤ 210$
Step 2: check possibility of virtualization price									
Sub-step 2–2: check the second condition	400$ > 250$	250$ > 200$	No need to check	540$ > 250$	210$ > 200$	No need to check	No need to check	190>200	No need to check
Step 3: build virtual price	VP = 30$	VP = 10$	No need to check	VP = 40$	VP = 15$	VP = 25$	No need to check	No need to check	VP = 15$
Step 4: check productivity of constructing the final and potential cross-price	U (400$ + 30$)? ≤ U (420$)	U (250$ + 10$)? ≤ U(270)	No need to check	U (540$ + 40$)? ≤ U (560$)	U (210$ + 15$)? ≤ U (230$)	U (170$ + 25$)? ≤ U (190$)	No need to check	No need to check	U (200$ + 15$)? ≤ U (210$)
	400$	270$	130$	540$	230$	170$	600$	230$	200$
Step 5: implementation protocol synchronization by balancing final price to deliver	Max (400$, 270$, 200$)	Max (270$, 200$)	130$	Max (540$, 270$, 200$)	Max (230$, 200$)	170$	Max (600$, 270$, 200$)	Max (230$, 200$)	200$
Making the final price	400$	270$	130$	540$	230$	170$	600$	230$	200$

Table 5 Details of offer price for three different virtual machines in Table 3

Virtual machine type	VM_G	VM_S	VM_B	VM_G	VM_S	VM_B	VM_G	VM_S	VM_B
Get the offer price from customer	350$	250$	130$	430$	450$	290$	410$	380$	200$
Step 1: build potential cross-price (according to strategy)	425$	270$	110$	440$	220$	170$	535$	200$	185$
Sub-step 1–2: check the second condition	350$ ≤ 425$	250$ ≤ 270$	130$ ≤ 150$	430$ ≤ 440$	450$ ≤ 220$	290$ ≤ 170$	410$ ≤ 535$	380$ ≤ 200$	200$ ≤ 185$
Step 2: check possibility of virtualization price									
Sub-step 2–2: check the second condition	350$ ≥ 450$	250$ > 290$	No need to check	430$ > 450$	No need to check	No need to check	410$ ≥ 450$	No need to check	No need to check
Step 3: build virtual price	No need to check	No need to check	VP = 20	No need to check	No need to check	No need to check	No need to check	No need to check	No need to check
Step 4: check productivity of constructing the final and potential cross-price and	No need to check	No need to check	U (130$ + 20$)? ≤ U (150$)	No need to check	No need to check	No need to check	No need to check	No need to check	No need to check
	425$	270$	130$	440$	450$	290$	535$	380$	200$
Step 5: implementation protocol synchronization by balancing final price to deliver	Max (425$, 450$, 290$)	Max (270$, 290$)		Max (440$, 450$, 290$)	Max (450$, 290$)		Max (535$, 450$, 290$)	Max (380$, 290$)	
Making the final price	450$	290$	130$	450$	450$	290$	535$	380$	200$

