Cumhuriyet Üniversitesi Fen Fakültesi
Fen Bilimleri Dergisi (CFD), Cilt:36, No: 3 Özel Sayı (2015)
ISSN: 1300-1949

Cumhuriyet University Faculty of Science
Science Journal (CSJ), Vol. 36, No: 3 Special Issue (2015)
ISSN: 1300-1949

Resource-provision scheduling in cloud datacenter

Anis VOSOOGH[1], Reza NOURMANDİ-POUR[2,*]

[1]*Department of computer engineering, Sirjan Science and research branch, Islamic azad university, Sirjan, Iran & Department of computer engineering, Sirjan branch, Islamic azad university, Sirjan, Iran*

[2]*Department of computer engineering, Sirjan branch, Islamic azad university, Sirjan, Iran*

Received: 01.02.2015; Accepted: 05.05.2015

Abstract. Cloud computing, the long-held dream of computing as a utility, has the potential to transform a large part of the IT industry, making software even more attractive as a service and shaping the way in which hardware is designed and purchased. We review the new cloud computing technologies, and indicate the main challenges for their development in future, among which resource management problem stands out and attracts our attention. Combining the current scheduling theories, we propose cloud scheduling hierarchy to deal with different requirements of cloud services. we settle the evaluation problem for on-line schedulability tests in cloud computing. We propose a concept of test reliability to express the probability that a random task set could pass a given schedulability test. The larger the probability is, the more reliable the test is. From the aspect of system, a test with high reliability can guarantee high system utilization. From the practical aspect, we develop a simulator to model MapReduce framework. This simulator offers a simulated environment directly used by MapReduce theoretical researchers. The users of SimMapReduce only concentrate on specific research issues without getting concerned about finer implementation details for diverse service models, so that they can accelerate study progress of new cloud technologies.

Keywords: Cloud computing, Game theory, Normal formulation, Types of games

1. INTRODUCTION

When we open any IT magazines, websites, radios or TV channels, ”cloud” will definitely catch our eye. The cloud has reached into our daily life and led to a broader range of innovations, but people often misunderstand what cloud computing is. Built on many old IT technologies, cloud computing is actually an evolutionary approach that completely changes how computing services are produced, priced and delivered. It allows to access services that reside in a distant datacenter, other than local computers or other Internet-connected devices. Cloud services are charged according to the amount consumed by worldwide users. Such an idea of computing as a utility is a long-held dream in the computer industry, but it is still immature until the advent of low-cost datacenters that will enable this dream to come true.

1.1. Objectives And Contributions

This thesis studies resource management problems related to cloud computing, such as resource allocation, scheduling and simulation. The major contributions are as follows.

• **A survey of current trends and research opportunities in cloud computing.** Weinvestigate the state-of-the-art efforts on cloud computing, from both industry and academic standpoints. Through comparison with other related technologies and computing paradigms, we identify several challenges from the cloud adoption perspective. We also highlight the resource management issue that deserves substantial research and development.

*Corresponding author. *Email address*: *noormandi_r@iausirjan.ac.ir*

Special Issue: The Second National Conference on Applied Research in Science and Technology

• **A cloud scheduling hierarchy to distinguish different requirements of cloud services.**

We systemize the scheduling problem in cloud computing, and present a cloud scheduling hierarchy, mainly splitting into user-level and system-level. Economic models are investigated for resource provision issues between providers and customers, while heuristics are discussed for meta-task execution on system-level scheduling. Moreover, priority scheduling algorithms are complemented for real-time scheduling.

• **A game theoretical resource allocation algorithm in clouds.** We introduce game theory to solve the user-centric resource competition problem in cloud market. Our algorithm substitutes the expenditure of time and cost in resource consumption, and allows cloud customers to make a reasonable balance between budget and deadline requirements. We supplement the bid-shared auction scheme in Cloudsim to support on-line task submission and execution. Under sequential games, a Nash equilibrium allocation among cloud users can be achieved.

• **A price prediction method for games with incomplete information.** We propose an effective method to forecast the future price of resources in sequent games, especially when common knowledge is inadequate. This problem arises from the nature of an open market, which enables customers holding different tasks to arrive in datacenters without a prior fixed arrangement. Besides that, the independent customers have little or limited knowledge about others. In that case, our Bayesian learning prediction has stable performance, which can accelerate the search of Nash equilibrium allocation.

• **A theoretical utilization bound for real-time tasks on MapReduce cluster.** We address the scheduling problem of real-time tasks on MapReduce cluster. Since MapReduce consists of two sequential stages, segmentation execution enables cluster scheduling to be more flexible. We analyze how the segmentation between Map and Reduce influences cluster utilization. Through finding out the worst pattern for schedulable task set, we deduce the upper bound of cluster utilization, which can be used for on-line schedulability test in time complexity O(1). This theoretical bound generalizes the classic Liu's result, and even performs better when there is a proper segmentation between Map and Reduce.

• **A reliability indicator for real-time admission control test.** We settle the comparison difficulty among real-time admission control tests. Admission control test aims at determining whether an arriving task can be scheduled together with the tasks already running in a system, so it can prevent system from overload and collapse. We introduce a concept of test reliability to evaluate the probability that a random task set can pass a given test, and define an indicator to show the test reliability. Our method is useful as a criterion

to compare the effectiveness of different tests. In addition, an insufficient argument in previous literature is questioned and then completed.

• **A performance analysis for schedulability test on MapReduce cluster.** We examine accepted ratio of several most used priority-driven schedulability tests on a simulated MapReduce cluster. The development of ubiquitous intelligence increases the real-time requirements for a cloud datacenter. If one real-time computation does not complete before its deadline, it is as serious as that the computation was never executed at all. To avoid ineffective computation, the datacenter needs a schedulability test to ensure its stability. From both realizability analysis and experimental results, we find out that the performance discrepancy of schedulability test is determined by a prerequisite pattern. This pattern can be deduced by a reliability indicator, so it may help system designers choose a good schedulability test in advance.

• **A simulation toolkit to model the MapReduce framework.** We develop a MapReduce simulator, named SimMapReduce, to facilitate research on resource management and

performance evaluation. SimMapReduce endeavors to model a vivid MapReduce environment, considering some special features such as data locality and dependence between Map and Reduce, and it provides essential entity services that can be predefined in XML format. With this simulator, researchers are free to implement scheduling algorithms and resource allocation policies by inheriting the provided java classes without implementation details concerns.

1.2 Deployment Models

Clouds are deployed in different fashions, depending on the usage scopes. There are four primary cloud deployment models.

• **Public cloud** is the standard cloud computing paradigm, in which a service provider makes resources, such as applications and storage, available to the general public over Internet. Service providers charge on a fine-grained utility computing basis. Examples of public clouds include Amazon Elastic Compute Cloud (EC2), IBM's Blue Cloud, Sun Cloud, Google AppEngine and Windows Azure Services Platform.

• **Private cloud** looks more like a marketing concept than the traditional mainstreamsense. It describes a proprietary computing architecture that provides services to a limited number of people on internal networks. Organizations needing accurate control over their data will prefer private cloud, so they can get all the scalability, metering, and agility benefits of a public cloud without ceding control, security, and recurring costs to a service provider. Both eBay and HP CloudStart yield private cloud deployments.

• **Hybrid cloud** uses a combination of public cloud, private cloud and even local infrastructures, which is typical for most IT vendors. Hybrid strategy is proper placement of workloads depending upon cost and operational and compliance factors. Major vendors including HP, IBM, Oracle and VMware create appropriate plans to leverage a mixed environment, with the aim of delivering services to the business. Users can deploy an application hosted on a hybrid infrastructure, in which some nodes are running on real physical hardware and some are running on cloud server instances.

• **Community cloud** overlaps with Grids to some extent. It mentions that several organizations in a private community share cloud infrastructure. The organizations usually have similar concerns about mission, security requirements, policy, and compliance considerations. Community cloud can be further aggregated by public cloud to build up a cross-boundary structure.

1.3. Getting Ready For Cloud

• **Datacenter**: Even faster than Moore's law, the number of servers and datacenters has increased dramatically in past few years. Datacenter has become the reincarnation of the mainframe concept. It is easier to build a large-scale commodity-computer datacenter than ever before, just gathering these building blocks together on a parking lot and plugging them into the Internet .

• **Internet**: Recently, network performance has improved rapidly. Wired, wireless and 4^{th} generation mobile communication make Internet available to most of the planet. Cities and towns are wired with hotspots. The transportation such as air, train, or ship also equips with satellite based wi-fi or undersea fiber-optic cable. People can connect to the Internet anywhere and at anytime. The universal, high-speed, broadband Internet lays the foundation for the widespread applications of cloud computing.

• **Terminals**: PC is not the only central computing device, various electronic devices including MP3, SmartPhone, Tablet, Set-top box, PDA, notebook are new terminals that have the requirement of personal computing. Besides, repeated data synchronization among different

terminals is time-consuming so that faults occur frequently. In such cases, a solution that allows individuals to access to personal data anywhere and from any device is needed.

