Special Issue Published in International Journal of Trend in Research and Development (IJTRD), ISSN: 2394-9333, www.ijtrd.com

A Novel Reputation System for Intelligent Economic Approach in Cloud Services

S.Vinothkumar, M.E., Assistant Professor,
Department of Computer Science and Engineering, ACS College of Engineering,
Bengaluru, Karnadaka.
svinothkumar1984@gmail.com

Abstract—With Inter-Cloud, distributed cloud and open cloud exchange (OCX) emerging, a comprehensive resource allocation approach is fundamental to highly competitive cloud market. Oriented to infrastructure as a service (IaaS), an intelligent economic approach for dynamic resource allocation (IEDA) is proposed with the improved combinatorial double auction protocol devised to enable various kinds of resources traded among multiple consumers and multiple providers at the same time enable task partitioning among multiple providers. To make bidding and asking reasonable in each round of the auction and determine eligible transaction relationship among providers and consumers, a price formation mechanism is proposed, which is consisted of a back propagation neural network (BPNN) based price prediction algorithm and a price matching algorithm. A reputation system is proposed and integrated to exclude dishonest participants from the cloud market. The winner determination problem (WDP) is solved by the improved paddy field algorithm (PFA). Simulation results have shown that IEDA can not only help maximize market surplus and surplus strength but also encourage participants to be honest.

Index Terms—Cloud, resource allocation, combinatorial double auction, price formation, reputation

Ç

1 INTRODUCTION

CLOUD computing provides virtually unlimited comput-ing power as utility service to consumers. It enables different provisioning models for on-demand access to applications (software as a service, or SaaS), platforms (plat-form as a service, or PaaS), and computing infrastructures (IaaS). It has created a competitive market where consumers pay providers for using resources and are usually billed using a pay-as-you-go model. To facilitate trading, a market mechanism should be explored to allocate and utilize resources within their capacities without over-provisioning or under-provisioning [1].

Resources in the cloud are usually geographically distrib-uted; may be heterogeneous and owned by multiple orga-nizations with different usage and cost policies. A large number of self-interested providers and consumers coexist. Resource allocation and reclamation can occur at any time with supply and demand relation varying frequently, and resource usage cannot be fully anticipated. Many issues, such as automatic resource provisioning, multi-objective multi-task scheduling, and workflow scheduling, must be solved [2], [3]. Especially, resource allocation must cater to the nature of decentralization, heterogeneity, and dynamics of cloud. Since economics is concerned with resource alloca-tion among individuals with different objectives in human societies, many economic models have been applied to cloud resource allocation [4].

Although the fixed-price based approaches (for example, commodity market model and posted price model) are used in cloud, they are economically inefficient [5]. In contrast, auction-based approaches are economically efficient and belong to dynamic pricing [5]. Components of a cloud mar-ket can be categorized into buyers (consumers), sellers (pro-viders), and auctioneers. Buyers are charged for their consumed resources based on their valuations, and thus competitions among buyers and also among sellers are encouraged. Auction offers incentive not only for sellers to provide their resources to get profits, but also for buyers to back off when necessary, regulating supply and demand to arrive at market equilibrium. It can cope with diverse and conflicting interests of participants, match dynamic supply and demand, and enable participants to make inde-pendent decisions.

Due to the above highly desirable advantages of the auc-tion, many auction based resource allocation approaches have been proposed (see Section 2 for details), and some cloud service providers have already used auction to sell their resources, for example, spot instances in Amazon’s EC2 [6]. With cloud computing becoming more and more popular and commercial cloud services being widely avail-able, especially as Inter-Cloud, distributed cloud, and OCX are emerging, a cloud market is now really complex and increasingly competitive [7], [8], [9]. In such an environ-ment, a consumer may apply for and a provider may provide various kinds of services and their combinations in terms of resources, making the problem more difficult than focusing on only one of them and calling for combinatorial auction [10]; at the same time, appropriate resources may be available from a number of providers, and a large number of consumers may compete for the same resources, that is, providers and consumers are treated symmetrically with providers submitting asks and consumers submitting bids, calling for double auction [11]; hence, combinatorial double auction should be provided [11]. In addition, the resources demanded by a consumer may be offered by one provider alone, or by multiple providers jointly in order to, for exam-ple, optimize market profits, balance system load, or parti-tion an extra-large task among several providers, which cannot be accommodated by any single provider, especially in Inter-Cloud or distributed cloud. This cannot be sup-ported by [10], [11] and other related solutions. Therefore, we improve the combinatorial double auction further to enable task partitioning among multiple providers.

In each round of the auction, consumers and providers submit their bidding and asking prices. Both price-related factors (for example, budget) and non-price factors (for example, resource usage time-frame) can significantly influ-ence their offers. Not only instant market status (for exam-ple, supply and demand relation) but also historical market experience (for example, historical transaction price) affects their pricing decisions. Thus, a price formation mechanism, which is adaptive to cloud market dynamics, is highly desired. It can be done automatically with agents intro-duced on behalf of participants, not only freeing partici-pants from such complex decision making but also representing them to make rational offers and determine eli-gible transaction relationship. This not only significantly reduces or even eliminates possibility of participants to per-form strategic behaviors, but also helps greatly speed up the auction process.

There inevitably exist some dishonest participants in cloud market. Auction is vulnerable to strategic behaviors of participants, and its trustfulness depends on participants' honesty and their free competition. Trustful design for auc-tion, for example, the randomized auction, can be devised to discourage participants from dishonest behaviors, and its effectiveness of the resultant auction mechanism has been proven by [10], [11], [12], [13], [14]. Reputation [15] is another good way to motivate honest interaction among participants, and participant reputation can significantly affect resource allocation decision. A reputation system can enforce confidence among participants and suppress their strategic behaviors [16]. For example, one hospital wants to outsource its patients' medical records to a cloud. With rep-utation system, the hospital can choose a cloud provider which is honest enough to ensure all medical records kept in privacy with a reasonable service billing, and the cloud provider can ascertain that the hospital is an honest con-sumer which never intentionally damages or misuses cloud resources with an guaranteed paying. In this paper, a novel reputation system is proposed and integrated into IEDA, and then free competition is encouraged and trustful auc-tion is boosted.

At the end of the auction, which provider offers the demanded service to which consumer based on the eligible transaction relationship at the same time whether and how a demanded service should be carried out by multiple providers jointly are decided. A winner determination algo-rithm (WDA) is needed so that those participants, who can not only bring high economic efficiency but also have good reputation, are chosen as winners.

The interactions during the auction, such as bidding, ask-ing, reputation judgment, and winner determination, should be done automatically without human intervention as much as possible to improve participant quality of expe-rience (QoE) [17] and enhance auction trustfulness.

In fact, a comprehensive cloud resource allocation approach is really fundamental in such a challenging cloud market. Oriented to IaaS, we propose IEDA to allocate the following basic resources: processing, memory, storage, net-work bandwidth. In particular, we consider the following basic services: virtual machine service (VMS), computation service (CPS), database service (DBS), and storage service (STS). The major contributions of this paper are as follows.

(1) With integration and necessary improvement of existing techniques, the IEDA system framework is proposed to comprehensively deal with the aforementioned resource allocation challenges, and agents are introduced to enable process automation. (2) An improved combinatorial double auction protocol is devised to enable various kinds of resources traded among multiple consumers and multiple providers, and at the same time enable task partitioning among multiple providers. (3) A price formation mechanism is devised. A BPNN [18] based price prediction algorithm is proposed with instant and historical price and non-price fac-tors considered to make bidding and asking reasonable; a price matching algorithm is proposed to determine eligible transaction relationship among consumers and providers.

(4) A reputation scheme is devised based on the perfor-mance of a participant in the auction to exclude the dishon-est one from the market. (5) The PFA [19] is improved and a WDA is proposed, called WDAPFA. Participants, who can bring the maximum market surplus and surplus strength and have the highest reputations, are preferred to be win-ners. Thus, IEDA is economic efficient and trustful.

The rest of this paper is organized as follows. In Section 2, we review related works and compare our work with them. In Section 3, we provide the IEDA sys-tem framework. In Section 4, we describe the proposed reputation system. In Section 5, we present the improved combinatorial double auction protocol, including tender description, price formation, and winner determination. In Section 6, we describe simulations and performance evaluations. We draw conclusions in Section 7.

2 Related Work

A lot of auction based cloud resource allocation researches have been done. In [20], several resource allocation strate-gies based on a reverse auction model for allocating one type of cloud resource from different providers are investi-gated. In [21], a reverse batch matching auction is proposed for allocating various kinds of cloud resources from differ-ent providers. In [14], a truthful online auction mechanism is proposed for a provider to allocate one type of cloud resource among consumers with heterogeneous demands. In [22], a continuous double auction mechanism is designed to enable consumers and providers to bid and offer one type of cloud resource. In [23], a knowledge based continuous double auction model is proposed to trade one type of cloud resource. In [24], a non-additive negotiation model is pro-posed with multiple objectives considered, by which a pro-vider can efficiently allocate various kinds of resources to a consumer. In [13], cloud resource allocation is done through the auction of different types of VM instances, and a ran-domized combinatorial auction is proposed, which is com-putationally efficient and truthful in expectation with guaranteed social welfare approximation factor. In [10], an online combinatorial auction framework is proposed, which can optimize system efficiency across temporal domain and model dynamic provisioning of heterogeneous VM types. In [12], a suite of truthful and computationally efficient auc-tion mechanisms for cloud resource pricing are proposed with the multi-unit combinatorial auction problem solved. In [11], a combinatorial double auction cloud resource allocation model is proposed, allowing double-sided com-petition and bidding on bundles of items. However, the aforementioned researches cannot deal with transactions of various kinds of resources among multiple consumers and multiple providers with task partitioning among multiple providers enabled, which is solved by our work. In addi-tion, we consider VMS, CPS, DBS, and STS, which usually provided in IaaS cloud; in contrast, [10], [11], [12], [13] only consider VMS.