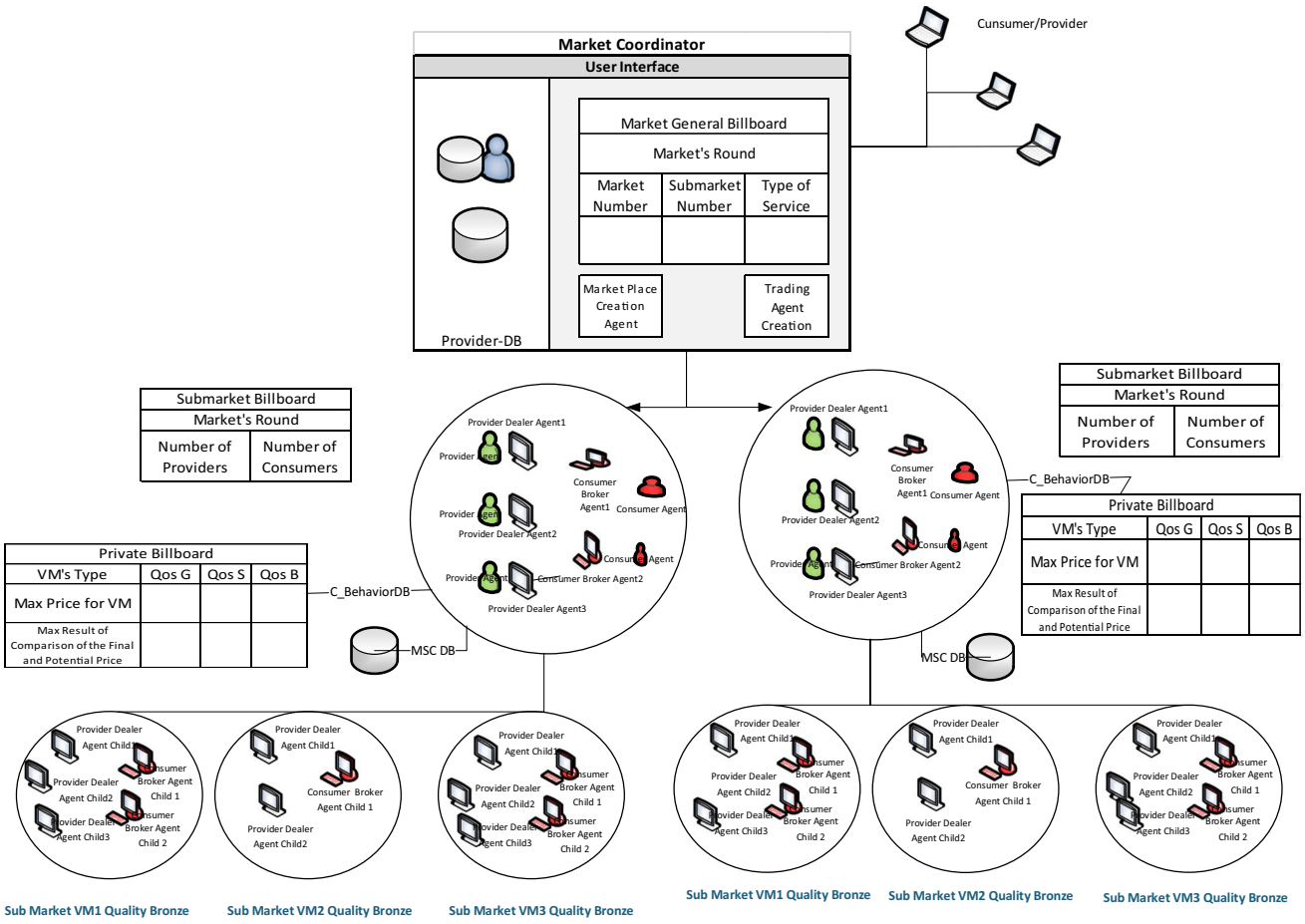

Fig. 7 The details of implementation for market coordinator

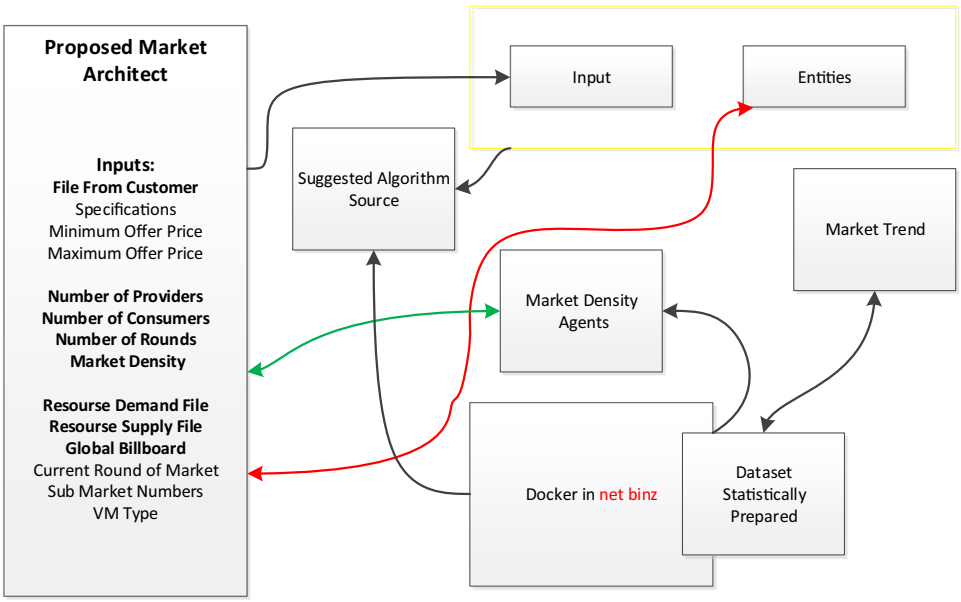

Fig. 8 The details of implementation for market coordinator parameters are divided into two submarket and global levels

