International Journal of Grid and Distributed Computing
Vol. 9, No. 8, (2016), pp.385-402
http://dx.doi.org/10.14257/ijgdc.2016.9.8.33

An Adaptive Learning System based on Cloud Computing: Implementation and Evaluation of BDS

Yi Liao[1,2], Lei Huang[1], Hang Zhou[2] and Bo Li[2]

[1]*School of Economics and Management, Beijing Jiaotong University, Beijing 100044, China*
[2]*Beijing Educational Network and Information Center, Beijing Municipal Commission of Education, Beijing 100038, China*
11113162@bjtu.edu.cn

Abstract

In this paper, we report the design and implementation of an adaptive individualized e-learning environment based on cloud computing named BDS (Beijing Digital School). With its five-layer architecture and its use of cloud computing technology, BDS possesses the characteristics of both a massive access system and an adaptive e-learning environment. This paper focuses on investigating three distributed solutions proposed for load balancing: an active clustering algorithm, a random algorithm and a honey bee foraging algorithm. An experiment comparing 2 groups of students was conducted to evaluate the impact of the proposed environment on learning performance, and the results were analyzed to determine the most effective strategy.

Keywords*: cloud computing; adaptive learning; load balancing; honey bee foraging;*

1. Introduction

Today, online education environments have progressed in parallel with the rapid development of information technology. E-learning systems have increased in value as the spread of the Internet has enhanced the growth and popularity of computer networks (Esichaikul, V., Supaporn, L., & Clemens, B, 2011) [1]. Traditional e-learning systems have been criticized for their limited nature of presenting the same content to each student below a predetermined performance threshold (Brusilovsky, 2001; Berge, 2002) [2,3], without considering the students' individual learning styles and abilities. Because of these restrictions of traditional e-learning systems, many students cannot effectively satisfy online course requirements and take control of their learning (Berge, 2002; Picciano, 2001; Saba, 2002) [3-5]. The dissatisfaction with these limitations has led to the birth of ALS (adaptive learning systems), which provide a personalized learning environment for each student by adapting both the presentation and progression of the course content (Retalis & Papasalouros, 2005) [6]. An ALS is usually a web-based application that is developed following a "one-size-fits-all" approach (Brusilovsky, 2001; Brusilovsky & Peylo, 2003) [2,7]. Unlike traditional e-learning systems, an ALS creates adaptive content and adaptive learning scenarios suitable for each student. The key concept of an ALS is to determine the individual differences exhibited by each student, such as their levels of knowledge about a topic, their preferences and their learning styles (Romero, Ventura, Zafra, & de Bra, 2009) [8].

As regards the cloud computing, it was defined by the National Institution of Standards and Technology (NIST) as "a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources that can be rapidly provisioned and released with minimal management effort or service-provider interaction" (Owens, 2010) [9]. These computing resources (such as hardware, development platforms and/or

ISSN: 2005-4262 IJGDC

services) can be dynamically reconfigured to adjust to a variable load (scale), allowing also for optimum resource utilization (Vaquero *et al*., 2008) [10].

The purpose of this study was to build an effective ALS based on cloud computing. We designed and developed an individualized e-learning environment named BDS (Beijing Digital School) [1] to service elementary schools and secondary schools. BDS must satisfy the individual learning needs of Beijing's 1.45 million students; therefore, cloud computing technology was adopted. We focused on determining the individual preferences of the students and arranging course learning activities according to these preferences. We introduced cloud computing technology into the adaptive system to solve the problem, which is related to personalized learning and access to a large number of students.

2. Related Work

Adaptive learning accounts for students' different learning styles and ability levels to improve their performance. In other words, adaptive learning refers to the dynamic adjustment of instruction to advance students' personal development; for this purpose, an ALS provides different learning paths and content to different students. An adaptive learning model is typically used to offer different types of learning content and learning activities to different students. It contains three modules: the learner model, the domain model and the adaptive engine [11]. The application of ALS is very widespread, especially in higher education (Carver *et al*., 1999; Gilbert & Han, 1999; Moore, Stewart, Zakaria, & Brailsford, 2003; Wolf, 2003; Brown, 2007; Brown, Brailsford, Fisher, & Moore, 2009; Mustafa & Sharif, 2011) [12-18]. However, a small number of studies have been conducted regarding the use of ALS in basic education. Brown *et al*. (2007) developed a system named DEUS, with a focus on the primary school level, to teach the life cycle of flowering plants in a biology course[19]. Dolenc *et al*. (2015) designed TECH8 model in lower secondary schools [20]. Bakiet *et al*. (2010) mainly focused on the role of ALS in high school mathematics classrooms [21]. Özyurt *et al*. (2013) designed an adaptive and intelligent individualized e-learning system named UZWEBMAT based on learning style for secondary school mathematics curricula [22]. There have been almost no studies that have focused on ALS for basic education in China So design and evaluation of an adaptive and intelligent ALS is what students need in China. This study is expected to fill this gap. Due to characteristics of primary and secondary students, ALS is more ease their comprehension and makes the learning process fun.

Cloud computing is a distributed paradigm based on the Internet that is designed for the remote sharing and usage of various resources and services for purposes such as storage and computing. As the size of the cloud scales up, it becomes necessary for the cloud computing service provider to handle a massive number of requests. Load balancing plays a vital role in determining the overall performance of the system. The three most commonly used load balancing algorithms are active clustering algorithm, random algorithm and honey bee foraging algorithm. The first one used system rewiring to improve the load balancing performance, the second sought to engineer a desired global outcome from biased random sampling, while the third was directly based on a naturally occurring phenomenon, honey bee foraging. Honey bee foraging algorithm was proposed by Karaboga in 2005 [23], which is motivated by the biological behavior of bees in searching for food. It has been applied in many fields. For example, it has been used as a block-matching algorithm for motion estimation [24] as well as to address the flow-shop scheduling problem [25], the leaf-constrained minimum spanning tree problem [26], inverse analysis problems [27]; it has also been used in digital signal processing [28] and authentication systems [29] as well as in various other optimization problems [30].

[1] Internet access to BDS: http://www.bdschool.cn/

Honeybee-based load balancing approach's primary advantage is that the global search capability of the algorithm is implemented based on a neighborhood source production mechanism [31]. It gives better performance when a diverse population of service types is required. Randles *et al*. (2010) point out that honeybee algorithm has maximum throughput with increased system diversity as compared to other two algorithms [32]. In load balancing virtual servers offering services is similar with the honey bees' behavior. Every server requiring services calculates the profit and posts it on its advert board. The servers interested in serving the request also calculate their profit and compare it with the colony profit. If case of high colony profit interested server serves the current virtual server otherwise returns to the scout behavior *i.e.* to choose another server randomly. The introduction of cloud computing technology, such as load-balancing techniques, into an ALS could provide a solution to address the need to provide access to a large number of users.