A lot of price formation mechanisms have been pro-posed. In [5], [14], [21], [24], [25], [26], bidding and asking prices are given directly, not reflecting supply and demand relation. In [27], the asking price is determined by a dynamic pricing scheme based on instant supply and demand information. In [20], the asking price is calculated based on instant capacity information or historical win/loss ratio information. In [22], bidding and asking prices are determined by a two-stage game strategy based on histori-cal price information. In [23], bidding and asking prices are determined by a learning algorithm based on historical price information. In [28], a genetic model based on both price and non-price historical information is proposed to offer suitable price, however, it does not adapt to rapid market changes. Encouraged by the successful application of the artificial neural network (ANN), for example, in stock market forecasting [29], we propose an ANN based price prediction algorithm, and especially due to BPNN's strong self-adaptability, we choose BPNN. We use historical trans-action samples to train BPNN and input instant information to BPNN to predict bidding and asking prices. We further propose a price matching algorithm to determine eligible transaction relationship among consumers and providers. Different from the [5], [14], [20], [21], [22], [23], [24], [25], [26], [27], [28], our method considers instant and historical price and non-price factors, which all influence bidding and asking prices, and at the same time has strong adaptability. It focuses on cloud rather than stock market with factors considered different from those surveyed in [29].

There have been researches on solving WDP. In [20], [22], [23], winners are simply determined by price matching. In [5], winners are determined by bid density greedily during combinatorial auction provision to maximize provider profit. In [21], an immune evolutionary algorithm is applied to solve WDP to maximize the difference between asking price and bidding price. In [14], WDP is solved to maximize the consumer utility gain with an auxiliary pricing function. In [24], a consumer chooses the provider to maximize the defined non-additive utility function. In [25], based on Karush-Kuhn-Tucker (KKT) conditions in the convex opti-mization theory, winners are determined to minimize task executing time. In [26], WDP is solved by the linear mixed integer programming to maximize the total difference between consumer budgets and provider costs. Our goal is different; we try to maximize market surplus, surplus strength and participant reputation with task partitioning among providers under multiple constraints. Due to the NP-hardness of WDP in combinatorial double auction [26], the improved PFA is devised to find the optimal solution.

Participant honesty is necessary to ensure auction trust-fulness. In [5], [20], [21], [22], [23], [24], [25], [26], partici-pants are simply assumed honest. In [10], [11], [12], [13], [14], trustful auction mechanisms are designed without rep-utation system integrated. In [30], a cheat-proof trust model is proposed to make participants be honest to others. The overall trust from A to B has two parts, one is what A directly knows about B, and the other is what the others say about B. Different from [30] and others, our proposed repu-tation system does not aim at any special kind of dishonest behaviors. It derives a participant's honesty from his actual performance in the auction and can effectively deal with participant dishonesty.

3 System Framework

At the outset, we list in Table 1 the abbreviations used throughout this paper.

The system framework of the proposed IEDA consists of five roles: CSP, PA, CSC, CA, and AI, shown in Fig. 1. A CSP provides services in terms of resources. A CSC generates service demands and leases resources. The PA and the CA provide necessary support to CSP and CSC, for example, submitting tender, predicting price, etc. AI is an agent in charge of, for example, collecting tender, running WDA, informing auction result and managing reputation system, etc. AI, PA and CA together relieve CSCs and CSPs of the complicated interaction process for efficient resource alloca-tion. What a CSC and a CSP need to do is to provide the related information, wait for the result, and then evaluate his partner's performance.

The workflow of the proposed IEDA is described as fol-lows and shown in Fig. 2.

At first, when a CSC requests services, he provides the related information to his CA, such as the demanded resour-ces and his own budget, then the CA makes the initial ten-ders; when a CSP can provide services, he provides the related information to his PA, then the PA makes the initial tenders; see Section 5.1 for details.

Second, the CA and the PA use price prediction algo-rithm (see Section 5.2.1) to get bidding and asking prices, put these prices into the corresponding initial tenders, and then submit these updated tenders to AI.

Third, AI collects tenders and performs the price match-ing algorithm (see Section 5.2.2) to determine the eligible transaction relationship among CSCs and CSPs.

TABLE 1
Abbreviations

Abbreviation	Full Name	Abbreviation	Full Name
CSP	Cloud Service Provider	CSC	Cloud Service Consumer
PA	Provider Agent	CA	Consumer Agent
AI	Auction Intermediary	CID	Consumer IDentifier
DS	Demanded Service	BPoDS	Bidding Price of DS
ToDS	Type of DS	SToDS	Starting Time of DS
EToDS	Ending Time of DS	CPUSDS	CPU Speed of DS
MI	Million Instructions	MIPS	MI Per Second
MEM	MEMory	MEMCDS	MEM Capacity of DS
GB	Giga Bytes	STO	STOrage
STOCDS	STO Capacity of DS	NETB	NETwork Bandwidth
NETBDS	NETB of DS	TS	Task Size
DV	Data Volume	MPN	Maximum Partition Number
P&SoDS	Platform and Software of DS	MREPCSP	the required Minimum REPutation to CSP
PID	Provider IDentifier	SS	Supplied Service
PoCPUoSS	Price of CPU of SS	CPUSSS	CPU Speed of SS
PoMEMoSS	Price of MEM of SS	MEMCSS	MEM Capacity of SS
PoSTOoSS	Price of STO of SS	STOCSS	STO Capacity of SS
PoNBoSS	Price of NETB of SS	NETBSS	NETB of SS
SToSS	Starting Time of SS	EToSS	Ending Time of SS
P&SoSS	Platform and Software of SS	MREPCSC	the required Minimum REPutation to CSC
SDR	Supply and Demand Ratio	BUD	BUDget
REP	REPutation	CST	CoST
TF	Time-Frame	APoRU	Asking Price of the Resource per Unit
TLS	TotaL Surplus	TUS	Total Unit Surplus
TRP	Total RePutation	MOO	Multi-Objective Optimization
SOO	Single-Objective Optimization	WSM	Weighted Sum Method
GA	Genetic Algorithm	SPD	SuPeriority Degree
AHP	Analytic Hierarchy Process	LSIA	Local Search Improvement Algorithm
CDA	Continuous Double Auction	SCDA	Stable CDA
TWR	Total Winning Rate	DWR	Dishonest Winning Rate
GM	Greedy Method	Mbps	Million bits per second

Fourthly, AI runs WDAPFA (see Section 5.3) to deter-mine the auction winners and informs CSCs and CSPs of the result. Then, the CSP winners provide services to the CSC winners with payments from the latter to the former.

Finally, after the auction, each CSC/CSP evaluates his partner's performance according to his own QoE on the trans-action and submits his evaluation to AI, and then AI updates CSC and CSP's reputation correspondingly (see Section 4).

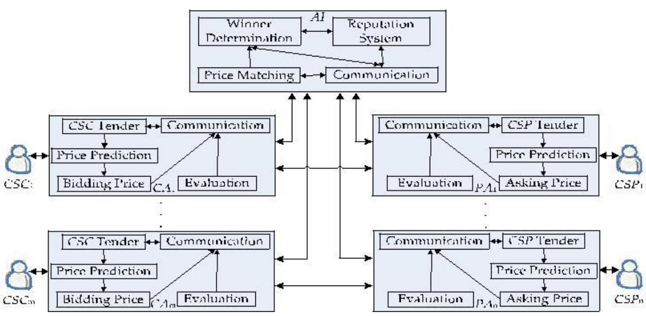

Fig. 1. The IEDA system framework.

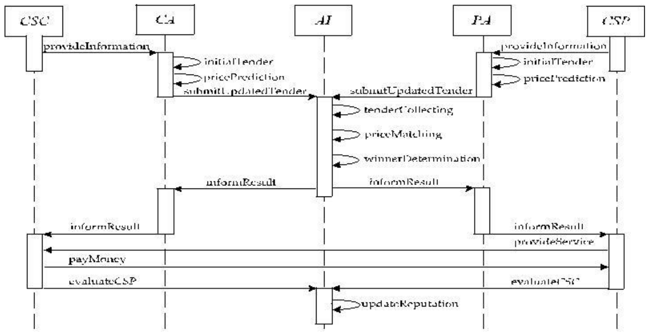

Fig. 2. The IEDA workflow.

From the above IEDA workflow, one possibility for a CSC to tell lies is that he provides a misleading budget on his demanded service at the beginning of the auction. If his provided budget was intentionally increased, even if he won the auction, he would pay a higher bid than what he should pay because of the competition, which is not what he expects. If his provided budget was inten-tionally reduced, he would lose the winning opportunity in the competition due to his lower bids. Another possi-bility is that he dishonestly evaluates his partner's perfor-mance after the auction, and his dishonest behavior will be punished by our proposed reputation system. There are no other chances for a CSC to tell lies during the auc-tion due to IEDA process automation. For a CSP, he can provide a misleading cost on his offered service or dis-honestly evaluate his partner's performance; the situation is similar to that of a CSC. Therefore, with price forma-tion, reputation, WDAPFA and agent integrated, IEDA can promote the economic efficiency and the trustfulness of the auction.

The following simple illustration example will be used throughout this paper to help clarify the discussion: there are four CSCs and six CSPs, all CSPs only provide STS and all CSCs only consume STS.

4 Reputation System

4.1 Basic Idea

Our proposed reputation system obeys the following intu-itions. (I1) If a participant takes part in auction frequently and his turnover is high, his reputation should be high, and vice versa. (I2) If a participant gets high evaluations on QoEs from his trading partners, his reputation should be high, and vice versa. (I3) If a participant evaluates his trad-ing partners objectively, that is, he is honest to his partners, his evaluations on his partners should be creditworthy, and vice versa. During the auction, AI makes participants with good reputations winning chance high. After the auction, each participant evaluates his partners' actual performances based on his QoEs on the transactions, and AI updates these participants' reputations. If a participant's evaluation value is much different from his partner's previous reputation, AI considers him to be slightly or seriously dishonest accord-ing to how big the difference is. If a participant has been considered slightly dishonest the prescribed consecutive times or seriously dishonest once, he is excluded from the market. Thus, the honesty is encouraged and the dishonesty is punished.