2. A BRIEF HISTORY

Along with the maturity of objective conditions (software, hardware), plenty of existing technologies, results, and ideas can be realized, updated, merged and further developed. Amazon played a key role in the development of cloud computing by initially renting their datacenter to external customers for the use of personal computing. In 2006, they launched Amazon EC2 and S3 on a utility computing basis. After that, several major vendors released cloud solutions one after another, including Google, IBM, Sun, HP, Microsoft, Forces.com, Yahoo and so on. Since 2007, the number of trademarks covering cloud computing brands, goods and services has increased at an almost exponential rate. Cloud computing is also a much favored research topic. In 2007, Google, IBM and a number of universities announced a research project, Academic Cloud Computing Initiative (ACCI), aiming at addressing the challenges of large-scale distributed computing. Since 2008, several open source projects have gradually appeared. For example, Eucalyptus is the first API-compatible platform for deploying private clouds. OpenNebula deploys private and hybrid clouds and federates different modes of clouds. In July 2010, SiteonMobile was announced by

HP for emerging markets where people are more likely to access the Internet via mobile phones rather than computers. With more and more people owning smartphones, mobile cloud computing has turned out to be a potent trend. Several mobile network operators such as Orange, Vodafone and Verizon have started to offer cloud computing services for companies. In March 2011, Open Networking Foundation consisting of 23 IT companies was founded by Deutsche Telecom, Facebook, Google, Microsoft, Verizon, and Yahoo. This nonprofit organization supports a new cloud initiative called Software-Defined Networking. The initiative is meant to speed innovation through simple software changes in telecommunications networks, wireless networks, data centers and other networking areas.

2.1 Comparison With Related Technologies

Cloud computing is a natural evolution of widespread adoption of virtualization, serviceoriented architecture, autonomic and utility computing. It emerges as a new computing paradigm to provide reliable, customized and quality services that guarantee dynamic computing environments for end-users, so it is easily confused with several similar computing paradigms such as grid computing, utility computing and autonomic computing.

2.2 Utility Computing

Utility computing was initialized in the 1960s, John McCarthy coined the computer utility in a speech given to celebrate MIT's centennial "If computers of the kind I have advocated become the computers of the future, then computing may someday be organized as a public utility just as the telephone system is. The computer utility could become the basis of a new and important industry." Generally, utility computing considers the computing and storage resources as a metered service like water, electricity, gas and telephony utility. The customers can use the utility services immediately whenever and wherever they need without paying the initial cost of the devices. This idea was very popular in the late 1960s, but faded by the mid- 1970s as the devices and technologies of that time were simply not ready. Recently, the utility idea has resurfaced in

new forms such as grid and cloud computing. Utility computing is highly virtualized so that the amount of storage or computing power available is considerably larger than that of a single time-sharing computer. The back-end servers such as computer cluster and supercomputer are used to realize the virtualization. Since the late 90's, utility computing has resurfaced. HP launched the Utility Datacenter to provide the IP billing-on-tap services. PolyServe Inc. built a clustered file system that created highly available utility computing environments for mission-critical applications and workload optimized solutions. With utility including database and file service, custumers of vertical industry such as financial services, seismic processing, and content serving can independently add servers or storage as needed.

2.3 Grid Computing

Grid computing emerged in the mid 90's. Ian Foster integrated distributed computing, objectoriented programming and web services to coin the grid computing infrastructure. "A Grid is a type of parallel and distributed system that enables the sharing, selection, and aggregation of geographically distributed autonomous resources dynamically at runtime depending on their availability, capability, performance, cost, and users' quality-of-service requirements."[1] The definition explains that a gird is actually a cluster of networked, loosely coupled computers which works as a super and virtual mainframe to perform thousands of tasks. It can divide the huge application job into several subjobs and make each run on large-scale machines. Generally speaking, grid computing goes through three different generations [2]. The first generation is marked by early metacomputing environment, such as FAFNER and I-WAY. The second generation is represented by the development of core grid technologies, grid resource management (e.g., GLOBUS, LEGION), resource brokers and schedulers (e.g., CONDOR, PBS) and grid portals (e.g., GRID SPHERE). The third generation merges grid computing and web services technologies (e.g., WSRF, OGSI), and moves to a more service oriented approach that exposes the grid protocols using web service standards.

2.3.1 Autonomic computing

Autonomic computing, proposed by IBM in 2001, performs tasks that IT professionals choose to delegate to the technology according to policies. [3] Adaptable policy rather than hard coded procedure determines the types of decisions and actions that autonomic capabilities perform. Considering the sharply increasing number of devices, the heterogeneous and distributed computing systems are more and more difficult to anticipate, design and maintain. The complexity of management is becoming the limiting factor of future development. Autonomic computing focuses on the self-management ability of the computer system. It overcomes the rapidly growing complexity of computing systems management and reduces the barriers that the complexity poses on further growth. In the area of multi-agent systems, several self-regulating frameworks have been proposed, with centralized architectures. These architectures reduce management costs, but seldom consider the issues of enabling complex software systems and providing innovative services. IBM proposed the self-managing system that can automatically process, including configuration of the components (Self-Configuration), automatic monitoring and control of resources to ensure the optimal (Self-Healing), monitor and optimize the resources (Self-Optimization) and proactive identification and protection from arbitrary attacks (Self-Protection), only with the input information of policies defined by humans [73]. In other words, the autonomic system useshigh-level rules to check its status and automatically adapt itself to changing conditions. According to the above introductions of the three computing paradigms, we conclude the relationship among them. Utility computing concerns whether the packing computing resources can be used as a metered service on the basis of the user's need. It is indifferent to the organization of the resources, both in the centralized and distributed systems. Grid computing is conceptually similar to the canonical Foster definition of cloud computing, but it does not take economic entities into account. Autonomic computing stresses the self management of computer

systems, which is only one feature of cloud computing. All in all, cloud computing is actually a natural next step from the grid-utility model, having grid technologies, autonomic characteristics and utility bills.

2.3.2. Technical aspects

Technical characteristics are the foundation that ensures other functional and economical requirements. Not every technology is absolutely new, but is enhanced to realize a specific feature, directly or as a pre condition.

• **Virtualization** is an essential characteristic of cloud computing. Virtualization in clouds refers to multilayer hardware platforms, operating systems, storage devices, network resources, etc. The first prominent feature of virtualization is the ability to hide the technical complexity from users, so it can improve independence of cloud services. Secondly, physical resource can be efficiently configured and utilized, considering that multiple applications are run on the same machine. Thirdly, quick recovery and fault tolerance are permitted. Virtual environment can be easily backed up and migrated with no interruption in service[4].

• **Multi-tenancy** is a highly requisite issue in clouds, which allows sharing of resources and costs across multiple users. Multi-tenancy brings resource providers many benefits, for example, centralization of infrastructure in locations with lower costs and improvement of utilization and efficiency with high peakload capacity. Tenancy information, which is stored in a separate database but altered concurrently, should be well maintained for isolated tenants. Otherwise some problems such as data protection will arise.

• **Security** is one of the major concerns for adoption of cloud computing. There is no reason to doubt the importance of security in any system dealing with sensitive and private data. In order to obtain the trust of potential clients, providers must supply the certificate of security. For example, data should be fully segregated from one to another, and an efficient replication and recovery mechanism should be prepared if disasters occur. The complexity of security is increased when data is distributed over a wider area and shared by unrelated users. However, the complexity reduction is necessary, owing to the fact that ease-of-use ability can attract more potential clients.

• **Programming environment** is essential to exploit cloud features. It should be capable of addressing issues such as multiple administrative domains, large variations in resource heterogeneity, performance stability, exception handling in highly dynamic environments, etc. System interface adopts web services APIs, which provide a standards-based framework for accessing and integrating with and among cloud services. Browser, applied as the interface, has attributes such as being intuitive, easy-to-use, standards-based, serviceindependent and multi-platform supported. Through pre-defined APIs, users can access, configure and program cloud services.