3. BDS Based on Cloud Computing

Cloud computing services are a form of public participation in an Internet-based computing model. Such services provide a dynamic and scalable resource pool that contains a large number of available virtual resources [33]. These virtual resources are dynamically allocated depending on the load to improve the efficiency of resource utilization [34]. First, a resource pool of computing tasks is assigned, and the application system can then determine the corresponding storage space, computing power and other services that must be allocated in accordance with the task requirements. Through the integration of IT resources and a variety of centralized products and equipment, cloud computing optimizes IT service processes to provide fast, secure, and on-demand technological services.

To address many of the issues encountered in ALS, including the existence of a single point of failure, load imbalance, the need for centralized access to resources, the waste represented by idle equipment resources and others, we adopted cloud computing technology for the development of BDS. Virtual integration server software is used to form a virtual resource pool, comprising a number of virtual machines serving as virtual server solutions. These virtual machines share and thereby optimize resources. The system uses a policy-driven approach for resource allocation.

The system provides resource control functionality through embedded software to ensure security; in this software, a pre-allocation strategy is automatically applied [35]for the purpose of controlling computing, storage, and networking resources as well as other customized client data and resources to achieve dynamic resource allocation from the target aggregated resource pool. As the application load increases, the software determines the application priority. This priority assessment is based on defined resource allocation rules, subject to confirmation, and drives the re-allocation of resources to the virtual machines. There are usually two possible approaches; one is to migrate applications to a server that is richer in resources, and the other is to migrate virtual machines to other applications on an idle server, thereby freeing up more resources on the local machine. This is equivalent to increasing the resources available to the virtual machine [36].

Virtual data centers provide the main services required for cloud computing, including computing, storage and network connectivity. Infrastructure services can be completely separated from the underlying hardware, whereas integrated perimeter protection, port-level firewalls, and NAT and DHCP services provide virtualization security in accordance with the standards required for interface implementation. The 76 servers that form the hardware foundation for cloud computing in the BDS platform are located at the information centers of each county and city. At run time, the processing capacity for each core process of BDS is increased as necessary based on the client's requirements and the data traffic. The platform must utilize multi-server collaboration to improve the available

processing power and the ability of the computer system to meet user needs. Thus, we solved the problems of load balancing and server utilization problems by designing the dynamic migration mechanism used by the cloud computing platform in accordance with the characteristics of the system itself.

3.1. Dynamic Migration Mechanism

When there is load disparity between nodes or when many nodes are operating with low resource utilization when the system is running, several problems may be encountered. Load disparity will result in certain nodes bearing excessively high loads, which will prevent the services they provide from achieving optimal performance, whereas the loads on other nodes will be too low to make full use of the available resources, resulting in waste. To improve resource utilization and reduce energy consumption, it is necessary to shift the loads from overly loaded nodes onto under loaded nodes to make use of their idle resources. To this end, loads are transferred in accordance with a load-balancing mechanism that dynamically migrates virtual machines when the system load is not balanced.

There are two common strategies that may be used for load balancing: migrating the virtual machines with the lowest loads or migrating the virtual machines with the highest loads.

a) Migration of the virtual machines with the lowest loads: In general, the most highly loaded virtual machines are those that are processing the most important data. Therefore, we could consider migrating the virtual machines with the lowest loads out to other physical nodes to allow the virtual machines with higher loads to use the resources thus released.

b) Migration of the virtual machines with the highest loads: When more desirable physical nodes are free, we could migrate the virtual machines with the highest loads to these physical nodes to achieve better performance. However, if the load is high because the virtual machine is running batch jobs, we should try to migrate to low-load virtual machines to allow the high-load virtual machines to consume more CPU cycles.

When a physical server goes down, the virtual machines that were running on that server will automatically migrate to other servers. Each application runs under the mutual supervision of two independent servers, which is a traditional fault-tolerance mechanism. When a host cannot continue operation, all of its services will immediately be taken over by another server. The system's key data are stored in a shared storage system to ensure real-time service. A highly available virtual machine can be built using existing hardware resources. This can greatly reduce the server interruption time caused by hardware malfunctions. Another disruption scenario occurs when a user accesses a virtual machine, causing its load to exceed a certain predetermined range and causing the virtual machine to exit from or move to a particular server. In settings that are limited by the overall availability of hardware resources, physical resources are shared throughout the cluster in accordance with the number of requests for CPUs to establish virtual resources for digital schools. The control algorithm is as follows:

$$\upsilon CPU_{eq} = \frac{\text{Total number of physical CPU cores in the cluster}}{\text{Total number of requested virtual machines, } \upsilon \text{CPU}}$$

$$\upsilon CPU_i > (1\text{-k}) \times \upsilon CPU_{eq} \quad (1)$$

$$\upsilon CPU_i < (1\text{-k}) \times \upsilon CPU_{eq} \quad (2)$$

If server i satisfies the condition expressed in equation (1), then it can join the server cluster. If it satisfies equation (2), it exits the server cluster in favor of the other services using this server. k is an accommodation coefficient. The value of k can be tuned to modify the exiting and adding conditions.

3.2. Mathematical Model

Load-balancing techniques can effectively reduce the response time and makespan of the system. The response time is defined as the time elapsed between the submission of a request and the first response produced. Reducing the wait time helps to improve the responsiveness of the VMs. The makespan is defined as the overall task completion time. We denote the completion time of task Ti on VMj by CTij:

$$\text{Makespan} = \max\{CT_{ij} | i \in T, i = 1, 2, \ldots n \text{ and } j \in VM, j = 1, 2, \ldots m\} \quad (3)$$

There are m virtual machines, VM = {VM1, VM2, . . . VMm}, that are processing n tasks, represented by the set T = {T1, T2, …, Tn}. We denote the completion time of task Ti by CTi. The processing time of task Ti on virtual machine VMj is denoted by Pij. The processing time of all tasks on VMj is defined by Eq. (4):

$$P_j = \sum_{i=1}^{n} P_{ij} \quad j = 1, \ldots, m \quad (4)$$

By minimizing CTmax, we obtain Eq. (5). From Eq. (4) and (5), we then obtain Eq. (6):

$$\sum_{i=1} P_{ij} \le CT_{max} \quad j = 1, \ldots, m \quad (5)$$

$$\Rightarrow P_j \le CT_{max} \quad j = 1, \ldots, m \quad (6)$$

At the time of load balancing, tasks will be transferred from one VM to another to reduce both CTmax and the response time.

$$CT_{max} = \left\{ \max_{i=1}^{n} CT_i , \max_{j=1}^{n} \sum_{i=1}^{n} P_{ij} \right\} \quad (7)$$