4.2 Reputation Computation

The CSP's reputation is computed as follows:

$$p\ rep_{jk} ¼ de\eth Dt^{k}_{k\_1}Þ \frac{total\ tr_{j\eth k\_1Þ}}{total\ tr_{jk}} \_ p\ rep_{j\eth k\_1Þ}$$

$$\_ \quad 1 \quad de\ Dt^{K} \quad \frac{\sum_i RPM_{ijk} \quad price_{ijk} \_ QoE_{ijk} \_ CR_{ijk}}{total\ \bar{t}r_{jk} \_ total\ tr_{j\eth k\_1Þ}}$$

$$Þ \ \_ \quad \eth \quad k\_1Þ \ \_ P \qquad (1)$$

$$de\ Dt^{K}_{k\_1}\ Þ ¼ \begin{cases} 1 & Dt^{k}_{k\_1} < t_{min} \text{ and } k > 1 \\ \frac{t_{max\_}D t^{k}_{k\_1}}{t_{max\_}t_{min}} & t_{min} \quad Dt^{K}_{k\_1} \quad t_{max} \text{ and } k > 1 \end{cases} \qquad (2)$$

$$CR_{ijk} ¼ \frac{N_{ijk\_}N_{ijk\_}}{N_{ijk}}: \qquad (3)$$

Here, $p\ rep_{jk}$ is the CSP_j's reputation after the kth auction. $total\ tr_{jk}$ is the accumulated turnovers of CSP_j after the kth auction. RPM_{ijk}, $price_{ijk}$ and RPM_{ijk} _ $price_{ijk}$ are trading volume, transaction price and turnover between CSP_j and CSC_i in the kth auction respectively. QoE_{ijk} 2 ½0; 1& is the evaluation of CSC_i on CSP_j's performance in the kth auction which reflects his QoE to CSP_j, where 1 denotes complete satisfaction and 0 entire dissatisfaction, and CSC_i submits it to AI after the auction. $de\eth Dt^{k}_{k-1}Þ$, defined in (2), is the time decay coefficient to reflect the decrement degree of reputa-tion with time passing, where $Dt^{k}_{k\_1}$ is the time interval between the kth auction and the (k-1)th auction which CSP_j took part in, t_{min} and t_{max} are the lower and upper experi-enced threshold respectively. CR_{ijk} is creditworthiness degree of CSC_i to CSP_j in the kth auction and defined in (3). If Diff ¼ $\bar{}QoE_{ijk}$ _ $p\ rep_{j\eth k\_1Þ}$, that is, the difference between CSP_j's current evaluation value on CSP_j and CSP_j's previous reputation, is too big, AI considers that the CSC_i's evaluation on CSP_j is not objective, and whether CSC_i is slightly or seriously dishonest depends on how big Diff is. N_{ijk} is the total times of CSC_i evaluating CSP_j up to the kth auction, and $N_{ijk\_}$ is the times which CSC_i is considered to be dishonest.

The reputation value is between 0 and 1. Initially, all participants' reputation values, for example, $p\ rep_{j0}$, are set to be 0.5, that is, their default honesties are "average", because AI has no experience with them and cannot evaluate their honesties before they enter the cloud market.

The CSC's reputation computation is similar to CSP's. Due to limited space, we do not describe it in detail.

For the example in Section 3, assume that in the 10th auction CSC_1 and CSC_2 have transactions with CSP_1 and have the following data: $total\ tr_{1;9}$ ¼ \$56, $p\ rep_{1;9}$ ¼ 0:6, $RPM_{1;1;10}$ ¼ 0:5; $RPM_{2;1;10}$ ¼ 0:5; $QoE_{1;1;10}$ ¼ 0:5, $QoE_{2;1;10}$ ¼ 0:6; $price_{1;1;10}$ ¼ \$9:3; $price_{2;1;10}$ ¼\$8:6; Dt^{10}_{9} ¼400 hour, t_{min} ¼ 300 hour; t_{max} ¼ 800 hour; $N_{1;1;10}$ ¼ 10; $N_{1;1;10\_}$ ¼ 0, $N_{2;1;10}$ ¼ 10; $N_{2;1;10\_}$ ¼1, we get $de\eth Dt^{10}_{9}Þ$ ¼ 0:8, $total\ tr_{1;10}$ ¼ $total\ tr_{1;9}$ Þ $RPM_{1;1;10}$ _ $price_{1;1;10}$ Þ $RPM_{2;1;10}$ _ $price_{2;1;10}$ ¼ \$65:0, $CR_{1;1;10}$ ¼ 1, $CR_{2;1;10}$ ¼ 0:9, then after the 10th auction, the reputation of CSP_1 is updated as $p\ rep_{1;10}$ ¼ 0:5 by (1). By the way, in this paper, just for simplicity, when we do calculation for the example in Section 3, we only keep one decimal place in the calculation result.

TABLE 2
CSC Tenders

Attribute \ Service	VMS	CPS	DBS	STS
CID	p	p	p	p
BPoDS	p	p	p	p
ToDS	p	p	p	p
SToDS	p	p	p	p
EToDS	p	p	p	p
CPUSDS	p	p	_	_
MEMCDS	p	_	p	_
STOCDS	p		p	p
NETBDS	p	_	p	_
TS	_	p	_	_
DV	_	_	p	_
MPN	_	p	p	p
P&SoDS	p	p	p	p
MREPCSP	p	p	p	p

5 Combinatorial Double Auction Protocol

5.1 Tender Description

5.1.1 CSC Tender

The attributes, which a CSC tender could have, are as fol-lows: CID, to identify a CSC uniquely in the cloud market; BPoDS, to denote a CSC's bidding price to the demanded service with unit \$; ToDS, to denote the type of service that a CSC demands; SToDS, to denote when the CSC demanded service begins; EToDS, to denote when the CSC demanded service ends; CPUSDS, to denote the CPU speed asked by the CSC demanded service with unit MIPS; MEMCDS, to denote the memory capacity asked by the CSC demanded service with unit GB; STOCDS, to denote the storage capac-ity asked by the CSC demanded service with unit GB; NETBDS, to denote the network bandwidth asked by the CSC demanded service with unit Mbps; TS, to denote the size of the executed task asked by the CSC demanded ser-vice with unit MI; DV, to denote the volume of the proc-essed data asked by the CSC demanded service with unit GB; MPN, to denote the maximum allowed number of CSPs to execute the CSC demanded service jointly (A task could be partitioned to be executed by several CSPs; however, it should not be partitioned into too many tiny parts in order to avoid too much communication and coordination over-head. A CSC can set MPN to do so.); P&SoDS, to denote the specific platform and software environment needed by the CSC demanded service; MREPCSP, to denote the required minimum reputation to CSP.

In fact, the tenders of a CSC to VMS, CPS, DBS and STS are subsets of the above attributes, listed in Table 2, where 'p' means that the tender has the corresponding attribute and '_' means no.

5.1.2 CSP Tender

The attributes, which a CSP tender could have, are as fol-lows: PID, to identify a CSP uniquely in the cloud market; PoCPUoSS, to denote the CPU price of the CSP supplied ser-vice with unit \$/(MIPS _ hour); CPUSSS, to denote the CPU speed of the CSP supplied service with unit MIPS; PoMEMoSS, to denote the memory price of the CSP

TABLE 3
CSP Tenders

Service / Attribute	VMS	CPS	DBS	STS
PID	p	p	p	p
PoCPUoSS	p	p	–	–
CPUSS	p	p	–	–
PoMEMoSS	p	–	p	–
MEMCSS	p	–	p	–
PoSTOoSS	p	–	p	p
STOCSS	p	–	p	p
PoNBoSS	p	–	p	–
NETBSS	p	–	p	–
SToSS	p	p	p	p
EToSS	p	p	p	p
P&SoSS	p	p	p	p
MREPCSC	p	p	p	p

supplied service with unit $/(GB _ hour); MEMCSS, to denote the memory capacity of the CSP supplied service with unit GB; PoSTOoSS, to denote the storage price of the CSP supplied service with unit $/(GB _ hour); STOCSS, to denote the storage capacity of the CSP supplied service with unit GB; PoNBoSS, to denote the network bandwidth price of the CSP supplied service with unit $/(Mbps _ hour); NETBSS, to denote the network bandwidth of the CSP supplied service with unit Mbps; SToSS, to denote when the CSP supplied service begins; EToSS, to denote when the CSP supplied service ends; P&SoSS, to denote the platform and software environment of the CSP supplied service; MREPCSC, to denote the required minimum repu-tation to CSC. The tenders of a CSP are listed in Table 3.

For the example in Section 3, the tenders of four CSCs and six CSPs are labeled as c_exp1-4 and p_exp1-6 respec-tively and given as follows.

c_exp1: {"CSC_1",$3.1,STS,2015-6-5-6:00,2015-6-10-9:00,,, 100GB,,,,3, {Linux},0.4},

c_exp2: {"CSC_2",$4.0,STS,2015-6-2-5:00,2015-6-6-00:00,,, 120GB,,,,3, {Linux},0.5},

c_exp3: {"CSC_3",$7.0,STS,2015-6-8-7:00,2015-6-11-10:00,,, 200GB,,,,4, {Linux},0.5},

c_exp4: {"CSC_4",$9.0,STS,2015-6-12–8:00,2015-6-17-9:00,,, 150GB,,,,3, {Linux},0.4}.

p_exp1: {"CSP_1",,,,,$0.00026/(GB_hour),120GB,,,2015-6-1-0:00,2015-6-20-0:00,{Linux},0.7},

p_exp2: {"CSP_2",,,,,$0.00031/(GB_hour),100GB,,,2015-6-1-1:00,2015-6-21-21:00, {Linux},0.6},

p_exp3: {"CSP_3",,,,,$0.00029/(GB_hour),120GB,,,2015-6-1-2:00,2015-6-17-20:00, {Linux},0.6},

p_exp4: {"CSP_4",,,,,$0.00037/(GB_hour),95GB,,,2015-6-1-3:00,2015-6-20-18:00, {Linux},0.8},

p_exp5: {"CSP_5",,,,,$0.00045/(GB_hour),80GB,,,2015-6-1-4:00,2015-6-21-17:00, {Linux},0.5},

p_exp6: {"CSP_6",,,,,$0.00055/(GB_hour),150GB,,,2015-6-1-5:00,2015-6-20-15:00, {Linux},0.7}.

For example, c_exp1 means that CSC_1 is willing to pay $3.1 for 100 GB storage between 2015-6-5-6:00 and 2015-6-10-9:00, the maximum allowed number of CSPs is 3, the supporting platform and software environment is Linux, and the CSP's reputation is at least 0.4; p_exp1 represents that CSP_1's storage unit price is $0.00026/(GB_hour), it can provide 120 GB storage between 2015-6-1-0:00 and 2015-6-20-0:00, the supporting platform and software environment is Linux, and the CSC reputation is at least 0.7.

5.2 Price Formation

5.2.1 Price Prediction

A price prediction algorithm is proposed for CAs and PAs respectively. Just for simplicity, we only consider SDR, BUD, REP, TF for CSCs, and SDR, CST, REP, TF for CSPs. Among them, SDR, BUD and CST are price factors and others are none-price ones.