resources to market demand and supply, includes many features of the demand market and free market. This shows the fact that every VM and its corresponding demand are known as a product in a market, called the processing resource market, and as a result, the cloud supplies its rental and processing products in the free market. One of the most important features of the products' free market, which is highly emphasized in the literature of free market, is the demand iterative algorithm. This algorithm provides a reliable structure for supply and demand products, which helps the business manager to make decisions. Therefore, in this research, the dataset is simulated based on the demand process repeated on different days. This means that, in the presence of a little noise, the trend and amount of demand in different workloads and demands are iterated.

To generate and simulate this dataset, the data generation is done in a 2-min period, wherein at the start of each round, the algorithm of the proposed method runs automatically. Therefore, in 1 day, there are $30 \times 24 = 720$ iterations. Another parameter that should be taken into account in dataset generation is how to input the demand into the cloud. This, indeed, shows the amount of demand distribution at different times of the day. Although, the prediction of the workload amount based on the time has not been made yet, and the relationship between consecutive values in the time series of predicted workload will be decisive. To distribute the demands at different times of the day, in this research, we make a function by combining some normal functions (Fig. 8) [7, 14, 17, 18].

Table 6 Details of parameters, the submarket level, and specific quality level

Input parameters	Expected values		
Number of data center	3		
Number of VM	3		
Number of physical sources (R)	3		
Characteristics of the VM	Small ($m = 1$)	Medium ($m = 2$)	Large ($m = 3$)
	CPU ($w_{11} = 1$)	CPU ($w_{21} = 2$)	CPU ($w_{31} = 4$)
	Memory (GB): $w_{12} = 1.7$	Memory (GB): $w_{22} = 3.75$	Memory (GB): $w_{32} = 7.5$
	Storage (GB): $w_{13} = 160$	Storage (GB): $w_{23} = 410$	Storage (GB): $w_{33} = 850$
Max available amount of each VM type (L)	200		
Final/reservation/price (RP)	By user		
Negotiation deadline (number of period)	Long: 50–100	Moderate: 40–50	Short: 20–40
Time-dependent strategy (A)	Conciliatory: 1.5	Linear: 1	Conservative: 3
Provider initial price (IP)	By user		
System initial price	Calculated by system based on dynamic factors (Section 3)		
Penalty rate (PR)	Demonstrate in percentage		

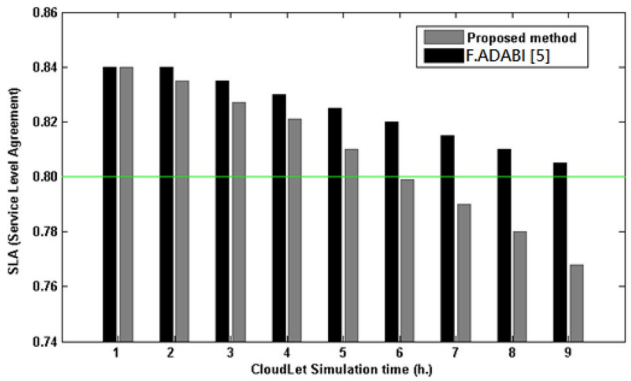

Fig. 9 The SLA for the proposed algorithm

The parameters on the global level mean that they do not depend on the market type, are consumer-dependent, and if calculated, it can be used for a consumer in all submarkets wherein there is consumer presence. The parameters in the market level mean that, by changing each submarket, the results will change. Therefore, for each submarket, this parameter must be calculated. Moreover, this parameter is the same for all consumers and does not need to be calculated separately for each consumer.