When transfers occur, the completion time of a task may change as a result of load balancing. The standard deviation of the load is

$$\sigma = \sqrt{\frac{1}{m} \sum_{i=1}^{m} (PT_i = PT)^2} \quad (8)$$

3.3. Load-balancing Algorithms

3.3.1. Active Clustering Algorithm: To address the load-balancing problem, changes must be made to the topology of the resource layer. We can use the cloud resource topology to solve the load-balancing problem. The active clustering algorithm is also known as self-aggregation. The basic concept is to iteratively gather similar services on each node of the network:

a) At a random point in time, a source node chooses one of its currently neighboring nodes as a matchmaker. The matchmaker node and the source node must be of different types.

b) Connections are initiated between the matchmaker's neighboring nodes and those of the source node. The matchmaker node initiates these connections.

c) The matchmaker establishes a mobile connection with the source node. The entire process consists of a complex set of organizational changes that eventually lead to the development of a stable network in which the similarity among the nodes can no longer be further modified. Current algorithms of this type maintain a certain number of connections while increasing the dissimilarity among nodes. However, the details of these complex algorithms are constantly changing.

This type of algorithm is a self-aggregation algorithm that rewires the network [37]. This procedure connects instances of similar service types together.

3.3.2. Random Algorithm: In this load-balancing approach, the load on the server is represented by the connectivity in a virtual graph. This approach provides a measure of

the initial availability. During balancing, the graph is altered in the following manner by the execution and completion of jobs [38]:

a) the in-degree of an executing node (available resources) decreases during job execution.

b) the out-degree of an allocating node (allocated jobs) and the in-degree of an executing node (available resources) increase after job execution, thus directing future load allocation.

Ultimately, we obtain a directed graph, and the directions of the edges direct the propagation for random sampling.

3.3.3. Honey Bee Foraging Algorithm: The honey bee foraging algorithm is designed to achieve the optimal allocation of resources [39]. The distribution server greatly enhances the capabilities and use of the system for digital school applications. This algorithm is applied in the cloud computing system to optimize each node to coordinate the operation of the BDS server.

1. Find capacity and loads of all VMs based on equations (4),(5),(6) and (7)
   Check if system is balanced or not:
   If σ <= Ts
      System is balanced
   Exit.
2. Load Balancing Decision:
   If L > maximum capacity
      Load Balancing is not possible
   Else
      Trigger Load Balancing.
3. Group VMs based on load as LVM,BVM and OVM
4. Load Balancing:
   Supply of Each machine in UVM is

$$Suppley\ of\ LVM_j = Maximum\ capacity - \frac{Load}{Capacity}$$

   Demand for Each machine in OVM is

$$Demand\ of\ OVM_j = \frac{Load}{Capacity} - Maximum\ Capacity$$

   Sort VMs in OVM by descending order.
   Sort VMs in LVM by ascending order.
   While LVM≠φ and OVM≠φ
      For s=1 to #(OVM) do
         Sort tasks in VMs by selection criterion(priority)
            For each task T in VMs find machine $VM_d \in$ LVM such as
            If(T is non preemptive)

$$T_h \rightarrow VM_d \mid \min(\sum T_h) \in VM_d \ and\ load_{VM_d} \leq Capacity_{VM_d}$$

$$T_m \rightarrow VM_d \mid \min(\sum T_h + \sum T_m) \in VM_d \ and\ load_{VM_d} \leq Capacity_{VM_d}$$

$$T_l \rightarrow VM_d \mid \min(\sum T) \in VM_d \ and\ load_{VM_d} \leq Capacity_{VM_d}$$

            If(T is preemptive)

$$T_h \rightarrow VM_d \mid \min(\sum T_h) \in VM_d$$

$$T_m \rightarrow VM_d \mid \min(\sum T_h + \sum T_m) \in VM_d$$

$$T_l \rightarrow VM_d \mid \min(\sum T) \in VM_d$$

               Update the number of tasks assigned to VM_d
               Update the number of priority tasks assigned to VM_d
               Update Load on both VMs, VM_d
      Update sets OVM,LVM,BVM
      Sort VMs in OVM by descending order
      Sort VMs in LVM by ascending order

We define three sets based on the load on the VMs: LVM (Low-loaded VMs) is the set that contains the low-loaded VMs. OVM (Overloaded VMs) is the set contains all overloaded VMs. BVM (Balanced VMs) is the set that contains all of the remaining VMs, which are balanced.

4. Architecture of the Adaptive Learning System: BDS

BDS supports all major computer operating systems and supports many types of intelligent mobile access terminals, including web platforms and mobile platforms. It provides on-demand access to teaching resources through a service platform for students, teachers and parents. The environment offers teacher curriculum videos, the availability of teachers online to answer questions, and psychological counseling to provide a more comprehensive range of services to guide students. The platform is intended to provide services not only for students but also for teachers and parents to support them in building digital school groups, a space for communication among teachers and among parents, and other channels and platforms for the exchange of learning. It provides a unified personal space, knowledge management, resource centers and learning centers for teachers and students. The platform supports teacher-student interactions, the creation of classes and associations, and resource sharing and provides integrated services such as teaching tools. It establishes a secure, convenient and easy-to-use digital school system. The system offers a variety of human-centric, intelligent applications and services to meet the needs of all parties.

4.1. Design of the Overall Architecture

BDS includes a web digital school platform and a mobile digital school platform. The web digital school platform has a B/S architecture and is a cross-platform service system. The mobile digital school platform supports the iOS and Android operating systems that are currently most popular on the market. The overall framework, as illustrated in Figure 1, is divided into five layers: the interface layer, the application layer, the support layer, the data layer, and the base layer. An adaptive learning model is implemented in the support layer to offer different types of learning content and learning activities to different students. It contains three modules: the learner model, the domain model and the adaptive engine.