If SDR is lower, the CSC/CSP bidding/asking price should be higher, otherwise lower. BUD is the CSC's budget of the demanded service and is the upper bound of his bid-ding price. REP is the lowest reputation that a CSC/CSP asks CSPs/CSCs to have. TF is the period during which the CSC's demanded service is used; if it is in peak period or prime time, the CSC/CSP bidding/asking price should be higher, otherwise lower. CST is the CSP's cost of the sup-plied service and is the lower bound of his asking price.

In this paper, the linear exponential smoothing method [31] is used to estimate the expected SDR of each type of ser-vice in the kth price matching, denoted as $SDR^{P;TS}_k$.

$$SDR_k^{P;TS} = c\,SDR_{k-1}^{A;TS} + (1 - c)\,SDR_{k-1}^{P;TS}. \quad (4)$$

Here, c is the smoothing coefficient, $0 \le c \le 1$; $SDR^{P;TS}_{k-1}$ and $SDR^{A;TS}_{k-1}$ are predicted and actual SDR of TS service in the (k _ 1)th price matching, TS represents VMS, CPS, DBS or STS. We assume that the initial supply and demand are in balance with $SDR^{P;TS}_0 = SDR^{A;TS}_0 = 1$. AI calculates the actual SDR of each type of resource by (5) in the kth price matching and publishes it after the kth price matching.

$$SDR_k^{A;TR} = \frac{\sum_{j=1}^{n} SUP_{j;k}^{TR}}{\sum_{i=1}^{m} DEM_{i;k}^{TR}}. \quad (5)$$

Here, $SUP_{j;k}^{TR}$ and $DEM_{i;k}^{TR}$ are the amount of TR resource supplied by CSP_j and the amount of TR resource demanded by CSC_i in the kth price matching, TR represents CPU, MEM, STO and NETB. Then, CSC obtains the actual SDR of each type of service in the kth price matching by (6)-(9).

$$SDR^{A;VMS}_k = \min(SDR^{A;CPU}_k; SDR^{A;MEM}_k; SDR^{A;STO}_k; SDR^{A;NETB}_k) \quad (6)$$

$$SDR_k^{A;CPS} = SDR_k^{A;CPU} \quad (7)$$

$$SDR_k^{A;DBS} = \min(SDR_k^{A;MEM}; SDR_k^{A;STO}; SDR_k^{A;NETB}) \quad (8)$$

$$SDR_k^{A;STS} = SDR_k^{A;STO}. \quad (9)$$

VMS is involved in CPU, MEM, STO and NETB usage, and we use the minimum of their SDRs to be VMS's SDR. The SDRs of CPS, DBS and STS are defined similarly.

We use BPNN to predict price based on historical sam-ples if sufficient, which accumulated from previous

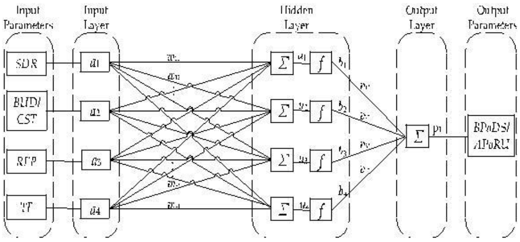

Fig. 3. The BPNN structure.

auctions (see line 24 of WDAPFA and the last paragraph in Section 5.3.2), and instant input information. The BPNN has three layers, shown in Fig. 3. Its hidden layer contains four nodes corresponding to the four considered factors and f is the log_sigmoid function as follows.

$$f\eth x Þ ¼ \frac{1}{1 þ e^{-x}} : \qquad (10)$$

The CA's BPNN based bidding price prediction algorithm is described as follows.

At first, Algorithm1 judges whether BPNN has been trained with sufficient samples (line 3): if so, the bidding price is predicted by BPNN (lines 11-22) and the supply-and-demand relation is adjusted between two successive calls of Algorithm1 (line 12); otherwise, it is determined directly without BPNN (lines 4-9), because insufficient training often lead to bad prediction results from BPNN.

Algorithm 1. Price_prediction

Input: SDR, BUD, REP, TF, sample-base, Label (indicating whether this is the first call to Algorithm1, 1 means yes, 0 means no)

Output: BPoDS

1: Set MNoS be the required minimum number of samples in sample-base to train BPNN;
2: Set N be the number of samples recorded in sample-base; 3: if N < MNoS then
4: if Label ¼ ¼ 1 then
5: Set BPoDS be a random number between 0 and BUD;
6: else
7: Get DBP randomly from the uniform distribution within the interval [0, BUD-BPoDS]; /¯DBP is the bid-ding price adjustment amplitude.¯/
8: BPoDS ¼ BPoDS þ DBP; 9: end if
10: else
11: if Label ¼ ¼ 0 then
12: Update SDR by (4); 13: end if
14: a_1 ¼ SDR, a_2 ¼ BUD, a_3 ¼ REP , a_4 ¼ TF ;
15: for j 2 f1; 2; 3; 4g do
16: u_j ¼ $\sum_{i=1}^{4} w_{ij} a_i$;
17: end for
18: for j 2 f1; 2; 3; 4g do
19: b_j ¼ fðujÞ;
20: end for
21: p_1 ¼ $\sum_{i=1}^{4} v_{i1} b_i$;
22: BPoDS ¼ p_1;
23: end if
24: return BPoDS;

MNoS should satisfy the following condition:

$$MNoS ¼ O\eth W=e Þ: \qquad (11)$$

Here, e is the permitted output error; W is the total num-ber of BPNN's free parameters (i.e., synapse weights and bias values) and calculated as follows [32].

$$W ¼ M \_ L_1 þ L_1 þ d: \qquad (12)$$

Here, M, L_1 and d are the number of the input, hidden and output layer nodes respectively.

The PA's price prediction algorithm is almost the same as the CA's. However, its input and output are changed to {SDR, CST, REP, TF, sample-base, Label} and APoRU respec-tively, line 7 is changed to "Get DAP randomly from the uniform distribution within the interval [0, APoRU-CST]; /¯DAP is the asking price adjustment amplitude.¯/", and line 8 is changed to "APoRU ¼ APoRU _ DAP;".

The CA's and PA's sample formats are {SDR, BUD, REP, TF, BPoDS} and {SDR, CST, REP, TF, APoRU} respectively. Here, BPoDS is the final bidding price of a CSC to the demanded service, and APoRU is the final ask-ing price of a CSP to the resource per unit. After each auction, we get samples (see line 24 of WDAPFA and the last paragraph in Section 5.3.2) and we can use them to train BPNN offline. After trained by MNoS samples, BPNN can be used to predict price.

5.2.2 Price Matching

What a CSP offers is the resource unit price, thus AI needs to get total asking price of a CSP to the CSC demanded service and match it with the CSC's bidding price to find the eligible transaction relationship among CSCs and CSPs. For VMS, CPS, DBS, and STS, the total asking price of CSP_j to CSC_i is calculated in (13)-(16) respectively.

$$\begin{aligned} ask\text{-}price_{ij} ¼ \eth & CPUSDS_i \_ PoCPUoSS_j \\ & þ MEMCDS_i \_ PoMEMoSS_j \qquad (13) \\ & þ STOCDS_i \_ PoSTOoSS_j \\ & þ NETBDS_i \_ PoNBoSS_j Þ \_ length_i \end{aligned}$$

$$ask\text{-}price_{ij} ¼ TS_i \_ PoCPUoSS_j \qquad (14)$$

$$\begin{aligned} ask\text{-}price_{ij} ¼ \eth & MEMCDS_i \_ PoMEMoSS_j þ STOCDS_i \\ & \_ PoSTOoSS_j Þ \_ length_i þ DV_i \_ PoNBoSS_j \end{aligned} \qquad (15)$$

$$ask\text{-}price_{ij} ¼ STOCDS_i \_ PoSTOoSS_j \_ length_i \qquad (16)$$

$$length_i ¼ EToDS_i \_ SToDS_i: \qquad (17)$$

For the example in Section 3, assume that CSC_1 demands STS from CSP_1. Based on c_exp1 and p_exp1, the total ask-ing price is calculated as follows.

$$ask\ price_{11} ¼ 100\ GB \_ \$0:00026=\eth GB \_ hourÞ \_ 123\ hour ¼ \$3:2: \qquad (18)$$

The price matching algorithm is described as follows.

In Algorithm2, price matching is done MRN rounds to prompt trading among CSCs and CSPs as much as possible. After each round, a CSC/CSP, whose bidding/asking price does not match any CSP/CSC asking/bidding price, is notified to re-bid/re-ask (line 21/line 26). Here, re-bid is done by Algorithm1, re-ask is done by PA's price prediction algorithm and (13)-(16).

Algorithm 2. Price_matching

Input: MRN (the maximum round number), TCT (the tender collection time at each round)

Output: $ask\_price_{m\_n}$

```
1:  Set m and n be the number of CSCs and CSPs respectively;
2:  Initialize all elements in matrix flag_m_n to be 0; /* flag_ij indicates
    whether price matching between CSC_i and CSC_j succeeds, 1
    means yes, 0 means no.*/
3:  k = 1;
4:  while (k ≤ MRN) do
5:    Collect tenders from CSCs and CSPs until TCT timeouts;
6:    for i ∈ {1, 2, . . . , m} do
7:      for j ∈ {1, 2, . . . , n} do
8:        switch (type of CSC_i's tender)
9:          case VMS:
              Calculate ask_price_ij by (13); break;
10:         case CPS:
              Calculate ask_price_ij by (14); break;
11:         case DBS:
              Calculate ask price_ij by (15); break;
12:         case STS:
              Calculate ask price_ij by (16);
13:       end switch
14:       if BPoDS_i ≥ ask price_ij then
15:         flag_ij = 1;
16:       end if
17:     end for
18:   end for
19:   for i ∈ {1, 2, . . . :, m} do
20:     if (all elements of ith row in flag_m_n are 0) then
21:       Notify the CSC_i to re-bid;
22:     end if
23:   end for
24:   for j ∈ {1, 2, . . . , n} do
25:     if (all elements of jth column in flag_m_n are 0) then
26:       Notify the CSP_j to re-ask;
27:     end if
28:   end for
29:   k = k + 1;
30: end while
31: return ask_price_m_n;
```