Degree of competition (DC) ∈ [0,1] denotes the degree of competition that the supply factor faces in round *t*. DC is calculated from the following formula (Eq. (1)) by considering the competitors and the trading partners:

$$\mathrm{DC} = \left[\frac{m_t^a - 1}{m_t^a}\right]^{n_t^a} \tag{1}$$

where $m_t^a - 1$ denotes the number of VM-type applicants in round *t*, andn_t^a denotes the number of competitors. The more/less the DC, the more/less competition a provider faces and needs more/less coordination negotiation to reach an agreement.

This parameter is calculated for a submarket level and a specific quality level of the service.

This parameter is calculated in each round.

Time pressure (time pressure: TP) ∈ [0,1], this parameter denotes the time pressure on the side of the provider factor. The provider factor calculates the time pressure according to the following model:

Where *T* denotes the deadline and *t* is the current time, the more/less Tp is denotes the more/less TP and, as a result, needs more/less coordination negotiation to reach an agreement (Table 6).

These parameters are calculated at the submarket level and for a specific quality level of service.

4.2 Running the proposed algorithm in the simulation platform

To evaluate the proposed method, the market algorithm was performed in the CloudSim environment in two cases of using/not using the regression. The SLA comparison obtained from the performing of both algorithms shows that using the regression for separating data in the market leads to an improvement in SLA (Fig. 7).

In Fig. 9, the green line indicates the standard value of SLA. During the performing market in the cloud network, if the SLA output falls under the green line, it means the algorithm is optimized and, if it falls above the green line, it indicates a failure

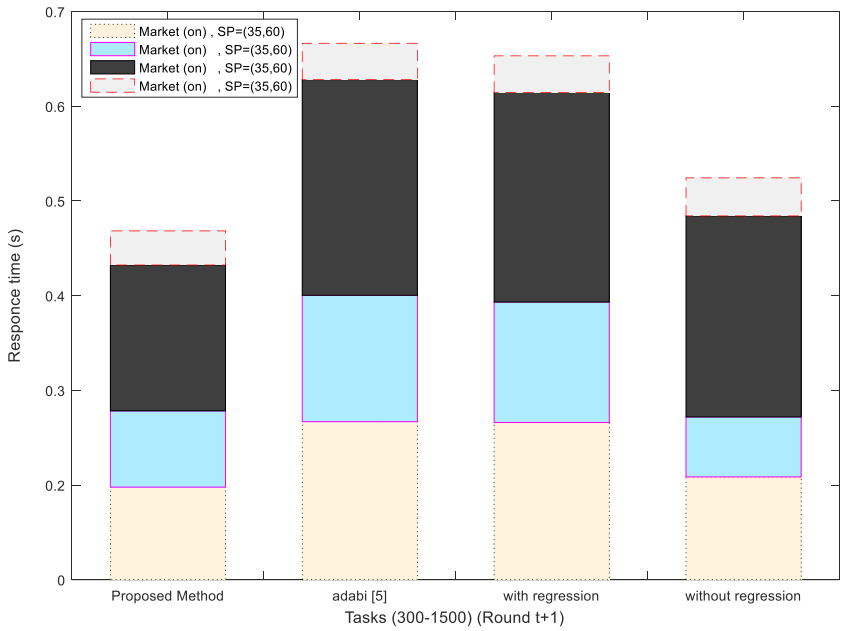

Fig. 10 Response-time review for the proposed algorithm

in SLA. As can be seen, when using the proposed algorithm, from time 6 s upward, the SLA moves toward being more optimal, which shows that the proposed method is efficient for the cloud platform.

4.2.1 Checking response time

A comparison is made between the proposed method and others in this section. SP (36, 60) means that BAZAR is considered with 36 input data and 60 rounds of reputation data. To do this, the variations of the response time from time slots t to $t + 1$ per market creation per hour were investigated. The proposed method is less time consuming than etiquette and other methods. The load formed with the initial price in the proposed method has a shorter response time, indicating that it has a shorter response time (Fig. 10). The number of tasks from 300 to 1500 was tested for active market modes. The result shows that the proposed method was not efficient for reducing the response time.