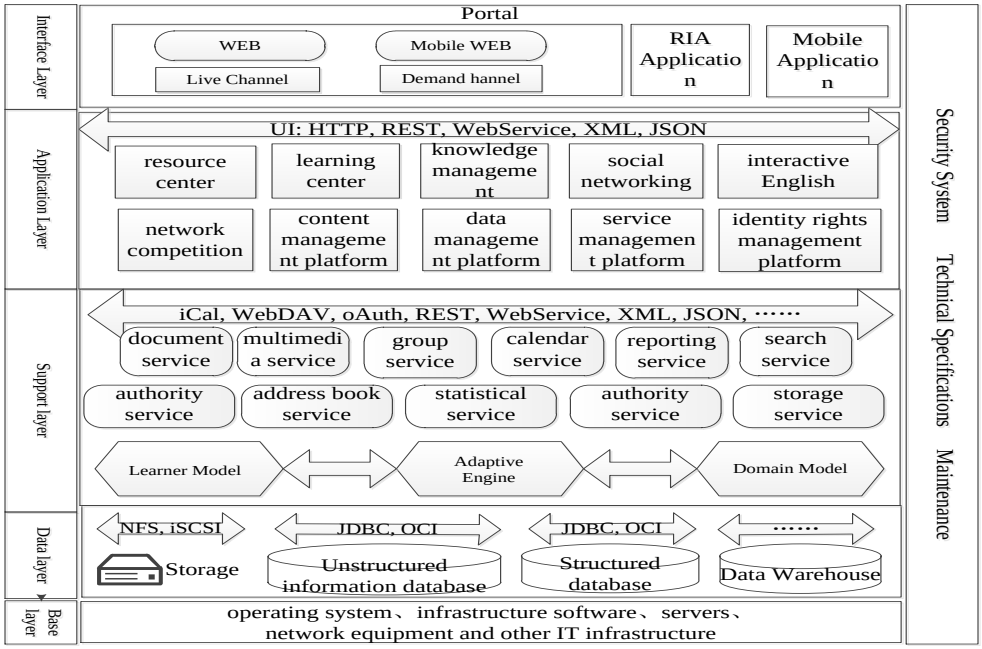

Figure 1. Overall Architecture Diagram

Interface Layer: This layer provides a user-oriented front-end interface. Through pages and components, the interface layer provides the user interface that is used to access the input and the operating system.

Application Layer: This layer is the critical part of the system that serves to support the core business operations; it includes the "resource centers", "learning centers", "knowledge management", "social networking", "interactive English", "network communication", "content management platform", "data management platform", "service management platform", and "identity rights management platform" for BDS.

Support layer: This layer provides the basic support for the functions and services of the functional modules on the application layer and the components for the main functions of the application platform. It includes an adaptive learning model and a service model. The service model comprises "storage services", "document services", "multimedia services", "search services", "authentication services", "authority services", "group services", "address book services", "calendar services", "statistical services", and "reporting services". The adaptive learning model comprises the “learner model”, the “domain model” and the “adaptive engine”. The learner model maintains a representation of information about each individual learner. The learner model stores information about each learner’s learning style and cognitive ability, and it is important in both representing this knowledge and providing the necessary infrastructure for adaptivity. The domain model is used for the storage, organization and description of the learning content. This model supports various types of educational content, including not only standard narrative content but also pedagogical activities [40]. The adaptation engine is implemented as a set of rules that are obtained from the connections between the domain model and the learner model to update the learner model and provide appropriate pedagogical activities.

Data layer: This layer provides the main storage of documents, images, and 10,050 teaching curriculum videos. It also includes an external service interface.

Base layer: This layer provides the underlying support for this system: the operating system, software infrastructure, servers, network equipment and other IT infrastructure. This layer is also responsible for the overall management and allocation of computing resources.

4.2. Web platform Technology Architecture

BDS is most often accessed through the web platform, so the architectural design of the web platform technology in practice must both meet the various study needs that the system is intended to address and also solve the problems of a single point of failure, load imbalance, centralized access, waste of idle equipment resources, among others. To address these concerns, a cloud computing platform was introduced into the framework. The cloud computing platform improves the availability of the system. The web platform is also divided into five levels, as shown in Figure 2: the interface layer, the application layer, the support layer, the data layer and the base layer.

The contents of the interface layer, the application layer and the base layer are consistent with the overall architecture of the web platform. The external interfaces and the adaptive model are realized in the support layer. For example, the Beijing No. 4 High School teaching resources and the educational resources for children provided by the Ministry of Foreign Affairs of the People’s Republic of China are located in the support layer. The adaptive model offers different types of learning content and learning activities for different students and functions in the same manner as in the overall architecture. For most access traffic on the web platform, the user accesses are concentrated on the data layer. The platform is an ultra-large-scale deployment structure. The support layer manages multiple sets of on-demand servers for remote videos and media resources at the same time. The platform supports the unlimited expansion of CDN video servers, where a single server can support more than 3,000 viewers. This will enable the system to

successfully and simultaneously provide quality instruction to students through video on-demand and live courses.

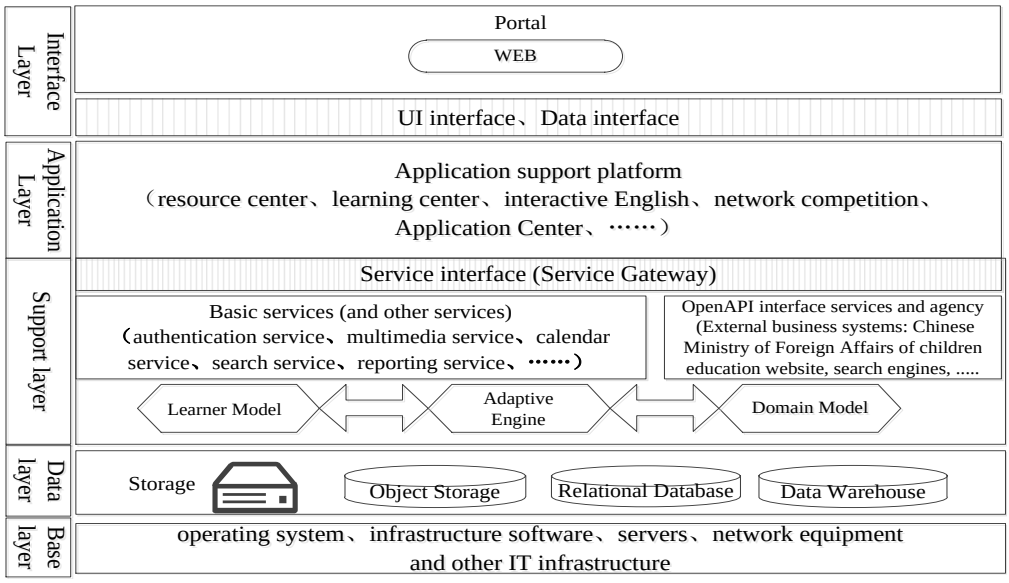

Figure 2. Web Platform Architecture Diagram

4.3. Mobile Platform Technology Architecture

The mobile digital school platform supports the iOS and Android operating systems that are currently most popular on the market. It can be accessed through mobile phones and tablet PCs, and the cloud computing platform is included within its framework. The mobile BDS learning platform, as shown in Figure 3, is divided into five levels: the interface layer, the application layer, the support layer, the data layer and the base layer.

The application layer encompasses the synchronous course multi-platform mobile learning system: the "mobile resource integration services," the "mobile learning community", the "3G education information center", the "personal center", the "statistics and analysis center" and other applications.