For the example in Section 3, we get $initial\_ask\_price_{4\_6}$ and $initial\_flag_{4\_6}$, e.g., $initial\_ask\_price_{11} = \$3.2$, BPoDS of CSC_1 is \$3.1, then $initial\_flag_{11} = 0$. From $initial\_flag_{4\_6}$, we can see that the CSC_1/CSP_6's bidding/ asking price does not match any CSP/CSC's asking/ bidding price, then AI notifies them to try again. Finally, we get $BPoDS_1 = \$5.0$, $BPoDS_2 = \$4.0$; $BPoDS_3 = \$7.0$, $BPoDS_4 = \$9.0$, $final\_ask\_price_{4\_6}$ and $final\_flag_{4\_6}$. See Fig. 4 for details.

$$\underset{(a)}{\overset{initial_ask_price_{4_6}}{\begin{bmatrix} 3.2 & 3.8 & 3.6 & 4.6 & 5.5 & 6.8 \\ 2.8 & 3.4 & 3.2 & 4.0 & 4.9 & 6.0 \\ 3.9 & 4.7 & 4.4 & 5.6 & 6.8 & 8.3 \\ 4.7 & 5.6 & 5.3 & 6.7 & 8.2 & 10.0 \end{bmatrix}}}
\underset{(b)}{\overset{initial_flag_{4_6}}{\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 \end{bmatrix}}}
\underset{(c)}{\overset{final_ask_price_{4_6}}{\begin{bmatrix} 3.2 & 3.8 & 3.6 & 4.6 & 5.5 & 4.9 \\ 2.8 & 3.4 & 3.2 & 4.0 & 4.9 & 4.4 \\ 3.9 & 4.7 & 4.4 & 5.6 & 6.8 & 6.0 \\ 4.7 & 5.6 & 5.3 & 6.7 & 8.2 & 7.3 \end{bmatrix}}}
\underset{(d)}{\overset{final_flag_{4_6}}{\begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}}}$$

Fig. 4. Matrices in the example.

5.3 Winner Determination

5.3.1 Problem Formulation

In this paper, winner determination is used to get the opti-mal solution in which the trading volume at the transaction price between each CSC and his partner CSP is decided. As a result, WDP becomes one to find an optimal partition matrix RPM_{m_n}, of which RPM_{ij} is the proportion of the demanded service that CSC_i receives from CSP_j,

$RPM_{ij} \in [0, 1], 1 \le i \le m, 1 \le j \le n.$

The CSC bidding price should not be less than the CSP asking price in one transaction, and the balance between them is called market surplus [33]. We call the balance between the total bidding prices of all CSCs and the total asking prices of their partner CSPs, the balance between their total unit time prices, and their total repu-tations as TLS, TUS, and TRP, calculated in (19)-(21) respectively.

$$TLS(RPM_{m\_n}) = \sum_{i=1}^{m}\sum_{j=1}^{n} (RPM_{ij} \times BPoDS_i) - \sum_{i=1}^{m}\sum_{j=1}^{n} (RPM_{ij} \times ask\ price_{ij}) \tag{19}$$

$$TUS(RPM_{m\_n}) = \sum_{i=1}^{m}\sum_{j=1}^{n} RPM_{ij} \times \frac{BPoDS_i}{length_i} - \sum_{i=1}^{m}\sum_{j=1}^{n} RPM_{ij} \times \frac{ask\_price_{ij}}{} \tag{20}$$

$$TRP(RPM_{m\_n}) = \sum_{i=1}^{m}\sum_{j=1}^{n} (RPM_{ij} \times c\_rep_i) + \sum_{j=1}^{n}\sum_{i=1}^{m} (RPM_{ij} \times p\_rep_j). \tag{21}$$

Here, c_rep_i and p_rep_j are CSC_i's and CSP_j's reputation respectively.

We want to optimize TLS, TUS and TRP so that not only gross surplus but also surplus strength are tried to be max-imized at the same time honest participants are encour-aged with high economic efficiency and trustfulness attained. It is a MOO problem and can be dealt with by, e.g., GA [28] and WSM [34]. GA can be effective regardless of the nature of the objective functions and constraints, but has relatively high computational expense. WSM is com-putationally efficient and easy-to-use, but needs to deter-mine the relative importance of multiple objectives. Due to its computation efficiency, we use WSM to convert MOO into SOO and define SPD of $RPM_m{}^{a}{}_{-n}$ to $RPM_m{}^{b}{}_{-n}$ as follows.

$$V^{\alpha} = \alpha \frac{TLS(RPM^{\alpha}_{m\_n}) - TLS(RPM^{\beta}_{m\_n})}{TLS(RPM^{\beta}_{m\_n})} + \beta \frac{TUS(RPM^{\alpha}_{m\_n}) - TUS(RPM^{\beta}_{m\_n})}{TUS(RPM^{\beta}_{m\_n})} + \gamma \frac{TRP(RPM^{\alpha}_{m\_n}) - TRP(RPM^{\beta}_{m\_n})}{TRP(RPM^{\beta}_{m\_n})}: \quad (22)$$

Here, α, β and γ are the weights of three objectives and their values can be determined, e.g., by experiments or by AHP [35], $\alpha + \beta + \gamma = 1$, $\alpha > 0$, $\beta > 0$, $\gamma > 0$. We determine them by experiments (see Section 6.2.1).

Assume that we have a list of partition matrices, $RPM^1_{m\_n}$, $RPM^2_{m\_n}$; . . ., $RPM^{LT}_{m\_n}$, and LT is the list length.

We select $RPM^{\#}_{m\_n}$ as the benchmark and get $V^i_{\#}$, $i = 1; \ldots; LT$. Denote the biggest $V^i_{\#}$ as $V^{BT}_{\#}$, then $RPM^{BT}_{m\_n}$ corresponding to $V^{BT}_{\#}$ is the optimal partition matrix, $BT \in \{1; 2; \ldots; LT\}$.

We formulate WDP in this paper as follows.

$$\text{maximize} V^i_{\#} \quad (23)$$

s.t.

$$BPoDS_i \geq ask\ price_{ij};\ \forall RPM_{ij} \neq 0 \quad (24)$$

$$p\ rep_j \geq MREPCSP_i;\ \forall RPM_{ij} \neq 0 \quad (25)$$

$$c\ rep_i \geq MREPCSC_j;\ \forall RPM_{ij} \neq 0 \quad (26)$$

$$\sum_{j=1}^{n} RPM_{ij} \leq MPN_i;\ \forall i \quad (27)$$

$$RPM_{ij} \in [\varepsilon; 1] \cup \{0\};\ \forall i;\ \forall j \quad (28)$$

$$\sum^{m} RPM_{ij} \in \{0; 1\};\ \forall i \quad (29)$$

$$SToDS_i \geq SToSS_j;\ \forall RPM_{ij} \neq 0 \quad (30)$$

$$EToDS_i \leq EToSS_j;\ \forall RPM_{ij} \neq 0 \quad (31)$$

$$P\&SoDS_i \subseteq P\&SoSS_j;\ \forall RPM_{ij} \neq 0 \quad (32)$$

$$CPUSSS_j \geq CPUSDS_i;\ \forall RPM_{ij} \neq 0 \quad (33)$$

$$MEMCSS_j \geq MEMCDS_i;\ \forall RPM_{ij} \neq 0 \quad (34)$$

$$STOCSS_j \geq STOCDS_i;\ \forall RPM_{ij} \neq 0 \quad (35)$$

$$NETBSS_j \geq NETBDS_i;\ \forall RPM_{ij} \neq 0: \quad (36)$$

Here, (24) says that the CSC bidding price cannot be lower than the CSP asking price. (25), (26) require that the CSP/CSC's reputation cannot be lower than the CSC/CSP requirement. (27) means that at most MPN_i CSPs are allowed to carry out the CSC demanded service jointly. (28) means that the partition granularity cannot be too fine and ε is its lower bound. (29) requires that the CSC demanded service must be provided by CSP(s) completely or none. (30)-(36) mean that the CSP service starting time, service ending time, platform and software environment, CPU, memory, storage and network bandwidth must satisfy the corresponding CSC requirement. Among them, (24)-(32) are common constraints satisfied by all types of services. However, each type of service has specific constraints to be satisfied, in particular, VMS, CPS, DBS, STS need to satisfy (33)-(36), (33), (34)-(36), (35) respectively.

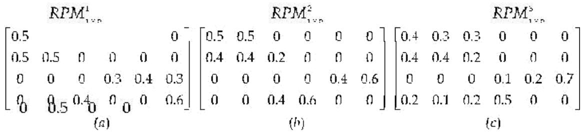

Fig. 5. Initial seeds

5.3.2 Winner Determination Algorithm

We improve PFA to solve WDP. PFA is bio-inspired and seeds correspond to problem solutions. When sown in field, seeds which fall into places with the favorable conditions tend to grow to become the healthy plants. Such plants are capable of producing more seeds than less fortunate ones. The healthiest plant of the population corresponds to the optimum which can be determined by a fitness function. A high plant density would increase pollination chance, thus the higher the plant density, the more likely the chance of proper pollination. Then, the seeds of these plants are scattered in field and become new plants, and the cycle con-tinues. PFA has strong global search ability and low compu-tation overhead. It does not depend heavily on initial values. However, its local search ability is not very good, thus we improve it with the simplex algorithm (SA) [34]. In addition, we devise a customized seed refinement procedure to make the solution feasible to satisfy (24)-(36). If a seed corresponds to a feasible solution, it is healthy, otherwise it is ill. The key operations of WDAPFA are described as follows.

(1) Sowing. The INoS seeds are generated randomly as initial solutions and INoS is population size. Each seed is a matrix RPM_{m_n} and RPM_{ij} is uniformly distributed within [0, 1]. One seed is randomly chosen as the benchmark. The SPD of each seed to the benchmark is defined as its fitness value to measure its health.

Corresponding to the example in Section 3, we generate initial seeds $RPM^1_{4\_6}$, $RPM^2_{4\_6}$, and $RPM^3_{4\_6}$, shown in Fig. 5. According to (24)-(32) and (35), the former two are healthy and the latter one is ill.

(2) Seed refinement. We can refine an ill seed to a healthy one so that an infeasible solution is changed into a feasible one. If price, reputation, time-frame, platform and software, or capacity constraints ((24), (25), (26), (30), (31), (32), or (33)-(36)) violated, we just set the corresponding RPM_{ij} be 0. If partition number, partition granularity, or service constraints ((27), (28), or (29)) violated, the refinement is complex.

(a) Partition number constraint violation handling.

If there exists $\sum_{j=1}^{n} RPM_{ij} > MPN_i$, that is, the parti-tion number is too

Step 1: $i = 1$.