2.3.3. Resource-provision scheduling in cloud datacenter

2.3.3.1. Introduction

Clouds gradually change the way we use computing resources. In cloud computing, everything can be treated as a service, which is customized and easily purchased in the market, like other consumption goods. This evolution is mainly caused by developed virtualization technology, which hides heterogeneous configuration details from customers. Therefore, the resource allocation problem in cloud computing needs to take market dealing behaviors into consideration, not only match-making scheduling tasks and machines [22]. Market mechanism is used as an effective method to control electronic resources, but the existing market models are dedicated either to maximizing suppliers' revenue, or to balancing the supply-demand relationship [40]. In

this chapter, we shall focus on helping a cloud customer make a reasonable decision in a competitive market. Game theory studies multi-person decision making problems. If no one wants to deviate from a strategy, the strategy is in a state of equilibrium. Although there has been researches on allocation strategies using game theory [5, 6, 7, 8, 9, 10, 11, 12], none suits the new computing service market perfectly. In order to establish a proper model for clouds, several important consumer characters should be highlighted. Firstly, cloud users are egocentric and rational, wishing to get better service at a lower cost. Secondly, these buyers have more than one behavioral constraint, so they have to make a trade-off of one constraint for another in management practice. Thirdly, the pay-as-you-go feature means that transactions are never static, but repeated gambling processes. Each user can adjust its bid price according to prior behaviors of other competitors. Fourthly, cloud customers are distributed globally, so they do not know each other very well. In other words, there is no common purchasing knowledge in the whole system. Fifthly, tasks arrive in datacenter without a prior arrangement. Sixthly, the accurate forecast becomes more challenging in such a complex scenario, so a good allocation model integrating compromise, competition and prediction should be further generalized and well evaluated. Given the above challenges, we therefore use game theoretical auctions to solve the resource allocation problem in clouds, and propose practicable algorithms for user bidding and auctioneer pricing. With Bayesian learning prediction, resource allocation can reach Nash equilibrium among non-cooperative users even if common knowledge is lacking or dynamically updated. The rest of this chapter is organized as follows. A short tutorial on game theory is given first, covering the different classes of games and their applications, payoff choice and utility function, as well as strategic choice and Nash equilibrium. Next, a non-cooperative game for resource allocation is built. The scheduling model includes bid-shared auction, user bid function, price forecasting and equilibrium analysis. Based on equilibrium allocation, we propose simulation algorithms running on the Cloudsim platform. After that Nash equilibrium and forecasting accuracy are evaluated.

2.4. Game Theory

Game theory models strategic situations, in which an individual's payoff depends on the choices of others. It provides a theoretical basis for the field of economics, business, politics, logic, computer science, and is an effective approach to achieve equilibrium in multi-agent systems, computational auctions, peer-to-peer systems, and security and information markets. With the development of cloud service market, game theory is useful to address the resource allocation problems in cloud systems where agents are autonomous and self-interested.

2.4.1. Normal Formulation

Game is an interactive environment where the benefit for an individual choice depends on the behaviors of other competitors. A normal game consists of all conceivable strategies, and their corresponding payoffs, of every player. There are several important terms to characterize a normal form of game [13].

Player is the game participant. There is a finite set of players P = {1, 2, · · · ,m}.

Strategy is the action taken by one player. Each player k in P has a particular strategy space containing finite number of strategies, $S_k = \{s_k^1, s_k^2, \cdots, s_k^n\}$. Strategy space is S =

S1 × S2 × · · · × Sm. The game outcome is a combination of strategies of m players s = (s1, s2, · · · ,sm), si ∈ Si.

Payoff is the utility received by a single player at the outcome of one game, which determines the player's preference. For resource allocation, payoff stands for the amount of resource received, for example, ui(s) represents the payoff of player i when the output of the game is s, s ∈ S. Payoff

function U = {u1(S), u2(S), · · · , um(S)} specifies for each player in the player set P. Therefore, the normal form of a game is a structure as:

$$G =< P, S,U > (4.1)$$

2.4.2. Types of games

Although classes of games are various, we only list three common criteria in cloud computing market.

2.4.3. Non-cooperative or cooperative players

A Non-cooperative game is characterized by a set of independent players who optimize their own payoff. This model is most used in a competitive market. We take cloud service market for instance. There are a great number of small and medium-sized enterprises as well as widely distributed customers. Efficient communication and cooperation among them are insufficient and impossible, so the non-cooperation game suits for analyzing the behaviors of these egocentric cloud agents. On the contrary, a cooperative game is the one where players from different coalitions may make cooperative decisions, so competition here is between coalitions, rather than between individual players. Cooperative game is useful when several agents have a common goal. For example, the users in P2P file-sharing network have the same object, maximizing the availability of desirable files. With the development of electronic commerce, worldwide cloud markets

are collective and localized, such as Groupon and Google offers. Compared with the above games, non-cooperative games model situations to the finest details, while cooperative games focus on the game at large.

2.5. Simultaneous or sequential actions

A simultaneous game is the one where all players make their decisions simultaneously, without knowledge of the strategies chosen by other players. Simultaneous game model is used in sealed-bid auctions in tendering for leases, where no one knows bids of other competitors. A repeated game is the one consisting of some number of repetitions of simultaneous game. A player has to take into account the impact of his action on the future actions of other players, and makes the current decision based on past experience. In a repeated game, the threat of retaliation is real, since one will play the game again with the same competitors. Proxy bidding on eBay is an example of repeated game, in which the current highest bid is always displayed.

Under a sophisticated mechanism, rational players bid the maximum amount on their first round, and never raise their bids. In a sequential game, one player chooses his strategy before the others do, so the later one

has some knowledge about the earlier players. The sequential game model is easily applied in English auction, where players bid openly against one another, with each subsequent bid higher than the previous one.

2.6. Complete or incomplete information

Information refers to the game characteristic including the number of players as well as their strategy spaces and payoffs. A game of complete information is the one in which information is available to all players. Each participant knows all strategies and corresponding payoffs, but does not necessarily know the actions taken by other players inside the game. Complete information is a strict assumption, which is difficult to be implemented in reality. For example in a sealed-bid

auction each player knows his own valuation for the service but does not know competitors' valuations. Although private information is not common knowledge

among players, everyone has some beliefs about what his competitors know. In the situation of asymmetric information, we assume that every player knows his own payoff function, but is uncertain about others'.

2.7. Payoff choice and utility function

In cloud computing market, service providers and their customers have their own preferences. Providers balance the investments on capital, operation, labor and device. Customers have different QoS requirements, such as cost, execution time, access speed, throughput and stability. All these preferences impact on agents' choices, thus an integrated indication to guide agents' behaviors is necessary. Utility is a measure of relative satisfaction in economics. It is often expressed as a function to describe the payoff of agents. More specifically, utility function combines more than one service requirements and analyzes Pareto efficiency under certain assumptions such as service consumption, time spending, money possession. Therefore, utility is very useful when a cloud agent tries to make a wise decision. High value of utility stands for great preference of service when the inputs are the same. One key property of utility function is constant elasticity of substitution (CES). It combines two or more types of consumption into an aggregate quantity. The CES function is

$$C = [\sum_{i=1}^{n} a_i^{\frac{1}{s}} c_i^{\frac{s-1}{s}}]^{\frac{s}{s-1}}$$

C is aggregate consumption, ci is individual consumptions, such as energy, labor, time, capital, etc. The coefficient ai is share parameter, and s is elasticity of substitution. These consumptions are perfect substitutes when s approaches infinity, and are perfect complements when s approaches zero. The preferences for one factor over another always change, so the marginal rate of substitution is not constant. For the sake of simplicity, s equals one in the following analysis. Let r = (s − 1)/s, we obtain

$$\ln C = \frac{\ln \sum_{i=1}^{n} (a_i^{1-r} c_i^r)}{r}$$

Apply l'Hopital's rule,

$$\lim_{r \to 0} \ln C = \frac{\sum_{i=1}^{n} a_i \ln c_i}{\sum_{i=1}^{n} a_i}$$

If $\sum_{i=1}^{n} a_i = 1,$ the consumption function has constant returns to scale, which means that the consumption increased by the same percentage as the rate of growth of each consumption good. If every ai is increased by 20%, C increases by 20% accordingly. If $\sum_{i=1}^{n} a_i < 1,$ the returns to scale decrease, on the contrary, returns to scale increase. We take two QoS requirements, speed and stability, for example. The CES function is shown in Figure 1.

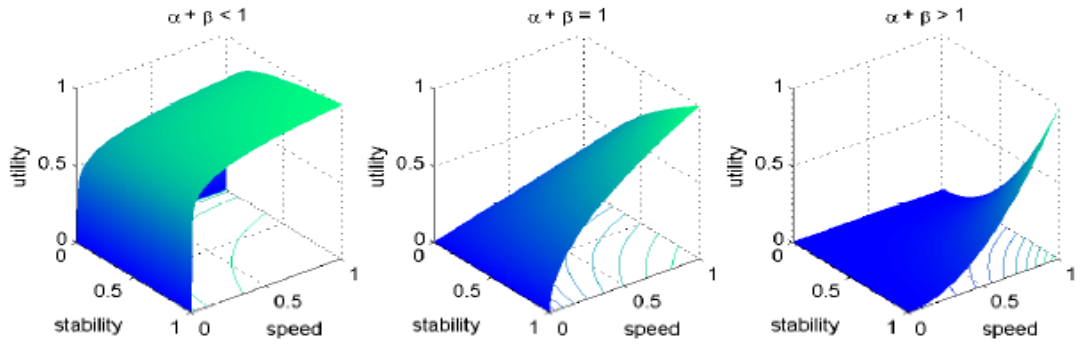

Figure 1. CES functions.