The support layer provides the main functions of the underlying platform for mobile applications, which are integrated by means of mobile app development tools. The function of the adaptive model is the same as that in the overall architecture, and the key features of this layer include "user management", "terminal management", "application management", "push services", "log management", and "statistical management".

The data layer can be divided into two categories, comprising the two types of data entities stored in the present system. The first category is structured data, which are stored in a content-based system of teacher resources, student information, educational resources, knowledge center resources and other data. The second category is unstructured data, which are mainly used to store documents, images, and 10,050 teaching course videos.

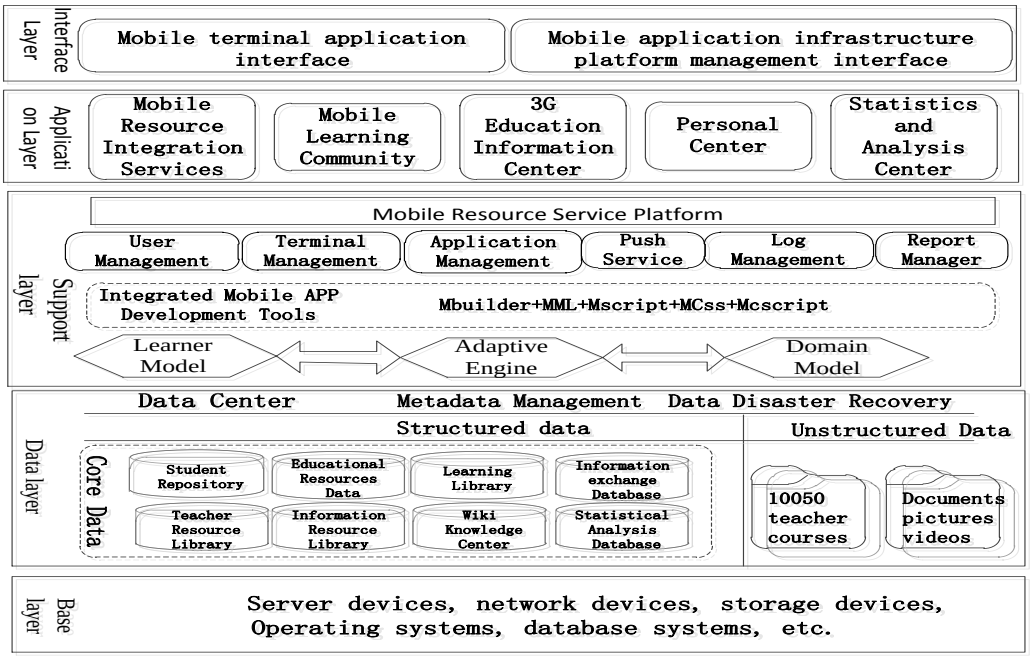

Figure 3. Mobile Platform Architecture Diagram

The five layers of the mobile platform have the following features: an integrated system architecture based on SOA(Service-Oriented Architecture) technology to achieve system interoperability, process reengineering and data sharing; mobile platform integration through component reuse of modules related to the workflow engine, user management, authentication and authorization, data integration, form statistics, data interfaces and retrieval, and other components of the component library; and the establishment of mobile applications based on the integrated development tools for mobile apps provided by the "Mcube mobile development platform" to implement certain specific functions. Mcube includes MClient (client), MServer (server), Mbuilder (integrated development environment, IDE), Mscript and Mcscript.

5. Implementation and Results

5.1. Access Analysis

Since the launch of BDS, the average number of visits per month has been consistently greater than 3 million. Figure 4 shows the amounts of data of different types that are accessed per month to facilitate an analysis of students' online behavior. We can see that BDS is most commonly accessed to view teaching courses. Students view these online teaching courses as supplementary instruction. The next greatest concentration of access traffic is concerned with the digital notifications provided by the schools to reduce the volume of paper documents issued.

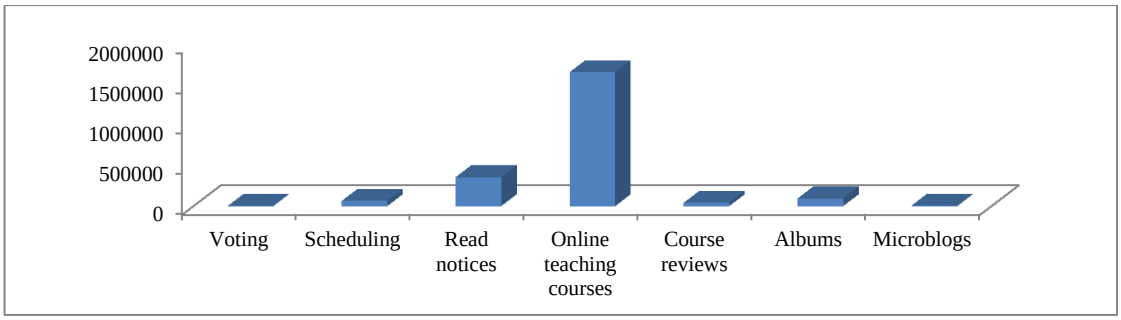

Figure 5. Statistics Regarding Students' Online Behavior

Figure 5 shows the distribution of student access to teaching courses in terms of academic discipline. The average duration of access to each lesson is approximately 15.5 minutes. The most frequently accessed courses are in mathematics, Chinese and English. This indicates that students are most concerned with the school curriculum and achieving a consistent learning process.

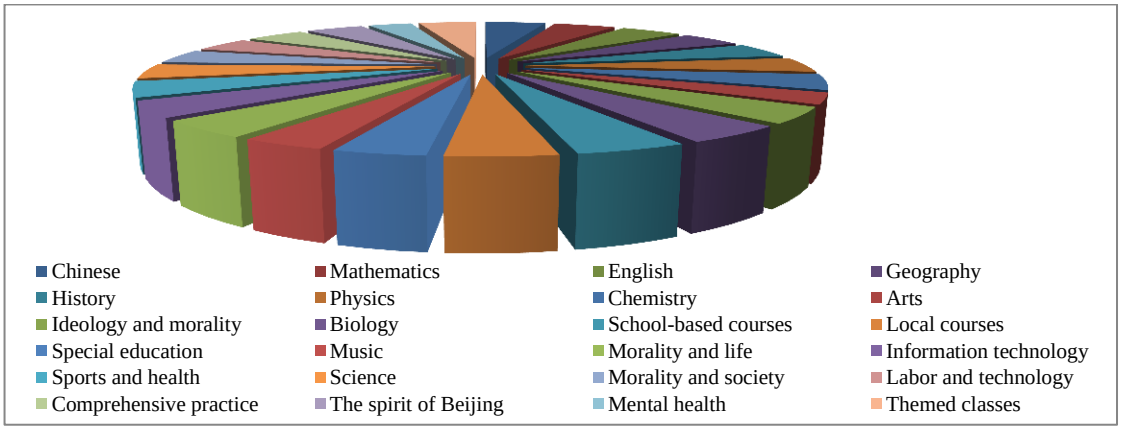

Figure 6. Distribution of Resource Access by Discipline

Figure 6 shows the BDS access distribution of primary and secondary school students in various districts and counties of Beijing. Because of the uneven distribution of educational resources, high-quality resources are concentrated in urban areas, whereas suburban school districts perform relatively badly. In the observed distribution, Haidian district is found to generate the most BDS visits among all districts in terms of the number of pupils.