Step 2: If $i > m$, go to Step 7.

Step 3: If $\sum_{j=1}^{n} RPM_{ij} \leq MPN_i$, go to Step6.

Step 4:

elements $RPM_{i;min}$ and $RPM_{i;smin}$; $RPM_{i;smin} = RPM_{i;smin} + RPM_{i;min}$; $RPM_{i;min} = 0$.

$$RPM^{3}_{4\times 6} = \begin{bmatrix} 0.4 & 0.3 & 0.3 & 0 & 0 & 0 \\ 0.4 & 0.4 & 0.2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.1 & 0.2 & 0.7 \\ 0.2 & 0 & 0.3 & 0.5 & 0 & 0 \end{bmatrix}$$

Fig. 6. Refined seed.

Step 5: If $\sum_{j¼1}^{n}$ RPMij > MPNi, go to Step 4.

Step 6: i ¼ i þ 1, go to Step 2.

Step 7: Refinement ends.

We merge the proportion of CSP_{min} into that of CSP_{smin} and CSP_{min} is no longer a provider in Step 4, then the parti-tion number decreases by 1. This iteration continues until (27) is satisfied.

(b) Partition granularity constraint violation handling.

If there exists RPM_{ij} < ", that is, there exists too fine par-tition, we do the following refinement.

Step 1: i ¼ 1.

Step 2: If i > m, go to Step 8.

Step 3: Find $RPM_{i;min}$.

Step 4: If $RPM_{i;min}$ _ ", go to Step 7.

Step 5: Find $RPM_{i;smin}$.

Step 6: $RPM_{i;smin}$ ¼ $RPM_{i;smin}$ þ $RPM_{i;min}$, $RPM_{i;min}$ ¼ 0, go to Step 3.

Step 7: i ¼ i þ 1, go to Step 2.

Step 8: Refinement ends.

We merge the proportion of CSP_{min} into that of CSP_{smin} and CSP_{min} is no longer a provider in Step 6. Then, the par-tition granularity minimum is increased. This iteration con-tinues until (28) is satisfied.

(c) Service constraint violation handling.

If there exist $\sum_{j¼1}^{n}$ RPMij ¼6 1 and $\sum_{j¼1}^{n}$ RPMij ¼6 0, that is, the CSC demanded service cannot be provided properly, we do the following refinement.

Step 1: i ¼ 1.

Step 2: If i > m, go to Step 8.

Step 3: If $\sum_{j¼1}^{n}$ RPMij ¼ 1 or $\sum_{j¼1}^{n}$ RPMij ¼ 0, go to Step 7.

Step 4: Find the maximum element $RPM_{i;max}$.

Step 5: Obtain ...

Step 6: If Temp _ 1, $RPM_{i;max}$ ¼ 1 _ Temp, go to Step 7, else $RPM_{i;max}$ ¼ 0, go to Step 3.

Step 7: i ¼ i þ 1, go to Step 2.

Step 8: Refinement ends.

We adjust the proportion of CSP_{max} in Step 6 to make $\sum_{j¼1}^{n}$ RPMij ¼ 1 or $\sum_{j¼1}^{n}$ RPMij ¼ 0 until (29) is satisfied.

By refinement, $RPM^{3}_{4\_6}$ becomes $RPM^{3^0}_{4\_6}$, see Fig. 6.

(3) Selection. After seeds are sown into the field and plants are produced, only the very healthy plants are selected, so that plants do not grow explosively. In this paper, we sort all plants in descending order based on their fitness values and only select the first INoS plants as new population.

(4) Seeding. Each plant P produces a number of seeds based on its fitness and the number is calculated as follows.

$$S_P ¼ \& q_{max} \frac{V^P_{\#} \_ V_{WT\#}}{V_{BT\#} \_ V_{WT\#}} , : \quad (37)$$

Here, q_{max} is the number of seeds produced by the plant with the highest fitness value, and $V^{WT}_{\#}$ is the smallest fit-ness value of the plant.

(5) Pollination. It determines whether the seeds survive or not. If euclidean distance between two plants is smaller than a preset threshold r, they are considered as neighbors. Pollination depends on the number of neighbors of a partic-ular plant. The more neighbors a plant has, the better its pol-lination is. Thus, a pollination factor for P is introduced and calculated as follows.

$$u_P ¼ e^{\frac{v_P}{v_{max}} - 1}: \quad (38)$$

Here, v_P is the number of neighbors of P, and v_{max} is the number of neighbors of the plant with the most neighbors in the population.

After pollination, the number of the survived seeds pro-duced by P is calculated as follows.

$$S_P ¼ \& u_P q_{max} \frac{V_{P\#} \_ V_{WT\#}}{V_{BT\#} \_ V_{WT\#}} , : \quad (39)$$

(6) Dispersion. Each new seed gets its value randomly which conforms to normal distribution with s as dispersion spread, that is, RPM_{ij}^{new} NðRPMij^old; sÞ. The spreading of seeds within the parameter space promotes that if the healthiest plant of a particular iteration corresponds to a local optimum, the dispersing seeds may fall in the parame-ter space corresponding to the global optimum.

(7) Local search ability improvement. SA is effective and computationally compact. It adapts itself to the local land-scape and contracts on to the optimum. The simplex cor-responds to problem solution. At first, its mirror center is built. Then, reflection is carried out to generate a new sim-plex at the mirror center, expansion to accelerate the reduction of the simplex to a better simplex, and contrac-tion to keep the simplex in good position. We use SA to improve PFA's local search ability. The LSIA based on SA is described as follows.

In LSIA, line 6, line 7, lines 12–7 and lines 20-27 corre-spond to building mirror center, reflection, expansion and con-traction respectively.

The proposed WDAPFA is described as follows.

In line 24, we get samples from successful transactions in the auction to train BPNN (see Section 5.2.1). If there is only one seed in OSS, it is the problem solution; otherwise, choose one seed randomly or by some user-specified rule (for example, TLS preferred, TUS preferred, or TRP pre-ferred) from OSS as the problem solution.

6 Simulations and Performance Evaluations

6.1 Simulation Setup

The proposed IEDA is implemented based on Simjava2.0 on Eclipse platform. The services and resource prices are set referring to Amazon and Ali clouds [6], [23], [36]. The resource capacity is set referring to TeraGrid [37]. The supply and demand relation is divided into four types, shown in Table 4. The market scale is classified into six categories according to the numbers of CSCs and CSPs in cloud market, shown in Table 5. The parameters of PFA and BPNN are set referring to [19] and [38], shown in Tables 6 and 7 respectively. c in (4) is set to be 0.5 refer-ring to [39].

TABLE 4
Supply and Demand Ratio

Supply and demand ratio	Value
Scarce Supply(SS)	[0.4, 0.9)
BaLance(BL)	[0.9, 1.1]
Over Supply(OS)	(1.1, 2]
Over sufficienT(OT)	(2, 4]

Algorithm 3. LSIA

Input: X1; . . . ; XP (seeds), MNoI (the maximum iteration number)

Output: X1; . . . ; XP (the improved seeds)

```
i ¼ 1;
while i _ MNoI do
  Choose one seed randomly as the benchmark X#;
  Calculate the SPD of each seed to X#;
  Set Xmax and Xmin be the seeds with the maximum and minimum SPD respectively (if multiple, choose one randomly);
  Average X1;X2; . . . ; XP to be X0;
  Xr ¼ X0 þ dðX0 _ XminÞ;
  if Vmin# _ Vr# _ Vmax# then
    Replace Xmin by Xr;
  end if
  if Vr# > Vmax# then
    Xe ¼ X0 þ _ðXr _ X0Þ;
    if Ve# > Vr# then
      Replace Xmin by Xe;
    else
      Replace Xmin by Xr;
    end if
  end if
  if Vr# < Vmin# then
    Xc ¼ X0 þ rðXmin _ X0Þ;
    if Vc# > Vmin# then
      ReplaceXminbyXc;
    else
      for j 2 f1; . . . ; pg do
        Xj ¼ ðXj þ XmaxÞ=2;
      end for
    end if
  end if
  i ¼ i þ 1;
end while
return X1; X2; . . . ; XP ;
```

TABLE 5
Market Scale

Market scale	CSC	CSP
TinY(TY)	8	4
SmalL(SL)	16	8
MediuM(MM)	32	8
LarGe(LG)	64	16
HuGe(HG)	128	16
OversiZed(OZ)	128	32

TABLE 6
PFA Parameter

Parameter	Value
Maximum iteration number	20
Maximum seeds produced by a plant	10
Population size	20
Maximum spread	0.1
Minimum spread	0.02
Radius	0.2

In order to evaluate economic efficiency and trustfulness of IEDA, we use simulation to verify its effectiveness and compare its performance with its counterpart which applies SCDA [40] to resource allocation. In SCDA, a compulsory bidding adjustment layer is added to CDA to promote con-tinuous matching and immediate allocation with low run-time overhead. In particular, it deals with resource allocation among self-interested participants in a dynamic and distributed market, and resource providers and con-sumers have their own asking/bidding strategies. Due to the treatment situation similarity and the auction nature, we choose the counterpart which applies SCDA as the com-parison benchmark to IEDA. The performance data used in the following are the average of 20 trials in corresponding simulation settings.

Algorithm 4. WDAPFA

Input: INoS (population size), MNoI (the maximum number of iterations)

Output: OSS (the optimal seed set)

```
Do sowing to generate INoS seeds initially and do seed refinement;
Choose one seed randomly as the benchmark;
Calculate the SPD of each seed to the benchmark;
Set MSPDBT and MSPD- to be the maximum SPD and initialize OSS with all seeds corresponding to MSPDBT;
i ¼ 1;
while i _ MNoI do
  Do seeding;
  Do pollination;
  Do dispersion;
  Improve local search ability;
  Do seed refinement;
  Calculate the SPD of each seed to the benchmark;
  Set MSPDBT be the current maximum SPD;
  if MSPDBT >MSPD- then
    Replace OSS with all seeds corresponding to MSPDBT;
    MSPD- ¼ MSPDBT;
  end if
  if MSPDBT ¼ ¼ MSPD- then
    Put all seeds corresponding to MSPDBT into OSS;
  end if
  Do selection;
  i ¼ i þ 1;
end while
Get necessary information from CSC and CSP winners as samples and put them into the corresponding PA's and CA's sample-base.
return OSS;
```

TABLE 7
BPNN Parameter

Parameter	Value
Learning factor	0.5
Output error	0.05
MNoS	400

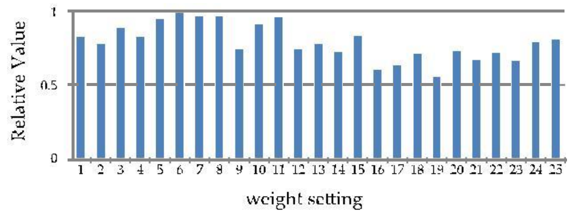

Fig. 7. SPD under different weight settings.