5 Conclusions

Using cloud services, we provided users with access to virtual resources. A market-driven approach is one of the most common methods to manage virtual and physical machines in a network based on qualitative and quantitative findings. A novel method for setting an SLA using dynamic initial pricing of VMs in a competitive market is presented in this study. Cloud services are most commonly managed using a market-oriented approach. In this study, the main problem is constructing a VM's initial price given the market's dynamic conditions. Deals in the market are negotiation-based, and the negotiations are conducted using multifactor architecture. By categorizing VMs using regression, the results indicate that SLA is improved by using regression. When the proposed method is applied and the market is constructed in the Docker and CloudSim simulators, using the resources and checking the response times of the market are optimized compared to when the proposed method is not applied. As part of this study, the process of the construction market and the price of the initial construction project are also discussed, as well as how the service-level agreements are handled. Simulated results indicate that when assigning virtual machines to the market, the cloud market system provides improved service-level agreements (SLAs) and response times. As a result, using the regression method for categorizing the VMs improved the SLA.

For future works, the machine learning [33] and metaheuristics, such as particle swarm optimization (PSO), whale [11, 34], or other prediction methods, can be used to find the initial price for virtual machines in a competitive market. Furthermore, we recommend incorporating the security value of the VMs into the new structure, considering the recommendations provided by the Information Technology Infrastructure Library (ITIL) and IT Service Management (ITSM).

Funding Open access funding provided by FCT|FCCN (b-on).

Declarations

Conflict of interest The authors declare no competing interests.

References

1. Wu, X.; Pellegrini, F. Performance analysis of QoS-differentiated pricing in cloud computing: an analytical approach. 2017 https://arxiv.org/abs/1709.08909.
2. Dastjerdi V, Buyya R (2015) An autonomous time-dependent SLA negotiation strategy for cloud computing. Comput J *58*:3202–3216
3. Raj JS, Smys S (2019) Virtual structure for sustainable wireless networks in cloud services and enterprise information system. J ISMAC 1(03):188–205
4. Hasselmeyer P, Koller B, Kotsiopoulos I, Kuo D, Parkin M (2007) Negotiating slas with dynamic pricing policies. Proceedings of the SOC@ Inside, Access online: 7. https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=365c3c929e976442062d39418f2fef268082083c
5. Adabi S, Alayin F, Sharifi A (2019) A new flexible pricing mechanism considering price–quality relation for cloud resource allocation. Evol Syst *12*:541–565
6. Pańkowska M, Pyszny K, Strzelecki A (2020) Users' adoption of sustainable cloud computing solutions. Sustainability *12*:9930
7. Javadpour A, Wang G, Rezaei S, Chend S (2018) Power curtailment in cloud environment utilising load balancing machine allocation. In: Proceedings of the 2018 IEEE SmartWorld, Ubiquitous Intelligence Computing, Advanced Trusted Computing, Scalable Computing Communications, Cloud Big Data Computing. Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI) IEEE, pp 1364–1370
8. Javadpour A, Wang G, Xing X (2018) Managing heterogeneous substrate resources by mapping and visualization based on software-defined network. In: Proceedings of the 2018 IEEE Intl Conf on Parallel Distributed Processing with Applications, Ubiquitous Computing Communications, Big Data Cloud Computing, Social Computing Networking. Sustainable Computing