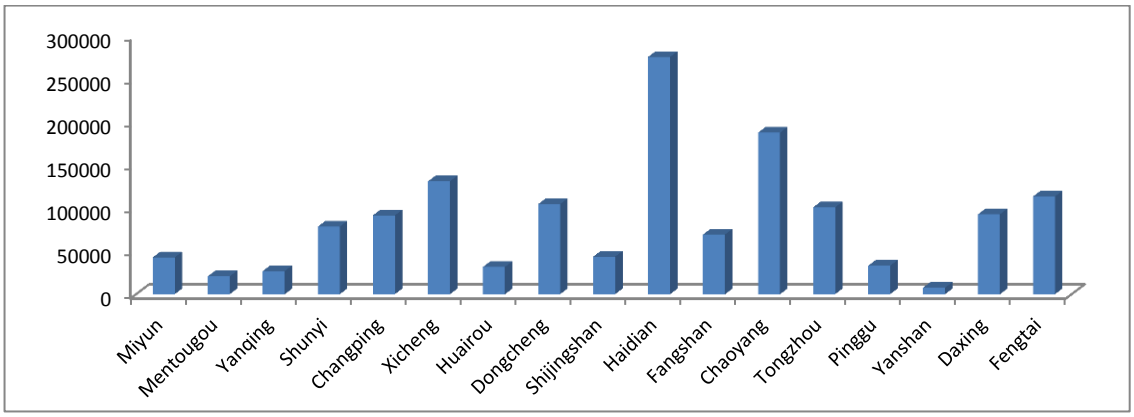

Figure 7. Access Distribution by Region

Figure 7 shows the access distribution with respect to the level of schooling. The largest number of visitors is in junior high. Primary school students have more spare time, but their abilities and their level of demand are not as strong; thus, few of them access the BDS system. Meanwhile, although the individual learning capabilities of high school students are better, they are under greater academic pressure and have less spare time and thus access the system similarly rarely. By contrast, junior high students have a good ability for self-selected learning and are subject to less academic pressure; thus, they are most likely to seek self-selective schooling through the digital platform.

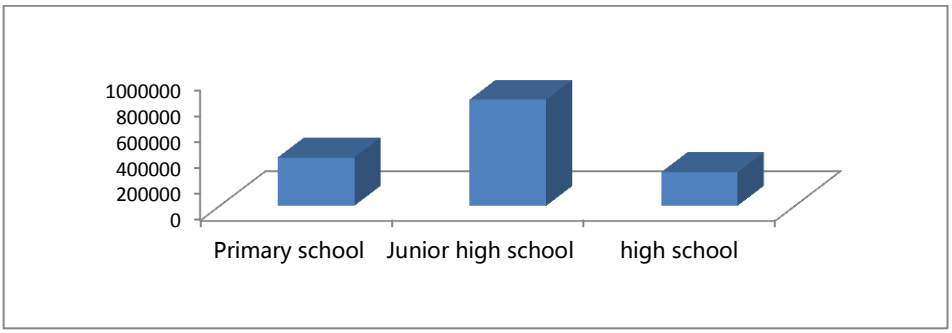

Figure 8. Access Distribution by Level of Schooling

5.2. System Performance Results

When a large number of users access the system at the same time, especially when accessing video resources, the core processing capacity of the system must increase. User access is the focus of video playback, including both live and on-demand video. Depending on the actual needs of a video task, it can be assigned to one of two types of video system: the physical on-demand video system or a virtual on-demand video system. An on-demand video system is deployed in the physical blade servers to ensure performance, whereas additional virtual on-demand video systems are also deployed in a resource pool through the virtualization software VMware. When the physical hardware resource utilization of an on-demand video system is greater than 90%, the high access pressure will lead to video delays and pauses. Using the applied strategy, the level of virtual system resource utilization for on-demand video can be maintained at approximately 60%, and even if the access pressure increases, EucMyptus can dynamically adjust the equipment resources as necessary.

Table 1. Makespan in Seconds using the Honey Bee Foraging Algorithm

No. of tasks	*Before load balancing*	*After load balancing*
100	*11.2*	*5.4*
200	*27.1*	*14.8*
300	*33.5*	*19*
400	*54.5*	*26*

Table 1 reports the makespans for various numbers of tasks before load balancing and after load balancing using the honey bee foraging algorithm. From Figure 8, it is evident that when the number of user connections established to view a teaching curriculum video reaches 2000, the degree of load balancing achieved using the honey bee foraging algorithm reaches a plateau, whereas the other algorithms do not stabilize until the number of concurrent connections reaches 3500. Moreover, for any arbitrary number of concurrent connections, the load balancing degree of the honey bee foraging algorithm is lower than those of the other algorithms, indicating superior results.

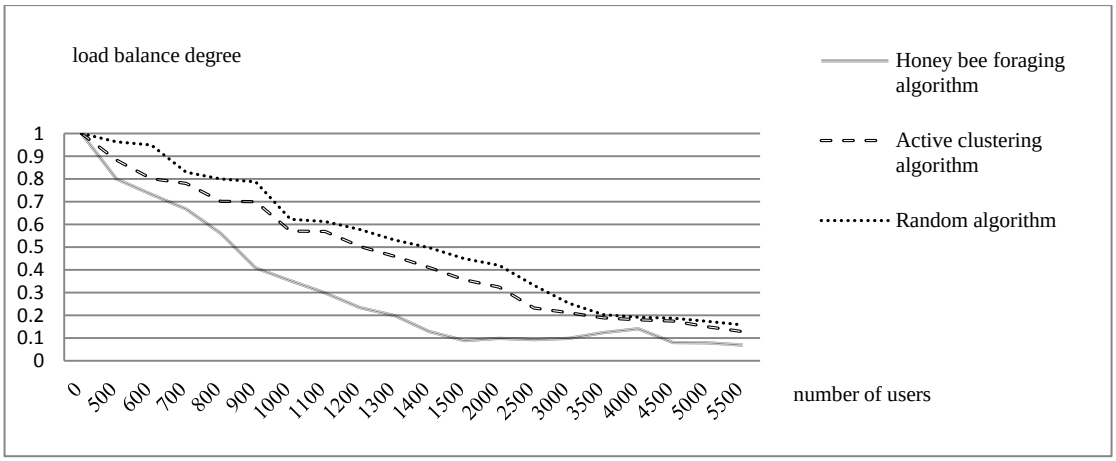

Figure 9. Comparison of Various Algorithms in Terms of the Load Balancing Degree

5.3. Learning performance Evaluation

An experiment was designed to evaluate the learning performance of different students using BDS. We chose "Information Technology" as the experimental course and designed a post-test experiment to determine the effects of the proposed e-learning system. One hundred and fifty students were selected to take part in this experiment. All were senior high school students in grade one from five schools (suburban and urban). They were randomly allocated into two groups: an experimental group that worked with the proposed version of BDS, which includes an adaptive learning model, and a control group that worked with a traditional e-learning system, which uses a non-adaptive learning model. Table 2 shows the distribution of the students.