The contour plot beneath the surface signifies a collection of indifference curves, which can represents observable demand patterns over good bundles. Every curve shows different bundles of goods, between which a consumer has no preference for one bundle over another. One can equivalently refer to each point on the indifference curve as rendering the same level of utility for the customer. Especially, CES function is a general expression of Cobb Douglas function. Cobb Douglas function has been widely used in consumption, production and other social welfare analysis. It can build a utility function. In a generalized form, where c1, c2, · · · , cn are the quantities consumed of n goods, the utility function representing the same preferences is written as:

$$\tilde{u}(c) = \prod_{i=1}^{n} c_i^{a_i}$$

with c = (c1, c2, · · · , cn). Set $a = \sum_{i=1}^{n} a_i$, we obtain the function $c \longmapsto c^{\frac{1}{a}}$, which is strictly monotone for c > 0.

$$u(c) = \tilde{u}(c)^{\frac{1}{a}}$$

represents the same preferences. Setting ρi = ai/a it can be shown that

$$u(c) = \prod_{i=1}^{n} c_i^{\rho_i}, \qquad \sum_{i=1}^{n} \rho_i = 1$$

The problem of maximum utility is solved by looking at the logarithm of the utility

$$\max_{c} \sum_{i=1}^{n} \rho_i \ln c_i$$

2.8. Strategy choice and Nash equilibrium

Nash equilibrium is a certain combination of strategy choices, under which no player can benefit by unilaterally changing his strategy while the other players keep theirs unchanged. Nash equilibrium is under the assumption that all players are rational and that their rationality is common knowledge. A formal definition of Nash equilibrium is as follows. Let G =< P, S,U > be a game and si be a strategy profile of all players except for player i. After each player i has chosen their strategies, player i obtains payoff ui(s1, · · · , sn). Note that the payoff depends on the strategy chosen by player i as well as the strategies chosen by all the other players. A strategy profile {s∗1

, · · · , s∗ n} ∈ S is a Nash equilibrium if no unilateral deviation in strategy by any single player is profitable for that player, that is

$$\forall i, s_i \in S_i, s_i \neq s_i^* : u_i(s_i^*, s_{-i}^*) > u_i(s_i, s_{-i}^*)$$

Nash equilibrium analyzes a strategy profile under the assumption of complete information. However, if some information is private, and not known to all players, the players with incomplete information have to evaluate the possible strategy profiles. In particular, every rational player tries to take an action which maximizes its own expected payoffs, supposing a particular probability distribution of actions taken by other competitors. Therefore, the belief about which strategies other players will choose is crucial. Only based on a correct belief, players can make the best responses. Each strategy is the best response to all other strategies in Bayesian Nash equilibrium. In Bayesian games, a type space Ti of player i is introduced, and each Ti has a probability distribution Di. Assume that all players know D1, · · · ,Dn, and the type ti of player i is the outcome drawn from Di independently. Bayesian Nash equilibrium is defined as a strategy profile with which every type of players is maximizing their expected payoffs given other type-contingent strategies. Especially for player i with the strategy si : Ti → Si, a strategy profile {s∗1 , · · · , s∗ n} ∈ S is Bayesian Nash equilibrium if

$$\forall i, t_i \in T_i, s_i \in S_i, s_i \neq s_i^* : E_{D_{-i}}[u_i(t_i, s_i^*(t_i), s_{-i}^*(t_{-i}))] > E_{D_{-i}}[u_i(t_i, s_i(t_i), s_{-i}^*(t_{-i}))]$$

However, Nash equilibrium may not be Pareto optimal from the global view. Nash equilibrium checks whether a profitable payoff exists when other payoffs are unchanged. Pareto efficiency examines whether a profitable payoff exists without reducing others payoffs. Therefore, for the egocentric agents in cloud market, Nash equilibrium is more suitable than Perato efficiency to evaluate the allocation decisions.

2.9. Motivation from equilibrium allocation

Market mechanism has been proven as a useful approach for many resource management systems, such as agent system [16], telecommunication networks [14], data mining [15], cluster computing [17] and grid computing [18]. In these systems, various management contexts including bandwidth pricing, TCP congestion control, contents delivery and routing are studied. The conventional market models are further categorized by modes of pricing and transition, including commodity model, contract model, bartering model and auction-related models. These models have their own strengths and weaknesses, so they are applied in different application scenarios. Stuer [19] preferred the commodity model, in which the price is balanced by analyzing the demand and supply values from the market participants. Stratford [20] developed an architecture based on the contract model. This model uses dynamic pricing as a congestion feedback mechanism, and enables system policy to control adaptation decisions, so it supports scalability and application specific adaptation. The bartering model [21] is studied as an alternative, because it realizes mutual resource cooperation in the way that one user obtains remote resources for free, letting others use its privacy resource in return. Moreover, various auction models including bid-wined and bid-shared schemes are widely used for resource management. In the bid-wined model, the highest bidder wins the resources and pays as much as the bid. Lynar [22] evaluates three types of bid-wined auctions and finds out the substantial difference in completion time and energy consumption. The bid-shared auction is inclined to solve cooperative problems which belong to a single administrative domain [23], so the companies as cloud suppliers are in accordance with bid-shared auction. so it supports scalability and application specific adaptation. The bartering model [24] is studied as an alternative, because it realizes mutual resource cooperation in the way that one user obtains remote resources for free, letting others use its privacy resource in return. Moreover, various auction models including bid-wined and bid-shared schemes are widely used

for resource management. In the bid-wined model, the highest bidder wins the resources and pays as much as the bid. Lynar [25] evaluates three types of bid-wined auctions and finds out the substantial difference in completion time and energy consumption. The bid-shared auction is inclined to solve cooperative problems which belong to a single administrative domain [26], so the companies as cloud suppliers are in accordance with bid-shared auction. Nash equilibrium analyzes how individuals make rational decisions in non-cooperative games, so it is used in the research of allocation strategies in mobile-agent and grid systems. Galstyan [27] studied a minimalist decentralized algorithm for resource allocation in grid environment. The agents using a particular resource are rewarded if their number does not exceed the resource capacity, and penalized otherwise. Thus, the system can fully utilize resources by adjusting its capacity. The limitation of this algorithm is that the number of agents can not be too large. Bredin [28] developed decentralized negotiation strategies in auctioning divisible resources. Mobile agents are given budget constraints in advance, and plan expenditures in the series of tasks to complete. Maheswaran [29] generalized Bredin's result, and investigated a divisible auction structure that allows for a quasi linear characterization of a wide variety of agent tasks. He also proved that the auction has a unique Nash equilibrium. This fundamental research inspires us to solve the allocation problem by sharing, rather than assigning an entire resource to a single user in a cloud market. A common flaw exists in both studies, that

is, their decentralized models idealize the competitive environment. The mobile agents know other competitors' information well, which is difficult to achieve in a real market. Kwok [30] pioneered the consideration of a hierarchical game theoretic model in grids. Kwok also derived both equilibrium and optimal strategies for general cases, based on a skillful utility function. This result can serve as valuable reference for designing appropriate strategies in a grid, and even in an exchanging cloud. An [31] presented a proportional resource allocation mechanism for multi-agent systems and provided analysis of the existence of equilibrium. Trading agents can optimize resource allocation results by updating beliefs and resubmitting bids. The upturn includes more variables (for example budget constraints and time constraints) into the current mechanism. Wei [32] considered a cloud-based resource provisioning problem, taking both optimization and fairness into account. Wei used approximated methods to solve independent optimization by binary integer programming, and to minimize their efficiency losses by an evolutionary game theoretic mechanism. However, the approximation ratio and time complexity should be further reduced to make the solution more practical.

2.10. Game-theoretical allocation model

Virtualization technology hides heterogeneous configuration details fromcustomers, and makes computation services functionally identical. Cloud users only need to choose a proper computing capacity that meets their requirements and pay according to the amount of usage. Cloud suppliers offer their customers more than one payment solution. For example, Amazon EC2 provides three different purchasing options: on-demand model, reserved model and spot model. Each model has different applicable scopes and limitations[33]. In order to satisfy more specific demands, we study bid-based model as a complementary payment option to give users the flexibility to optimize their costs.