6.2 IEDA Effectiveness

When we do simulations in this section, we assume a cloud market with medium scale and balanced supply and demand.

6.2.1 Objective Weight Determination

We compare IEDA performances under different settings of a, b and g, shown in Table 8. In Fig. 7, we use the relative value of SPD to make the comparison more visualized, that is, we set the largest value of SPD be 1 and others be the ratios to 1. It can be seen that the best performance is pro-duced under a ¼ 0:5, b ¼ 0:4, and g ¼ 0:1. Thus, we use this weight setting in the following performance evaluations.

6.2.2 Reputation System

We define TWR as the ratios of the number of winners to the number of all participants in one auction, and DWR as the proportion of winners in dishonest participants respec-tively. They are computed under two different scenarios. In Scenario 1 (S1), IEDA is equipped with the proposed repu-tation system, and in Scenario 2 (S2) without. Specifically, when we do simulations, after the auction, if a participant A gives his partner B a QoE which is different from B's

TABLE 8
Weight Settings

Weight setting	a	b	g
1	0.6	0.2	0.2
2	0.6	0.3	0.1
3	0.6	0.1	0.3
4	0.5	0.3	0.2
5	0.5	0.2	0.3
6	0.5	0.4	0.1
7	0.5	0.1	0.4
8	0.2	0.6	0.2
9	0.3	0.6	0.1
10	0.1	0.6	0.3
11	0.3	0.5	0.2
12	0.2	0.5	0.3
13	0.4	0.5	0.1
14	0.1	0.5	0.4
15	0.2	0.2	0.6
16	0.1	0.3	0.6
17	0.3	0.1	0.6
18	0.2	0.3	0.5
19	0.3	0.2	0.5
20	0.4	0.1	0.5
21	0.1	0.4	0.5
22	0.4	0.2	0.4
23	0.2	0.4	0.4
24	0.4	0.4	0.2
25	0.33	0.33	0.33

previous reputation no more than 40 percent, A is consid-ered by AI honest; if more than 40 percent but no more than 70 percent, A is considered slightly dishonest; if more than 70 percent, A is considered seriously dishonest. If consid-ered slightly dishonest three consecutive times or seriously dishonest once, A is excluded from the cloud market.

Fig. 8a shows the ratio of TWR under S1 to that under S2 in a cloud market without dishonest participants. It can be seen that their TWRs are the same, that is, if all participants are honest, there is no need for reputation system.

We further evaluate the effectiveness of the proposed reputation system on suppressing the dishonest partici-pants. Fig. 8b shows the ratio of DWR under S1 to that under S2 in a cloud market where 10 percent CSCs and 10 percent CSPs are dishonest. Among dishonest partici-pants, 30 percent are seriously dishonest and others are slightly dishonest. We can see that in the first auction, the DWRs are the same under S1 and S2, because at the begin-ning the reputation system does not identify those dishon-est participants, thus it seems to be ineffective. However, after the first auction, all dishonest participants are identi-fied. In the second auction, the DWR under S1 is much lower than that under S2, because at this time all seriously dishonest ones have already been excluded from the mar-ket. The situation in the third auction is the same as that in the second auction, because all slightly dishonest partici-pants still take part in the auction although they are already suspected. From the fourth auction on, the DWR under S1 becomes 0, because all dishonest participants have been excluded from the market.

6.2.3 Price Formation and Winner Determination

We compare TLS, TUS, TRP and SPD under Scenario 3 (S3) and Scenario 4 (S4), Scenario 5 (S5) and Scenario 6 (S6) to evaluate the effectiveness of our proposed price formation mechanism (see Fig. 9a) and winner determination method (see Fig. 9b) respectively. In S3, IEDA is equipped with the proposed price formation mechanism while in S4 with GM inspired from [41]. GM means that the CSC with the highest bidding price transacts with the CSP with the lowest asking

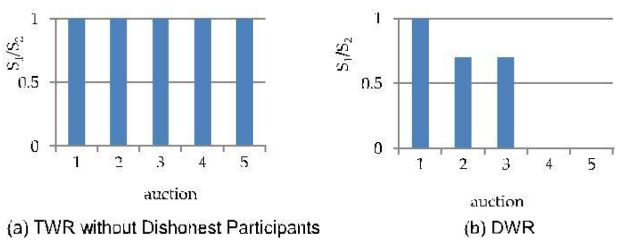

Fig. 8. TWR without dishonest participants and DWR.

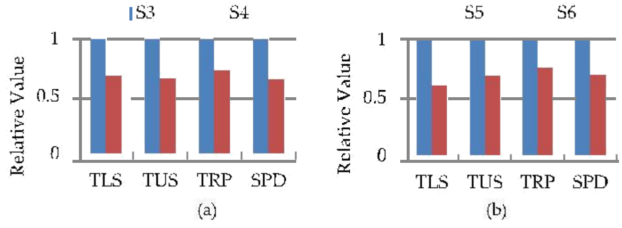

Fig. 9. Price formation and winner determination effectiveness.

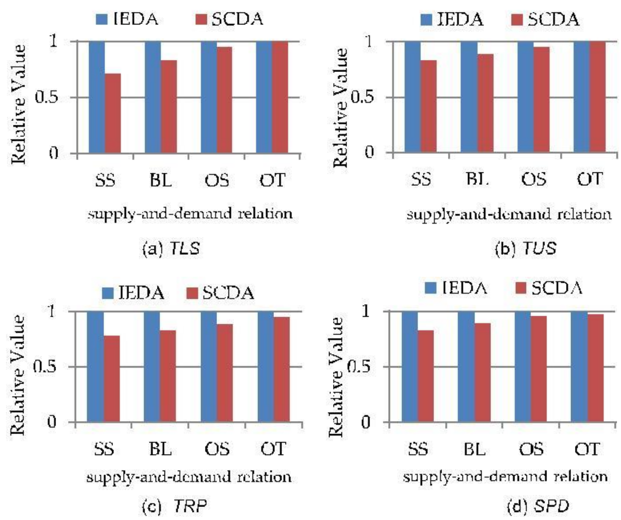

Fig. 10. Under different supply-and-demand relations.

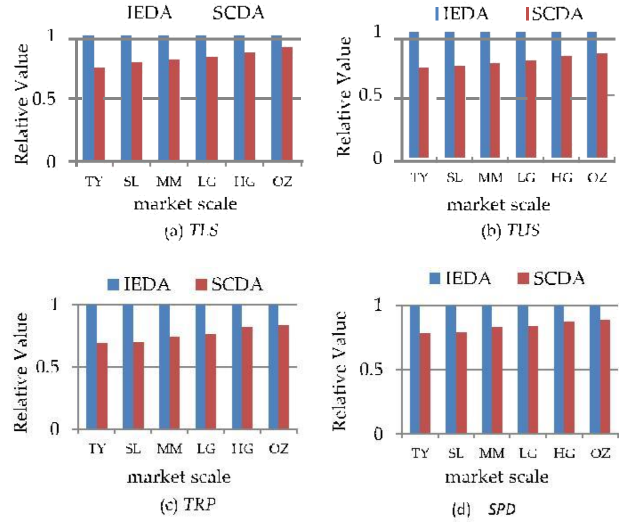

Fig. 11. Under different market scales.

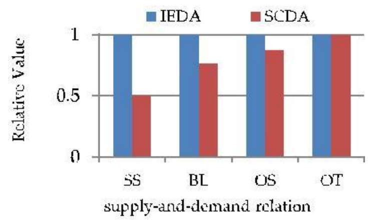

Fig. 12. Transaction number.

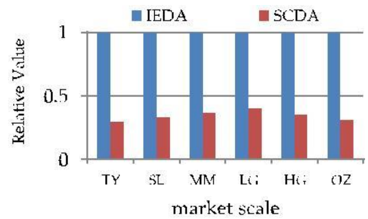

Fig. 13. Runtime overhead.

price. In S5, IEDA is equipped with WDAPFA while in S6 with GM. In Figs. 9a and 9b, we use the relative values of TLS, TUS, TRP and SPD, that is, we set their values in S3 and S5 be 1, and their values in S4 and S6 be the ratios to 1. It can be seen that our proposed methods are effective.

6.3 IEDA and SCDA Comparison

In this section, when we compare performance between IEDA and SCDA, we use the relative values of TLS, TUS, TRP, SPD, transaction number and runtime overhead, that is, we set their values in IEDA be 1, and their values in SCDA be the ratios to 1.

6.3.1 TLS, TUS, TRP and SPD

(1) Under different supply and demand relations. Fig. 10 shows comparison of TLS, TUS, TRP and SPD under different sup-ply and demand relations in a cloud market with medium scale. It can be seen that IEDA outperforms SCDA, however, as the SDR increases, the superiority of IEDA decreases. This is because the CSC demanded service cannot be parti-tioned to and carried out by multiple CSPs in SCDA, and thus some CSC demanded services cannot be accommo-dated due to insufficient resources, leading to TLS, TUS, TRP and SPD of IEDA better than those of SCDA. The scarcer the resources, the better the performance of IEDA than that of SCDA. When resources are over-sufficient, a CSC can always get resources from one CSP, thus IEDA and SCDA get almost the same performance.

(2) Under different market scales. Fig. 11 shows comparison of TLS, TUS, TRP and SPD under different market scales in a cloud market with balanced supply and demand relation. It can be seen that IEDA outperforms SCDA. However, as the market scale increases, the performance of IEDA tends to decrease slightly, because the stability of the bio-inspired PFA becomes worse when the problem space gets larger.

6.3.2 Transaction Number and Runtime Overhead

(1) Transaction number. Fig. 12 shows comparison of transac-tion number between IEDA and SCDA under different sup-ply and demand relations with medium scale. It can be seen that the number of transactions successfully dealt with in IEDA is the same as that in SCDA when resources are over-sufficient; however, in other cases, it is larger in IEDA than that in SCDA, because one CSC demanded service can be partitioned to and carried out by multiple CSPs in IEDA, then more transactions are accommodated.