Control group: Traditional e-learning using the old version of BDS without the adaptive learning model: the students browsed the materials following their own plans and selections.

Experimental group: Adaptive e-learning using the proposed version of BDS, in which experts have defined different learning paths and content in accordance with individual needs: the students studied the content by navigating the adaptive learning paths determined by the adaptive learning model.

Table 2. Demographic Data and Percentage Distribution

	Control group		***Experimental group***		***Total***	
Sex	*Frequency*	*%*	*Frequency*	*%*	*Frequency*	*%*
Male	*80*	*53.3*	*85*	*56.7*	*165*	*55*
Female	*70*	*46.7*	*65*	*43.3*	*135*	*45*
Total	*150*	*100*	*150*	*100*	*300*	*100*

The experiment was performed in two phases. In the first phase, the students in the experimental group were provided with information about the learning style categories and were asked to complete a learning style questionnaire. No tasks were required of the control group. In the second phase, all of the students regularly followed the presented lessons through the completion of the course and then took the post-test, which was used to determine the differences, if any, in their learning performance. Our research hypothesis was as follows: because the participating students were young and possessed only some basic computer skills, we assumed that they all possessed the same prior knowledge regarding the "Information Technology" topic. An independent sample t-test was performed to compare the mean scores of the two groups from the post-test. Table 3

reveals that the mean post-test scores of the students who used the proposed version of BDS were higher than those of the students who used the traditional e-learning system without the adaptive learning model.

Table 3 Comparison between the Control and Experimental Groups in Terms of Post-test Performance

Subject	Control group		Experimental group		Independent t-test	
	Mean	SD	Mean	SD	t-test value	P-value
Information technology	76.44	10.11	83.13	7.97	-2.310	0.0262
concepts	77.23	8.14	82.78	7.39	-2.238	0.0308
Information processing						
Database operation	79.98	9.97	85.72	7.47	-2.099	0.0421

5.4. Summary

Through the access analysis, system performance and learning performance evaluation proved that BDS provided an efficient and individual learning environment to learners. We introduced cloud computing technology into the adaptive system to solve not only the problem of personalized learning but also the problem of providing access to a large number of students. The access analysis is composed of statistics students' online behavior, access discipline resource, access level of schooling and region. First, according to obtained data, we conclude that most junior high students from urban areas commonly accessed to view teaching courses in mathematics, Chinese and English. In order to solve the different needs of a large number of students, we used the honey bee foraging algorithm to improve the BDS performance. The results of the load balancing degree show that of the honey bee foraging algorithm is better than active clustering algorithm and random algorithm. At last, on the basis of effectively solve a large number of students simultaneous access problem, we designed an experiment. The result of the evaluation clearly shows that, with the use of BDS, students' achieved results are better. Results of the study showed that BDS system was considered rather beneficial by all the primary and secondary students. Through BDS they had a positive impact on their learning and eased their comprehension. Taking this architecture into consideration, it is possible to say that BDS is a totally students centered system, and it offers choices to students in each step according to their cognitive performances.

6. Conclusion

This paper discusses the implementation and evaluation of BDS. Since the launch of the system services, 1.45 million students in Beijing have accessed the system and 88,000 Beijing teachers have exchanged their experiences. More than 1600 teaching and research staff in the various municipalities and districts have participated in discussion activities. Approximately 6400 students and nearly 500 teachers in the provinces of Guizhou, Yunnan and Hunan have created accounts and engaged in learning and exchange through BDS. It is now gradually being opened up to parent accounts. Using this system, individual students receive different educational contents and activities in accordance with their own particular learning characteristics. Cloud computing technology effectively solves the problems of the availability of computing resources and user access balance. By virtue of the adaptive learning module, the system can meet the different learning needs of different students. It effectively solves the problem of the sharing of high-quality teaching resources, and it serves as an exemplar and reference for other departments of education.

Acknowledgements

This research was supported by the Beijing Municipal Commission of Education Projects "Research on Beijing Basic Education Data Services Construction and Application" (No. AJA11171) and "Research and Experimentation on Education in the Large-scale Application Mode" (PXM2014_014257_000032) as well as "The Beijing Digital School Project".