2.11. Bid-shared auction

In a cloud market, there are N users asking for services, each having a sequence of tasks to complete. The maximum number of tasks is K. Cloud provider entirely virtualizes K resources, each of which can render a specific service with a fixed finite capacity C. C = [C1,C2, . . . ,CK] (4.11) We characterize one task by its size, which means the amount of computing capability required to complete the task.

$$\mathbf{q} = \begin{bmatrix} q_1^1 & \cdots & q_k^1 & \cdots & q_K^1 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ q_1^i & \cdots & q_k^i & \cdots & q_K^i \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ q_1^N & \cdots & q_k^N & \cdots & q_K^N \end{bmatrix}$$

Not all users have the same task itinerary, the size of an inexistent task is zero in the above matrix q. If a task qik can occupy its corresponding resource Ck, the computation is processed fastest, at a speed of ωik

$\omega_k^i = q_k^i / C_k$. However, in our model, resource capacity is never for exclusive use but shared by multi users. It is reasonable and fair that resource partition is proportional to the user's outlay. We assume that a resource is always fully utilized and unaffected by how it is partitioned among users. In the real commodity market, consumers needing the same commodity are competitors, and are reluctant to cooperate with each other. Thus, resource allocation in clouds is a noncooperative allocation problem. Every user has a bidding function, which decides the bid in any round considering task size, priority, QoS requirement, budget and deadline. The repeated bidding behavior is considered as a stochastic process indexed by a discrete time set. The outputs are random variables that have certain distributions, when these above deterministic arguments and time are fixed.

$$\{B^i(k), k \in (1, 2, \ldots, K)\}$$

Where B^i is the money that a user is willing to pay for one unit of resource per second. User i bids for task k at price b^i_k, which can be treated as a sample for B^i

$$\mathbf{B} = \begin{bmatrix} B^1 \\ \vdots \\ B^i \\ \vdots \\ B^N \end{bmatrix} = \begin{bmatrix} b_1^1 & \cdots & b_k^1 & \cdots & b_K^1 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ b_1^i & \cdots & b_k^i & \cdots & b_K^i \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ b_1^N & \cdots & b_k^N & \cdots & b_K^N \end{bmatrix}$$

The sum Θ_k of total bids for task k indicates the resource price.

$$\Theta_k = \sum_{i=1}^{N} b_k^i$$

Meanwhile, $\theta_k^{-i} = \sum_{j \neq i}^{N} b_k^j$ is given as the sum of other bids except bid b^i_k

Bid-shared model indicates that resource k obtained by the user i is proportional to his bid price. The portion is $x_k^i = \frac{b_k^i}{\sum_{i=1}^{N} b_k^i}$, and obviously, $\forall k, \sum_{i=1}^{N} x_k^i = 1.$ Time spent on task k is defined by

$$t_k^i = \frac{q_k^i}{C_k x_k^i} = \omega_k^i + \omega_k^i \frac{\theta_k^{-i}}{b_k^i}$$

Cost taken to complete task k is

$$e_k^i = b_k^i t_k^i = \omega_k^i \theta_k^{-i} + \omega_k^i b_k^i$$

Two illuminations are obtained from the time and cost functions.

2.12. Non-cooperative game

Both time and expenditure depend not only on b^i_k that an user is willing to pay, but also on θ_k^{-i} that other competitors will pay. We therefore construct a non-cooperative game to analyze the bid-shared model. In games, the set of players is denoted by N cloud users. Any player i independently chooses the strategy b^i_k from his strategy space B^i. The preference is determined by payoff, for example, we take computationtime t_k^i as the payoff. Every player wishes his tasks to becomputed as fast as possible, so the payoff value is the lower the better. Regardless of the valueof θ_k^{-i} the dominated strategy of player i is a low value of b^i_k if he wants to get the optimal payoff. On the contrary, when we choose cost as the game payoff, the dominated strategy is high value of b^i_k which is different from the former dominated strategy. This difference alerts us that the payoff must be carefully selected in order to indicate the outcome preference of a game. Absolute dependence on time or money is unreasonable. We combine cost expense and computation time into an aggregate quantity, which stands for the total amount of substituted consumption. Similar to utility function discussed above, constant elasticity of substitution function indicates the players' payoff.

$$\mathcal{C} = \frac{\rho_e \ln \sum_{k=1}^K e_k^i + \rho_t \ln \sum_{k=1}^K t_k^i}{\rho_e + \rho_t}$$

Where ρe, ρt are the output elasticities of cost and time, respectively.

2.13. Bid function

In a cloud market, customers are rational decision makers who seek to minimize their consumption, and have constraints of cost E = [E1,E2, . . . ,EN] and time T = T[T1, T2, . . . , TN].With a limited budget Ei and deadline Ti, the optimal object function of user i is:

$$\begin{aligned} \text{Min} \quad & \mathcal{C} \\ \text{s.t.} \quad & \sum_{k=1}^K e_k^i \le E^i \\ & \sum_{k=1}^K t_k^i \le T^i \end{aligned}$$

The Hamilton equation is built by introducing the Lagrangian

$$\begin{aligned} \mathcal{L} &= \frac{\rho_e \ln \sum_{k=1}^K e_k^i + \rho_t \ln \sum_{k=1}^K t_k^i}{\rho_e + \rho_t} \\ &+ \lambda_e^i \Big(\sum_{k=1}^K e_k^i - E^i\Big) + \lambda_t^i \Big(\sum_{k=1}^K t_k^i - T^i\Big) \\ &= \frac{\rho_e \ln \sum_{k=1}^K (\omega_k^i \theta_k^{-i} + \omega_k^i b_k^i) + \rho_t \ln \sum_{k=1}^K (\omega_k^i + \omega_k^i \frac{\theta_k^{-i}}{b_k^i})}{\rho_e + \rho_t} \\ &+ \lambda_e^i \Big(\sum_{k=1}^K (\omega_k^i \theta_k^{-i} + \omega_k^i b_k^i) - E^i\Big) + \lambda_t^i \Big(\sum_{k=1}^K (\omega_k^i + \omega_k^i \frac{\theta_k^{-i}}{b_k^i}) - T^i\Big) \end{aligned}$$

L is a function of three variables of b_k^i, λ_e^i and λ_t^i. To obtain the dynamic extreme point, gradient vector is set to zero.

$$\nabla \mathcal{L}(b_k^i, \lambda_e^i, \lambda_t^i) = 0$$

1. Take partial derivative with respect to b_k^i:

$$\frac{\partial \mathcal{L}}{\partial b_k^i} = \frac{\rho_e}{\rho_e + \rho_t} \frac{\omega_k^i}{\sum e_k^i} - \frac{\rho_t}{\rho_e + \rho_t} \frac{\omega_k^i \theta_k^{-i}}{\sum t_k^i {b_k^i}^2} + \lambda_e^i \omega_k^i - \lambda_t^i \frac{\omega_k^i \theta_k^{-i}}{{b_k^i}^2} = 0$$

which gives

$$\frac{\frac{\rho_e}{\sum e_k^i} + \lambda_e^i}{\frac{\rho_t}{\sum t_k^i} + \lambda_t^i} = \frac{\theta_k^{-i}}{(b_k^i)^2}$$

A similar result is obtained by setting the gradient of L at b^i_j to zero $\frac{\partial \mathcal{L}}{\partial b_j^i} = 0$,

$$\frac{\frac{\rho_e}{\sum e_k^i} + \lambda_e^i}{\frac{\rho_t}{\sum t_k^i} + \lambda_t^i} = \frac{\theta_j^{-i}}{(b_j^i)^2}$$

For user i, the capital sum $\sum e_k^i$ and time sum $\sum t_k^i$ remain the same for any two tasks, we could therefore determine the relationship between any two bids in one task sequence, which is:

$$\frac{\theta_k^{-i}}{(b_k^i)^2} = \frac{\theta_j^{-i}}{(b_j^i)^2}$$

Then bid k is expressed by bid j, $b_k^i = b_j^i \sqrt{\frac{\theta_k^{-i}}{\theta_j^{-i}}}$.

2. Take partial derivative with respect to λ_e^i

$$\frac{\partial \mathcal{L}}{\partial \lambda_e^i} = \sum_{k=1}^{K} e_k^i - E^i = \sum_{k=1}^{K} \omega_k^i (b_k^i + \theta_k^{-i}) - E^i = 0$$

Substituting b_j^i for $\sqrt{\frac{\theta_j^{-i}}{\theta_k^{-i}}} b_k^i$ the equation is expanded

$$\sum_{j=1}^{k-1}\omega_j^i\left(\sqrt{\frac{\theta_j^{-i}}{\theta_k^{-i}}}b_k^i+\theta_j^{-i}\right)+\omega_k^i\left(b_k^i+\theta_k^{-i}\right)$$
$$+\sum_{j=k+1}^{K}\omega_j^i\left(\sqrt{\frac{\hat{\theta_j^{-i}}}{\theta_k^{-i}}}b_k^i+\hat{\theta_j^{-i}}\right)-E^i=0$$

Simplifying the above equation, user i will bid for task k at price

$$b_k^i=\frac{E^i-\sum_{j=1}^{k-1}\omega_j^i\theta_j^{-i}-\omega_k^i\theta_k^{-i}-\sum_{j=k+1}^{K}\omega_j^i\hat{\theta_j^{-i}}}{\sum_{j=1}^{k-1}\omega_j^i\sqrt{\frac{\theta_j^{-i}}{\theta_k^{-i}}}+\omega_k^i+\sum_{j=k+1}^{K}\omega_j^i\sqrt{\frac{\hat{\theta_j^{-i}}}{\theta_k^{-i}}}}$$

$$\frac{\partial\mathcal{L}}{\partial\lambda_t^i}=\sum_{k=1}^{K}t_k^i-T^i=\sum_{k=1}^{K}\frac{\omega_k^i(b_k^i+\theta_k^{-i})}{b_k^i}-T^i=0$$