Communications (ISPA/IUCC/BDCloud/SocialCom/SustainCom) IEEE, pp 316–321
9. Martin JP, Cendrowski H (2014) Cloud computing and electronic discovery. Wiley, New York, NY, USA
10. Ardagna D, Casale G, Ciavotta M, Pérez JF, Wang W (2014) Quality-of-service in cloud computing: modeling techniques and their applications. J Internet Serv Appl *5*:11
11. Javadpour A, Rezaei S, Li K-C, Wang G (2019) A scalable feature selection and opinion miner using whale optimization algorithm. In: Proceedings of the International Symposium on Signal Processing and Intelligent Recognition Systems. Springer, Singapore, pp 237–247
12. Kumar N, Saxena S (2015) A preference-based resource allocation in cloud computing systems. Procedia Comput Sci *57*:104–111
13. Sangaiah AK, Javadpour A, Pinto P, Chiroma H, Gabralla LA (2023) Cost-effective resources for computing approximation queries in mobile cloud computing infrastructure. Sensors 23(17):7416
14. Javadpour A, Wang G (2021) cTMvSDN: improving resource management using combination of Markov-process and TDMA in software-defined networking. J Supercomput *78*:3477–3499
15. Chun S-H (2020) Cloud services and pricing strategies for sustainable business models: analytical and numerical approaches. *Sustainability 12*:49
16. Mirmohseni SM, Javadpour A, Tang C (2021) LBPSGORA: create load balancing with particle swarm genetic optimization algorithm to improve resource allocation and energy consumption in clouds networks. Math Probl Eng 2021:1–15
17. Mirmohseni SM, Tang C, Javadpour A (2020) Using Markov learning utilization model for resource allocation in cloud of thing network. Wirel Pers Commun 115:653–677
18. Javadpour A, Abharian SK, Wang G (2017) Feature selection and intrusion detection in cloud environment based on machine learning algorithms. In: Proceedings of the 2017 IEEE International Symposium on Parallel and Distributed Processing with Applications and 2017 IEEE International Conference on Ubiquitous Computing and Communications (ISPA/IUCC). IEEE, pp 1417–1421
19. Ja'fari F, Mostafavi S, Mizanian K, Jafari E (2021) An intelligent botnet blocking approach in software defined networks using honeypots. J Ambient Intell Humaniz Comput 12:2993–3016
20. Javadpour A (2019) Improving resources management in network virtualization by utilizing a software-based network. Wirel Pers Commun 106:505–519
21. Mihailescu M, Teo YM (2010) Dynamic resource pricing on federated clouds. In: Proceedings of the 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing. IEEE, pp 513–517
22. Pittl A, Mach W, Schikuta E (2015) A negotiation-based resource allocation model in IaaS-markets. In: Proceedings of the 2015 IEEE/ACM 8th International Conference on Utility and Cloud Computing (UCC). IEEE, pp 55–64
23. Laatikainen G, Ojala A, Mazhelis O (2013) Cloud services pricing models. In: Proceedings of the International Conference of Software Business. Springer Berlin Heidelberg
24. Wu L (2014) SLA-based resource provisioning for management of cloud-based software-as-a-service applications (Doctoral dissertation). Retrieved from University of Melbourne Cloud Laboratory. http://cloudbus.org/students/LinlinPhDThesis2014.pdf
25. Rong J, Qin T, An B (2019) Competitive cloud pricing for long-term revenue maximization. J Comput Sci Technol *34*:645–656
26. Basu S, Chakraborty S, Sharma M (2015) Pricing cloud services—the impact of broadband quality. Omega *50*:96–114
27. Li Z, Tan D (2017) Two-stage dynamic pricing and advertising strategies for online video services. Discret Dyn Nat Soc 2017:1–8
28. Sungheetha A, Sharma R (2020) Service quality assurance in cloud data centers using migration scaling. J Inform Technol 2(01):53–63
29. Xiaoyong Y, Ying L, Tong J, Tiancheng L, Zhonghai W (2015) An analysis on availability commitment and penalty in cloud SLA. In: Proceedings of the 2015 IEEE 39th Annual Computer Software and Applications Conference, vol 2. IEEE, pp 914–919
30. Adabi S, Mosadeghi M, Yazdani S (2018) A real-world inspired multi-strategy based negotiating system for cloud service market. J Cloud Comput 7:17
31. Javadpour A, Adelpour N, Wang G, Peng T (2018) Combing fuzzy clustering and PSO algorithms to optimize energy consumption in WSN networks. In: Proceedings of the 2018 IEEE Smart World, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing. Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI), Guangzhou, China, IEEE, pp 1371–1377
32. Javadpour A, Abadi AMH, Rezaei S, Zomorodian M, Rostami SA (2021) Improving load balancing for data-duplication in big data cloud computing networks. Cluster Comput 25(4):2613–2631
33. Javadpour A, Wang G, Rezaei S, Li K-C (2020) Detecting straggler MapReduce tasks in big data processing infrastructure by neural network. J Supercomput 76:6969–6993
34. Javadpour A, Rezaei S, Sangaiah AK, Slowik A, Khaniabadi SM (2021) Enhancement in quality of routing service using metaheuristic PSO algorithm in VANET networks. In: Soft Computing. Springer