References

[1] Esichaikul, V., Supaporn, L., & Clemens, B. Student modeling in adaptive e-learning systems. Knowledge management and E-Learning: An International Journal, 3(3), **(2011)**, 342–355.
[2] Brusilovsky, P. Adaptive hypermedia. User Modeling and User-Adapted Interaction, 11, **(2001)**, 87–110.
[3] Berge, Z. L. Active, interactive, and reflective e-Learning. Quarterly Review of Distance Education, 3(2), **(2002)**, 181–190.
[4] Picciano, A. G. Distance learning: Making connections across virtual space and time . Upper Saddle River, NJ: Prentice-Hall **(2001)**.
[5] Saba, F. Student attritions: How to keep your online learner focused. Distance Education Report, 14(4), **(2002)**, 1–2.
[6] Retalis, R., & Papasalouros, A. Designing and generating educational adaptive hypermedia applications. Educational Technology & Society, 8 (3), **(2005)**, 26–35.
[7] Brusilovsky, P., & Peylo, C. Adaptive and intelligent web based educational systems. International Journal of Artificial Intelligence in Education, 13, **(2003)**, 156–169.
[8] Romero, C., Ventura, S., Zafra, A., & de Bra, P. Applying web usage mining for personalizing hyperlinks in web-based adaptive educational systems. Computers & Education, 53(2009), **(2009)**, 828–840.
[9] Dustin oWens, B. Y. Securing elasticity in the cloud. Communications of the ACM, 53(6), **(2010)**, 46–51.
[10] Vaquero, L. M., Rodero-Merino, L., Caceres, J., & Lindner, M. A break in the clouds: towards a cloud definition. ACM SIGCOMM Computer Communication Review, 39(1), **(2008)**, 50-55.
[11] Brusilovsky, P. (1998). Methods and techniques of adaptive hypermedia. In Adaptive hypertext and hypermedia (pp. 1-43). Springer Netherlands.
[12] Carver, C. A., Howard, R. A., & Lane, W. D. Enhancing student learning through hypermedia courseware and incorporation of student learning styles. IEEE Transactions on Education, 42 (1), **(1999)**, 33–38.
[13] Gilbert, J. E., & Han, C. Y. Adapting instruction in search of ''A Significant Difference''. Journal of Network and Computing Applications, 22 (3), **(1999)**, 149–160.
[14] Moore, A., Stewart, C. D., Zakaria, M. R., & Brailsford, T. J. WHURLE – an adaptive remote learning framework. In International conference on engineering education (ICEE-2003), **(2003)** July 22–25; Valencia, Spain.
[15] Wolf, C. iWeaver: Towards 'learning style' based e-learning in computer science education. In Proceedings of the fifth Australasian conference on computing education, **(2003)** February 4–7, 273–279; Adelaide, Australia.
[16] Brown, E. The use of learning styles in adaptive hypermedia. Unpublished Doctoral Thesis. England: The University of Nottingham **(2007)**.
[17] Brown, E., Brailsford, T., Fisher, T., & Moore, A. Evaluating learning style personalization in adaptive systems: Quantitative methods and approaches. IEEE Transactions on Learning Technologies, 2 (1), **(2009)**, 10–22.
[18] Mustafa, Y. E. A., & Sharif, S. M. An approach to adaptive e-learning hypermedia system based on learning styles (AEHS-LS): Implementation and evaluation. International Journal of Library and Information Science, 3(1), **(2011)**, 15–28.
[19] Brown, E., Fisher, T., & Brailsford, T. Real users, real results: Examining the limitations of learning styles within AEH. In Proc. of the eighteenth ACM conference on hypertext and hypermedia **(2007)**.
[20] Dolenc, K., & Aberšek, B. TECH8 intelligent and adaptive e-learning system: Integration into Technology and Science classrooms in lower secondary schools. Computers & Education, 82, **(2015)**, 354-365.
[21] Baki, A., & Çakıroglu, Ü. Learning objects in high school mathematics classrooms: Implementation and evaluation. Computers & Education, 55,**(2010)**, 1459–1469.
[22] Özyurt, Ö., Özyurt, H., Baki, A., & Güven, B. Integration into mathematics classrooms of an adaptive and intelligent individualized e-learning environment: Implementation and evaluation of UZWEBMAT. Computers in Human Behavior, 29(3), **(2013)**, 726-738.
[23] Karaboga, D. An idea based on honey bee swarm for numerical optimization (Vol. 200). Technical report-tr06, Erciyes university, engineering faculty, computer engineering department **(2005)**.

[24] Cuevas, E., ZaldíVar, D., PéRez-Cisneros, M., Sossa, H., & Osuna, V. Block matching algorithm for motion estimation based on Artificial Bee Colony (ABC). Applied Soft Computing, 13(6), **(2013)** 3047-3059.
[25] Pan, Q. K., Tasgetiren, M. F., Suganthan, P. N., & Chua, T. J. A discrete artificial bee colony algorithm for the lot-streaming flow shop scheduling problem. Information sciences, 181(12), **(2011)**, 2455-2468.
[26] Singh, A. An artificial bee colony algorithm for the leaf-constrained minimum spanning tree problem. Applied Soft Computing, 9(2), **(2009)**, 625-631.
[27] Kang, F., Li, J., & Xu, Q. Structural inverse analysis by hybrid simplex artificial bee colony algorithms. Computers & Structures, 87(13), **(2009)**, 861-870.
[28] Karaboga, N., & Cetinkaya, M. B. A novel and efficient algorithm for adaptive filtering: artificial bee colony algorithm. Turkish Journal of Electrical Engineering & Computer Sciences, 19(1), **(2011)**, 175-190.
[29] Tsai, P. W., Khan, M. K., Pan, J. S., & Liao, B. Y. Interactive artificial bee colony supported passive continuous authentication system. Systems Journal, IEEE, 8(2), **(2014)**, 395-405.
[30] TSai, P. W., Pan, J. S., Liao, B. Y., & Chu, S. C. Enhanced artificial bee colony optimization. International Journal of Innovative Computing, Information and Control, 5(12), **(2009)**, 5081-5092.
[31] Rao, R. S., Narasimham, S. V. L., & Ramalingaraju, M. Optimization of distribution network configuration for loss reduction using artificial bee colony algorithm. International Journal of Electrical Power and Energy Systems Engineering, 1(2), **(2008)**, 116-122.
[32] Randles, M., Lamb, D., & Taleb-Bendiab, A. A comparative study into distributed load balancing algorithms for cloud computing. InAdvanced Information Networking and Applications Workshops (WAINA), 2010 IEEE 24th International Conference on (pp. 551-556). IEEE **(2010)**.
[33] Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R., Konwinski, A., ... & Zaharia, M. A view of cloud computing. Communications of the ACM, 53(4), **(2010)**, 50-58.
[34] Vaquero, L. M., Rodero-Merino, L., Caceres, J., & Lindner, M. A break in the clouds: towards a cloud definition. ACM SIGCOMM Computer Communication Review, 39(1), **(2008)**, 50-55.
[35] Youseff, L., Butrico, M., & Da Silva, D. Toward a unified ontology of cloud computing. In Grid Computing Environments Workshop, 2008. GCE'08 (pp. 1-10). IEEE **(2008)**.
[36] Ortigosa, A., Martín, J. M., & Carro, R. M. Sentiment analysis in Facebook and its application to e-learning. Computers in Human Behavior,31, **(2014)**, 527-541.
[37] Saffre, F., Tateson, R., Halloy, J., Shackleton, M., & Deneubourg, J. L. Aggregation dynamics in overlay networks and their implications for self-organized distributed applications. The Computer Journal, 52(4), **(2009)**, 397-412.
[38] Rahmeh, O. A., Johnson, P., & Taleb-Bendiab, A. A dynamic biased random sampling scheme for scalable and reliable grid networks. INFOCOMP Journal of Computer Science, 7(4), **(2008)**, 1-10.
[39] Nakrani, S., & Tovey, C. On honey bees and dynamic server allocation in internet hosting centers. Adaptive Behavior, 12(3-4), **(2004)**, 223-240.
[40] Mahnane, L., Laskri, M. T., & Trigano, P. An adaptive hypermedia system integrating thinking style (AHS-TS): Model and experiments. International Journal of Hybrid Information Technology, 5, **(2012)**, 12–28.

Authors

Yi Liao, He is a doctor at economics and management college, Beijing Jiaotong University. His research direction is the information theory and application of education.