3. Take partial derivative with respect to λ_t^i

The expanded expression is obtained

$$\sum_{j=1}^{k-1}\omega_j^i\left(\frac{\sqrt{\frac{\theta_j^{-i}}{\theta_k^{-i}}}b_k^i+\theta_j^{-i}}{\sqrt{\frac{\theta_j^{-i}}{\theta_k^{-i}}}b_k^i}\right)+\omega_k^i\left(\frac{b_k^i+\theta_k^{-i}}{b_k^i}\right)+\sum_{j=k+1}^{K}\omega_j^i\left(\frac{\sqrt{\frac{\hat{\theta_j^{-i}}}{\theta_k^{-i}}}b_k^i+\hat{\theta_j^{-i}}}{\sqrt{\frac{\hat{\theta_j^{-i}}}{\theta_k^{-i}}}b_k^i}\right)-T^i=0$$

The above equation is further simplified by

$$b_k^i=\frac{\sum_{j=1}^{k-1}\omega_j^i\sqrt{\theta_j^{-i}\theta_k^{-i}}+\omega_k^i\theta_k^{-i}+\sum_{j=k+1}^{K}\omega_j^i\sqrt{\hat{\theta_j^{-i}}\theta_k^{-i}}}{T^i-\sum_{j=1}^{K}\omega_j^i}$$

2.14. Parameter estimation

The existence of Nash Equilibrium with complete information has been proved by Bredin[37]. However, new problems arise when buyers do not intend to expose their bids to other competitors or when they are allowed to join or leave a datacenter from time to time. How does one deal with the lack of information? How do users predict the price trend on the basis of inadequate knowledge? We record historical purchasing prices $\Theta_1,\cdots\Theta_{k-1}$ in past auctions, and then use statistical forecasting method to evaluate the future price. In probability theory, Bayes' theorem shows how the probability of a hypothesis depends on its inverse if observed evidence is given. The posteriori distribution can be calculated from the priori $p(\Theta)$, and its likelihood function $p(\Theta\mid\Theta_k)$ is:

$$p(\Theta\mid\Theta_k)=\frac{p(\Theta_k|\Theta)p(\Theta)}{\int p(\Theta_k|\Theta)p(\Theta)d\Theta}$$

The posteriori hyperparameters $p(\Theta|\Theta_k)$ can be achieved by using the Bayesian learning mechanism, the value of which determines the maximum likelihood prediction of resource price. So future bids are forecasted as:

$$\hat{\theta_{k+1}^{-i}} = \mathrm{E}(\Theta|\Theta_k) - \mathrm{E}(B^i)$$
$$\vdots$$
$$\hat{\theta_K^{-i}} = \mathrm{E}(\Theta|\Theta_{K-1}) - \mathrm{E}(B^i)$$

Three parameters α_k^i, β_k^i and γ_k^i are introduced, which stand for information from other competitors.

$$\alpha_k^i = \sum_{j=1}^{k-1} \omega_j^i \theta_j^{-i} + \sum_{j=k+1}^{K} \omega_j^i \hat{\theta_j^{-i}}$$
$$\beta_k^i = \sum_{j=1}^{k-1} \omega_j^i \sqrt{\theta_j^{-i}} + \sum_{j=k+1}^{K} \omega_j^i \sqrt{\hat{\theta_j^{-i}}}$$
$$\gamma_k^i = \sum_{j=1}^{k-1} \omega_j^i + \sum_{j=k+1}^{K} \omega_j^i$$

deadline is extended, the solid budget curve meets the dashed deadline curve at a lower position. It indicates that the possible bid should be above the solid deadline curve in order to complete all tasks in finite time. For the same reason, if one user holds more funds, the intersection moves right along the solid deadline curve, so the left side of solid budget curve will contain the possible bids. The bid region is surrounded by cross and plus curves. Specifically, the crosses mean that all capital is used up with time remaining, while the pluses mean that deadline is reached with redundant money. Outside this region, there is no feasible bidding solution, which indicates the given constraints are over rigid. Users must loosen either of the two constraints slightly if they still wish to accomplish this impossible mission.

2.15. Cloudsim toolkit

Cloudsim [43] is designed to emulate cloud-based infrastructure and application service, and can be used in research of economy driven resource management policies on large scale cloud computing systems. Researchers benefit from focusing on resource allocation problems without implementation details. These features are not supported by other cloud simulators [43]. We apply Cloudsim as our simulation framework, but make some improvements aiming at the following shortcomings. Firstly, sequential auctions are complemented, accompanied by several specific policies. Secondly, Cloudsim only supports static assignment with predetermined resources and tasks. We realize that multi-users can submit their tasks over time according to certain arrival rate or probability distribution and that resource nodes can freely join or leave cloud datacenter. The assignment in our simulation model is much closer to a real market than before.

2.16. Communication among entities

There are four types of entities to be simulated. CIS Registry provides a database level matchmaking service for mapping application requests to datacenter. Datacenter integrates distributed hardware, database, storage devices, application software and operating systems to build a resource pool, and is in charge of virtualizing applicable computing resources according to users' requests. Cloud users have independent task sequences, and they purchase resources from datacenter to execute tasks. All these users bid according to their economic capabilities and priorities under different constraints. Auctioneer is the middleman in charge of maintaining an open, fair and equitable market environment. In accordance with the rules of market economy, auctioneer fixes an equilibrium price for non-cooperative users to avoid blind competition.

Figure 1 depicts the flow of communication among main entities. At the beginning, datacenter initializes current available hosts, generating provision information and registers in CIS. Meanwhile, cloud users who have new tasks report to auctioneer and queue up in order of arrival time. At regular intervals, auctioneer collects information and requests datacenter to virtualize

corresponding resources. Once virtual machines are ready according to users' service requirements, datacenter sends the provision information to the auctioneer, and successive auctions start. In each auction stage, users ask the auctioneer individually about configuration information such as virtual machine provision policy, time zone, bandwidth, residual computing processors, and bid according to their asset valuations. Auctioneer collects all bids then informs users of the sum of bids. Under the game of incomplete information, cloud users only know their own price functions as well as the incurred sum of bids. They dynamically predicate

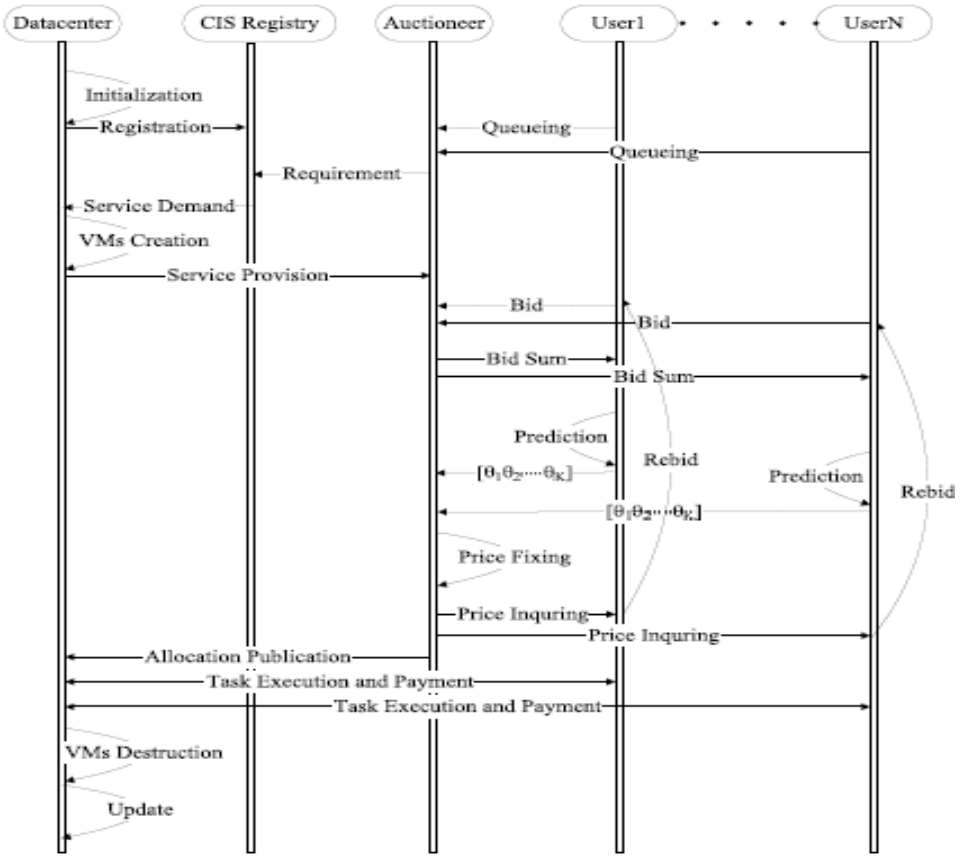

Figure 1. Flowchart of communication among entities.

the future resource price, and update competitors' information $[\theta_1^{-i}, \cdots, \theta_k^{-i}, \hat{\theta_{k+1}^{-i}}, \cdots, \hat{\theta_K^{-i}}]$. Subsequently, holding all price functions auctioneer makes an equilibrium allocation decision and inquires whether everyone is satisfied with the result. If the result is agreeable, auctioneer publishes allocation proportions to datacenter and users. Users then execute their tasks and pay for the resource allocated. At the end, datacenter deletes the used VMs and waits for new service demands.