(2) Runtime overhead. Fig. 13 shows comparison of run-time overhead between IEDA and SCDA under different market scales with balanced supply and demand relation. It can be seen that the runtime overhead of IEDA is larger than that of SCDA. The main reason is that IEDA has

integrated the BPNN-based price prediction, the PFA-based winner determination and the devised reputation system. At the cost of runtime overhead, IEDA not only brings good market surplus and surplus strength but also suppresses dishonest participants. SCDA emphasizes the instant resource allocation, thus its runtime overhead is low, but it does not offer the advantages of IEDA.

7 Conclusion

Based on economic method and bio-inspired algorithm, an intelligent combinatorial double auction based dynamic resource allocation approach is proposed for cloud services. The system framework is devised to provide a comprehen-sive solution. A reputation system is used to suppress dis-honest participants. A price formation mechanism is proposed to predict price and determine eligible transaction relationship. WDP is optimally solved by the improved PFA. Simulation results validate the effectiveness of our proposed approach and demonstrate its superiority on eco-nomic efficiency and trustfulness. In the near future, we expect to implement our proposed approach in a prototype system and do experiment on CERNET2 [42], which can deploy and provide cloud services to faculties and students at universities, to make it more practical.

References

[1] R. Buyya, C. S. Yeo, and S. Venugopal, "Market-oriented cloud computing: Vision, hype, and reality for delivering IT services as computing utilities," in Proc. 10th Int. Conf. High Perform. Comput. Commun., 2008, pp. 5–13.

[2] F. Zhang, J. W. Cao, K. Hwang, K. Q. Li, and S. U. Khan, "Adaptive workflow scheduling on cloud computing platforms with iterative ordinal optimization," IEEE Trans. Cloud Comput., 2014.

[3] F. Zhang, J. W. Cao, K. Q. Li, S. U. Khan, and K. Hwang, "Multi-objective scheduling of many tasks in cloud platforms," Future Generation Comput. Syst., vol. 37, pp. 309–320, 2014.

[4] G. N. Iyer and B. Veeravalli, "On the resource allocation and pric-ing strategies in compute clouds using bargaining approaches," in Proc. IEEE 17th Int. Conf. Netw., 2011, pp. 147–152.

[5] S. Zaman and D. Grosu, "A combinatorial auction-based mecha-nism for dynamic VM provisioning and allocation in clouds," IEEE Trans. Cloud Comput., vol. 1, no. 2, pp. 129–141, Oct. 2013.

[6] (2014). Amazon Web Services [Online]. Available. http://aws. amazon.com

[7] N. Grozev and R. Buyya, "Inter-cloud architectures and applica-tion brokering: Taxonomy and survey," Softw.: Practice Experience, vol. 44, no. 3, pp. 369–390, 2014.

[8] P. T. Endo, A. V. de Almeida Palhares, N. N. Pereira, G. E. Goncalves, D. Sadok, J. Kelner, B. Melander, and J.-E. Mangs, "Resource allocation for distributed cloud: Concepts and research challenges," IEEE Netw., vol. 25, no. 4, pp. 42–46, Jul./Aug. 2011.

[9] A. Bestavros and O. Krieger, "Toward an open cloud marketplace vision and first steps," IEEE Internet Comput., vol. 18, no. 1, pp. 72–77, Jan./Feb. 2014.

[10] W. J. Shi, L. Q. Zhang, C. Wu, Z. P. Li, and F. C. M. Lau, "An online auction framework for dynamic resource provisioning in cloud computing," ACM SIGMETRICS Perform. Eval. Rev., vol. 42, no. 1, pp. 71–83, 2014.

[11] P. Samimia, Y. Teimourib, and M. Mukhtara, "A combinatorial double auction resource allocation model in cloud computing," Inf. Sci., 2014.

[12] Q. Wang, K. Ren, and X. Q. Meng, "When cloud meets ebay: Towards effective pricing for cloud computing," in Proc. IEEE INFOCOM, 2012, pp. 936–944.

[13] L. Q. Zhang, Z. P. Li, and C. Wu, "Dynamic resource provisioning in cloud computing: A randomized auction approach," in Proc. IEEE INFOCOM, 2014, pp. 433–441.

[14] H. Zhang, B. Li, H. B. Jiang, F. M. Liu, A. V. Vasilakos, and J. C. Liu, "A framework for truthful online auctions in cloud comput-ing with heterogeneous user demands," in Proc. IEEE INFOCOM, 2013, pp. 1510–1518.

[15] O. Khalid1, S. U. Khan, S. A. Madani, K. Hayat, M. I. Khan, N. Min-Allah, J. Kolodziej, L. J. Wang, S. Zeadally, and D. Chen, "Comparative study of trust and reputation systems for wireless sensor networks," Security Commun. Netw., vol. 6, no. 6, pp. 669–688, 2013.

[16] Z. Wang and J. W. Cao, "Committee-based evaluation and selec-tion of grid resources for QoS improvement," in Proc. 10th IEEE/ ACM Int. Conf. Grid Comput., 2009, pp. 138–144.

[17] T. Hobfeld, R. Schatz, M. Varela, and C. Timmerer, "Challenges of QoE management for cloud applications," IEEE Commun. Mag., vol. 50, no. 4, pp. 28–36, Apr. 2012.

[18] J. W. Yuan and S. C. Yu, "Privacy preserving back-propagation neural network learning made practical with cloud computing," IEEE Trans. Parallel Distrib. Syst., vol. 25, no. 1, pp. 212–221, Jan. 2014.

[19] U. Premaratne, J. Samarabandu, and T. Sidhu, "A new biologically inspired optimization algorithm," in Proc. Int. Conf. Ind. Inf. Syst., 2009, pp. 279–284.

[20] K. Chard and K. Bubendorfer, "High performance resource alloca-tion strategies for computational economies," IEEE Trans. Parallel Distrib. Syst., vol. 24, no. 1, pp. 72–84, Jan. 2013.

[21] X. W. Wang, J. J. Sun, H. X. Li, C. Wu, and M. Huang, "A reverse auction based allocation mechanism in the cloud computing envi-ronment," Appl. Math. Inf. Sci., vol. 7, no. 1, pp. 75–84, 2013.

[22] X. L. Shi, K. Xu, J. C. Liu, and Y. Wang, "Continuous double auc-tion mechanism and bidding strategies in cloud computing mar-kets," Comput. Sci. Game Theory, 2013.

[23] S. F. Shang, J. L. Jiang, Y. W. Wu, G. W. Yang, and W. M. Zheng, "A knowledge-based continuous double auction model for cloud market," in Proc. IEEE 6th Int. Conf. Semantics Knowl. Grid, 2010, pp. 129–134.

[24] M. Mac_ıas and J. Guitart, "Using resource-level information into non-additive negotiation models for cloud market environments," in Proc. IEEE Netw. Oper. Manag. Symp., 2010, pp. 325–332.

[25] S. Di and C. L. Wang, "Dynamic optimization of multi-attribute resource allocation in self-organizing clouds," IEEE Trans. Parallel Distrib. Syst., vol. 24, no. 3, pp. 464–478, Mar. 2013.

[26] I. Fujiwara, K. Aida, and I. Ono, "Applying double-sided combi-national auctions to resource allocation in cloud computing," in Proc. IEEE/IPSJ Int. Symp. Appl. Internet, 2010, pp. 7–14.

[27] H. Xu and B. C. Li, "Dynamic cloud pricing for revenue maxi-mization," IEEE Trans. Cloud Comput., vol. 1, no. 2, pp. 158–171, Jul.–Dec. 2013.

[28] M. Maci_ıas and J. Guitart, "A genetic model for pricing in cloud computing markets," in Proc. ACM Symp. Appl. Comput., 2011, pp. 113–118.

[29] G. S. Atsalakis and K. P. Valavanis, "Surveying stock market fore-casting techniques-Part II: Soft computing methods," Expert Syst. Appl., vol. 36, no. 3, pp. 5932–5941, 2009.

[30] M. Mac_ıas and J. Guitart, "Cheat-proof trust model for cloud com-puting markets," Econom. Grids, Clouds, Syst., Services Lecture Notes Comput. Sci., vol. 7714, pp. 154–168, 2012.

[31] K. Y. Chan, T. S. Dillon, J. Singh, and E. Chang, "Neural-network-based models for short-term traffic flow forecasting using a hybrid exponential smoothing and Levenberg–Marquardt algo-rithm," IEEE Trans. Intell. Transp. Syst., vol. 13, no. 2, pp. 644–654, Jun. 2012.

[32] S. Haykin, Neural Networks: A Comprehensive Foundation. New York, NY, USA: Pearson Education Press, 1999.

[33] P. Klemperer, Auctions: Theory and Practice. Princeton, NJ, USA: Princeton Univ. Press, 2004.

[34] S. Z. Martinez, A. A. Montano and C. A. Coello Coello, "A nonlin-ear simplex search approach for multi-objective optimization," in Proc. IEEE Conf. Evolutionary Comput., 2011, pp. 2367–2374.

[35] H. Y. Tsai and Y. L. Huang, "An analytic hierarchy process-based risk assessment method for wireless networks," IEEE Trans. Rel., vol. 60, no. 4, pp. 801–816, Dec. 2011.

[36] (2014). Ali Cloud Services [Online]. Available: http://www.aliyun. com

[37] (2010). Teragrid Compute and Visualization Resources [Online]. Available: https://www.teragrid.org/web/user support/ compute_resources/

[38] B. Subudhi and D. Jena, "Nonlinear system identification using memetic differential evolution trained neural networks," Neuro-computing, vol. 74, no. 10, pp. 1696–1709, 2011.

[39] H. B. Mi, H. M. Wang, G. Yin, Y. F. Zhou, D. X. Shi, and L. Yuan, "Online self-reconfiguration with performance guarantee for energy-efficient large-scale cloud computing data centers," in Proc. IEEE Int. Conf. Services Comput., 2010, pp. 514–521.

[40] Z. Tan and J. R. Gurd, "Market-based grid resource allocation using a stable continuous double auction," in Proc. IEEE/ACM 8th Int. Conf. Grid Comput., 2007, pp. 283–290.

[41] Y. W. Lan, W. Q. Tong, Z. H. Liu, and Y. Hou, "Multi-unit contin-uous double auction based resource allocation method," in Proc. 3rd Int. Conf. Intell. Control Inf. Process., 2012, pp. 773–777.

[42] (2014). CERNET2 [Online]. Available: http://www.cernet2.edu.cn