2.17. Implementation algorithm

Concrete algorithms for users and auctioneer are explained in more details by Algorithm 1 and Algorithm 2. From an user's point of view, after task submission, observer focuses on analyzing the received messages that prescribe user's next move. If auctioneer announces a new auction, user adds it to the auction list. If bids are called, an appropriate bid is calculated and reported to auctioneer. If user receives the message calling for parameters, he examines the historical prices and estimates the future bid sum by Bayesian learning mechanism, then sends information back.

Finally, if user receives resource price and proportion, he immediately updates his price list and begins to execute the task.

Algorithm 1 User i bidding algorithm

```
submit tasks to auctioneer
if observer receives message of inform start then
    add current auction
end if
if observer receives message of call for bids then
    set {b_1^i, ..., b_{k-1}^i} ← b_k^i
    send message of proposal to auctioneer
end if
if observer receives message of call for parameters then
    inquiry historical price θ_1^{-i}, ..., θ_k^{-i}
    forecast future price θ̂_{k+1}^{-i}, ..., θ̂_K^{-i}
    send message of competitors information to auctioneer
end if
if observer receives message of resource price then
    {Θ_1, ..., Θ_{k-1}} ← Θ_k
    send message of task execution to resource
    delete current auction
end if
```

Froman auctioneer's perspective, a new auction is triggered off whenever a new type of task arrives. Once an auction begins, auctioneer broadcasts the bid calling message to current users. As soon as all proposals arrive, auctioneer informs users the sum Θ_k. Similarly, auctioneer collects bidding function parameters from all the bidders, and then decides a reasonable bound. If the bound is too narrow, poor users quit gambling. Resource price is modified repeatedly until the difference between $\sum h_k^i$ and Θ_k is less than a predetermined threshold. Once the equilibrium price is found, allocation proportions are broadcast to all cloud users. After that auctioneer deletes the current auction and waits for a new task request.

Algorithm 2 Auctioneer allocation algorithm

Require: $N \geq 2$

initialize auctioneer
while auction k **do**
  set bidders to auction k
  broadcast message to call for bids
  while bidder's proposal arrives **do**
    collect proposal message from bidder
  end while
  broadcast message to inform Θ_k
  while bidder's parameter arrives **do**
    collect parameter message from bidder
  end while
  while bidders disagree proportion **do**
    for all cloud users **do**
      build new bid function h_k^i
    end for
    $difference = \sum h_k^i - \Theta_k$
    if $difference > threshold$ **then**
      $\Theta_k = \sum h_k^i$
    else
      exit
    end if
    update vector $[\theta_1^{-i}, \cdots, \theta_k^{-i}, \hat{\theta_{k+1}^{-i}}, \cdots, \hat{\theta_K^{-i}}]$
  end while
  broadcast message to inform resource price
  stop the current and wait for a new auction
end while
delete auctioneer

2.18. Evaluation

2.18.1. Experiment Setup

We now present the simulated experiments in Cloudsim. Datecenter is usually composed of a set of hosts, each of which represents a physical computing node in the cloud. In our simulation, 60 hosts are created with heterogeneous configuration characteristics randomly picked in Table1

Table 1. Resource characteristics

Characteristics	Parameters
Machine architecture	x86, Sun Ultra, PowerPC
Operating system	Linux, Windows, Solaris
Virtual machine monitor	Xen, UML, VMware
Number of PE	2, 4, 8
MIPS rating per PE	100, 200, 300, 400
Memory	512M, 1024M,2048MB
Storage	160G, 320G, 500G
Bandwidth	128M, 256M, 512M

To model cloud users, we create application tasks that contain information related to execution details such as task processing requirements, disk I/O operations and the size of input files. We simulate 32 users in a cloud system, and each with an exponentially distributed number of tasks. Two common distributions, Normal and Pareto, signify preferences about the prices.

2.18.2. Nash equilibrium allocation

Firstly, normal distribution is used to describe the financial capability of the users. Bidding function Bi has mean μi and variance σ2. We choose one user as our observable object, and assign a mean purchasing price of 10$/s and bid variance of 0.1. Other mean bids are generated randomly in the range of 1-100$/s. This user is unaware of other economic situations, but keeps on estimating others from their prior behaviors. Figure 2 illustrates how closing price changes as time goes by. We conclude that budget exerts a huge influence on preliminary equilibrium price, because selfish but rational users always wish to seek extra benefits from others. With limited budget, the user will behave conservatively at the initial stages, to avoid overrunning the budget and to save enough money to complete remaining tasks. Therefore, in the beginning, the equilibrium price is lower than the mean price. On the contrary, if the user has sufficient capital, he is eager to improve current payment to get a larger proportion. Competition leads equilibrium price to rise, higher than the anticipated cost. However, with the money available for the current job decreasing, the user

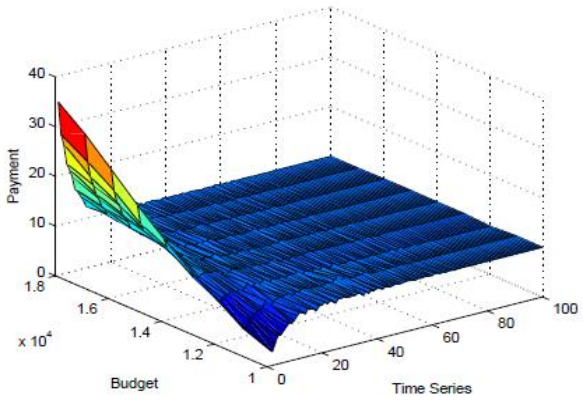

Figure 2. Convergence of Nash equilibrium bid.

becomes less aggressive. As bidding is underway, price will gradually converge to the original mean value.

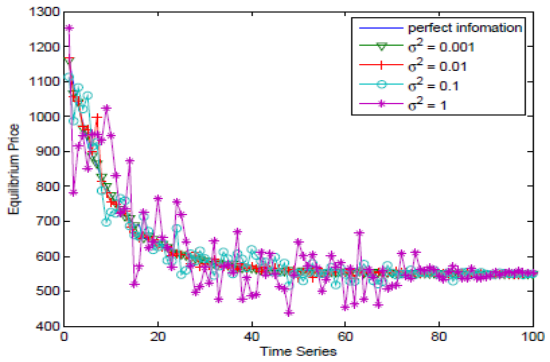

Figure 3. Prediction of resource price

Next the accuracy of Bayesian learning prediction is evaluated when the cloud market is full of uncertainties, such as insufficient common knowledge and on-line task submitting. Figure 3 exhibits the predication of resource price in dynamic game of incomplete information. If the common knowledge is insufficient, the user experientially predicts other bids using the published equilibrium prices. When the bidding variance is low, no more than 0.01, the estimation works quite well. Our policy differs a little from the scheme that hypothesizes that all users' information is fixed and public. If users perform unstably in the gambling process and the offered bids are more random, accurate price forecast becomes difficult. Provided

that rivals' information is learned iteratively, experiment results show that resource price still converges to the equilibrium price stage by stage.

2.18.3. Comparison of forecasting methods

Three forecasting methods are compared, including Bayesian learning, historical averaging and last-value following.

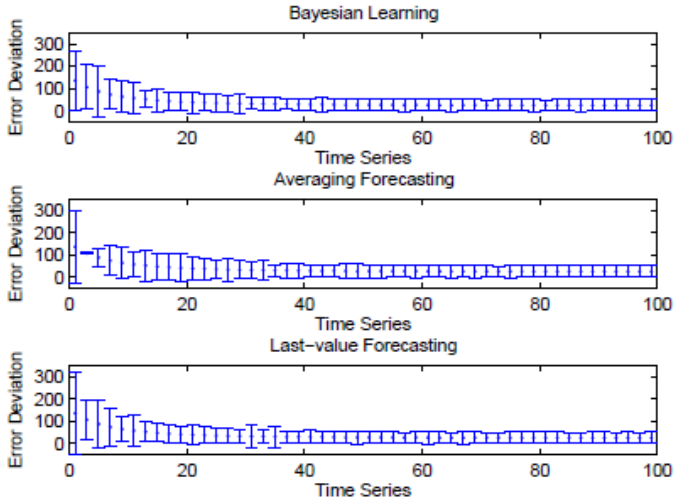

Figure 4. Forecast errors with normal distribution.

Figure 4 shows the standard deviations of three forecast methods versus time series. All three forecasting methods are able to converge to the result with perfect information, as long as the user keeps on training his belief of others' bid functions over time. The cases with abundant budgets are examined. Some users would like to increase bids to get more resource, so the price keeps rising, to much higher than the estimated bid. If all the historical prices are used for prediction, the history averaging method behaves poorly at the beginning of auctions, and is less stable than other two methods. Compared with the last-value method, Bayesian learning converges in a smoother manner, because historical prices are used to calculate the likelihood function rather than simply following the price in the previous auction as last-value method. Now we apply another distribution, Pareto, to express users' bidding rules, meanwhile keeping other experiment setups the same. A similar conclusion can be reached in Figure 5, except that the worst forecast is last-value method. The result is due to the attribute of Pareto distribution. The Pareto principle stands for the probability that the variable is greater than its minimum, while normal distribution reveals how close data clusters are around its mean. For one specific round of bidding, it's more difficult to estimate the precise value with Pareto distribution than with normal distribution. In other words, the more historical data is accumulated,

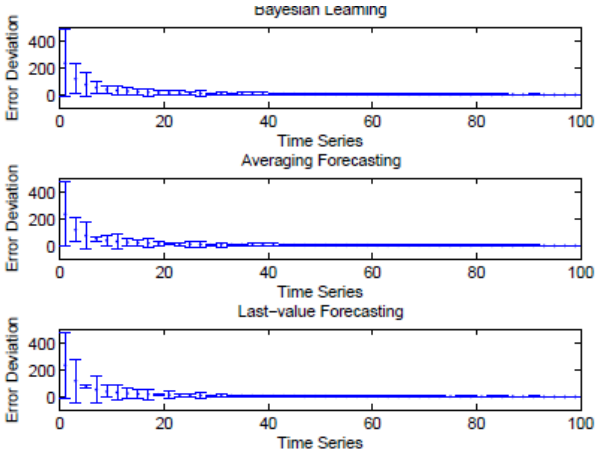

Figure 5. Forecast errors with Pareto distribution.

the more accurate the forecast would be. In Figure 5, convergence of Bayesian learning is still the most stable one of the three schemes. As a result, it is recommended as a forecast method in practical applications.

3. CONCLUSIONS

In this chapter, we solve the resource allocation problem in the user-level of cloud scheduling. We survey game theory, covering the different classes of games and their applications, payoff choice and utility function, as well as strategic choice and Nash equilibrium. Based on that, we build a non-cooperative game to solve the multi-user allocation problem in cloud scenario. The scheduling model includes bid-shared auction, user strategy (bid function), price forecasting and equilibrium analysis. We propose game theoretical algorithms for user bidding and auctioneer pricing, and then supplement bid-shared auction schemes in a cloud simulation framework, named Cloudsim, in order to realize sequential games. Results show that resource allocation reaches Nash equilibrium among non-cooperative users when common knowledge is insufficient and that Bayesian learning forecast has the best and most stable performance. Our algorithms can support financially smart customers with an effective forecasting method, and can help auctioneer decide an equilibrium resource price. Therefore, they are potential to solve resource allocation problems in cloud computing.

REFERENCES

[1] Foster, C. Kesselman, and S. Tuecke. The anatomy of the grid - enabling scalable virtual organizations. International Journal of Supercomputer Applications, 15:2001, 2001.
[2] D. D. Roure, M. A. Baker, N. R. Jennings, and N. R. Shadbolt. The evolution of the grid. In Proceedings of Grid Computing: Making the Global Infrastructure a Reality, pages 65–100. John Wiley & Sons, 2004.
[3] M. Parashar and S. Hariri. Autonomic computing: An overview. In Proceedings of Unconventional Programming Paradigms, pages 247–259. Springer Verlag, 2005.
[4] J. O. Kephart and D. M. Chess. The vision of autonomic computing. Computer, 36:41–50, January 2003.
[5] M. Cafaro and G. Aloisio. Grids, Clouds and Virtualization. Springer-Verlag New York, Inc., 1st edition, 2010.

[6] M. Armbrust, A. Fox, and R. Griffith. Above the clouds: A berkeley view of cloud computing. Technical Report UCB/EECS-2009-28, EECS Department, University of California, Berkeley, Feb 2009.
[7] R. Buyya, D. Abramson, J. Giddy, and H. Stockinger. Economic models for resource management and scheduling in grid computing. Concurrency and Computation: Practice and Experience, 14:1507–1542, 2002.
[8] Galstyan, S. Kolar, and K. Lerman. Resource allocation games with changing resource capacities. In Proceedings of the International Conference on Autonomous Agents and Multi-Agent Systems, pages 145–152. ACM Press, 2003.
[9] J. Bredin, D. Kotz, D. Rus, R. T. Maheswaran, C. Imer, and T. Basar. Computational markets to regulate mobile-agent systems. Autonomous Agents and Multi-Agent Systems, 6:235–263, 2003.
[10] R. T. Maheswaran and T. Basar. Nash equilibrium and decentralized negotiation in auctioning divisible resources. Group Decision and Negotiation, 12:361–395, 2003.
[11] S. Khan and I. Ahmad. Non-cooperative, semi-cooperative, and cooperative games-based grid resource allocation. Parallel and Distributed Processing Symposium, 0:101, 2006.
[12] B. An, C. Miao, and Z. Shen. Market based resource allocation with incomplete information. In Proceedings of the 20th International Joint Conference on Artifical Intelligence, pages 1193– 1198. Morgan Kaufmann Publishers Inc., 2007.
[13] G. Wei, A. Vasilakos, Y. Zheng, and N. Xiong. A game-theoretic method of fair resource allocation forcloud computing services. The Journal of Supercomputing, 54:1–18, 2009.
[14] Q. Fan, Q. Wu, F. Magoul`es, N. Xiong, A. V. Vasilakos, and Y. He. Game and balance multicast architecture algorithms for sensor grid. Sensors, 9(9):7177–7202, 2009.
[15] F. Teng and F. Magoul`es. A new game theoretical ressource allocation algorithm for cloud computing.In Proceedings of Advances in Grid and Pervasive Computing, volume 6104, pages 321–330. Springer, 2010.
[16] R. Gibbons. A Primer in Game Theory. Pearson Higher Education, 1992.
[17] T. W. Sandholm. Distributed rational decision making. Multiagent systems: a modern approach to distributed artificial intelligence, 37:201–258, 1999.
[18] M. A. Gibney, N. R. Jennings, N. J. Vriend, and J.-M. Griffiths. Market-based call routing in telecommunications networks using adaptive pricing and real bidding. In Proceedings of the Third International Workshop on Intelligent Agents for Telecommunication Applications, pages 46–61. Springer-Verlag, 1999.
[19] L. Joita, O. F. Rana, F. Freitag, I. Chao, P. Chacin, L. Navarro, and O. Ardaiz. A catallactic market for data mining services. Future Generation Computer Systems, 23(1):146–153, 2007.
[20] B. N. Chun and D. E. Culler. User-centric performance analysis of market-based cluster batch schedulers. In Proceedings of the 2nd IEEE/ACM International Symposium on Cluster Computing and the Grid, page 30, Washington, DC, USA, 2002. IEEE Computer Society.
[21] T.-M.-H. Nguyen and F. Magoul`es. Autonomic data management system in grid environment. Journal of Algorithms & Computational Technology, 3:155–177, 2009.
[22] G. Stuer, K. Vanmechelen, and J. Broeckhove. A commodity market algorithm for pricing substitutable grid resources. Future Generation Computer Systems., 23(5):688–701, 2007.
[23] N. Stratford and R. Mortier. An economic approach to adaptive resource management. In Proceedingsof the The Seventh Workshop on Hot Topics in Operating Systems, pages 142 - 147. IEEE Computer Society, 1999.
[24] C. Ozturan. Resource bartering in data grids. Science of Computer Programming, 12(3):155–168, 2004.
[25] T. M. Lynar, R. D. Herbert, and S. Simon. Auction resource allocation mechanisms in grids of heterogeneous computers. WSEAS Transactions on Computers, 8(10):1671–1680, 2009.
[26] R. Buyya, D. Abramson, J. Giddy, and H. Stockinger. Economic models for resource management and scheduling in grid computing. Concurrency and Computation: Practice and Experience, 14:1507–1542, 2002.

[27] Y. kwong Kwok, S. Song, and K. Hwang. Selfish grid computing: Game-theoretic modeling and nash performance results. In Proceedings of International Symposium on Cluster Computing and the Grid, pages 9–12, 2005.
[28] S. Yi, D. Kondo, and A. Andrzejak. Reducing costs of spot instances via checkpointing in the amazon elastic compute cloud. In Proceedings of IEEE International Conference on Cloud Computing, pages 236–243, 2010.
[29] R. Buyya, R. Ranjan, and R. N. Calheiros. Modeling and simulation of scalable cloud computing environments and the cloudsim toolkit: Challenges and opportunities. In Proceedings of the 7thHigh Performance Computing and Simulation Conference. IEEE Computer Society, 2009.
[30] M. Armbrust, A. Fox, R. Griffith, A. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, M. Zaharia, "A View of Cloud Computing", Communication of the ACM Magazine, New York, USA, pp. 50–58, 2010.
[31] A.Kaleeswaran, V. Ramasamy, P. Vivekanandan, Dynamic scheduling of data using genetic algorithm in cloud computing, international journal of advances in engineering & technology, 2013.
[32] M. Armbrust, A. Fox, R. Griffith, A. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, M. Zaharia, "A View of Cloud Computing", Communication of the ACM Magazine, New York, USA, pp. 50–58, 2010.