Future Generation Computer Systems 38 (2014) 1–10

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

A scalable blackbox-oriented e-learning system based on desktop grid over private cloud

Lung-Pin Chen [a,*], Jien-An Lin [a], Kuan-Ching Li [b], Ching-Hsien Hsu [c], Zhi-Xian Chen [a]

[a] Department of Computer Science, TungHai University, Taiwan, ROC
[b] Department of Computer Science and Information Engineering, Providence University, Taiwan, ROC
[c] Department of Computer Science and Information Engineering, Chung Hua University, Taiwan, ROC

HIGHLIGHTS

- Enabling e-learning users to contribute to the server computation via desktop grid.
- Developing agent-based worker deployment for the workers hosted in VMs.
- Developing new "workunit-based scheduling algorithm" for assigning tasks to workers of different capability.

ARTICLE INFO

Article history:
Received 7 December 2011
Received in revised form
10 February 2014
Accepted 24 February 2014
Available online 20 March 2014

Keywords:
Cloud computing
Desktop grid
E-learning
Volunteer computing

ABSTRACT

Traditional web-based e-learning system suffers from unstable workloads and security risks of incorporating external executable objects to servers. This paper addresses these issues with emerging technologies, as desktop grid and cloud computing. Learning users are motivated to be volunteers by hosting the virtual machines equipped with e-learning desktop grid applications. We develop components to integrate the e-learning system and desktop grid into the circumstance in which each user serves not only a task producer, but also a volunteer that solves tasks. In order to enhance the responsiveness between the passive desktop grid server and e-learning system, we have also developed asynchronous processes to enable the server and volunteer workers to cooperate in a tightly coupled manner. The system achieves the scalability by maintaining the ratio between the number of volunteers and the number of online users beyond certain threshold.

1. Introduction

E-learning is a rapid growing and important application in the Internet and is widely used in many enterprises and educational communities [1]. With the continuous evolution of the Internet, there is an increasing demand for incorporating online practice or online certification into traditional e-learning systems. This work focuses on a web-based computer language e-learning system that allows users to upload their programs to the server where the code is parsed, performed and evaluated.

The client–server computing model provides the economical and robust medium for adaptive learning management. However, system non-responsiveness is often experienced since the client request rate may exceed the fixed server capacity. A traditional solution is to build a dedicated server cluster with a task dispatcher. Since users can submit tasks intermittently in an e-learning system, a server may stay idle during non-training time.

Today, many research works have been devoted to various e-learning systems. Instead of developing new functionalities, this work attempts to explore the feasibility of building a low-cost and scalable e-learning system via user cooperation. The idea is to gather computing resources from e-learning users by using the recently emerged volunteer computing and cloud computing techniques [2–6].

Volunteer computing is a type of distributed computing that allows people donating their computing resources to certain projects [7,8]. BOINC is a volunteer computing platform that has demonstrated its efficacy for large scientific research projects in fields as biology, medicine, astrophysics, engineering and climate study [9,10]. By breaking a monolithic job into a large number

* Correspondence to: Department of Computer Science, TungHai University, No. 1727, Sec. 4, Taiwan Boulevard, Xitun District, Taichung 40704, Taiwan, ROC. Tel.: +886 4 23590415; fax: +886 4 23591567.
E-mail address: lbchen2007@gmail.com (L.-P. Chen).

http://dx.doi.org/10.1016/j.future.2014.02.017
0167-739X/

of units and distributing into different desktop computers called *workers*, BOINC can perform jobs efficiently via massive parallelism. Cloud computing [2] is a computing paradigm that describes a model of supplement, consumption, and delegation of computing resources to the users on-demand. Cloud computing platform is usually viewed as a management system of a pool of virtualized resources [2]. Virtualization brings many benefits for manipulating computing resources across complex and heterogeneous environments.

In this work, the *e-learning desktop grid* refers to a VM infrastructure built on top of the user's computers, in which each user is not only a generator of tasks, but also a volunteer to solving tasks. The following main issues are addressed.

Promotion. Motivating has been always critical for volunteer computing. A naive *restrict approach* is to prohibit users from submitting new tasks unless they contribute their resources proportionally. In this paper, we adopt the *non-restrict approach* which motivates users via user credits in the e-learning web site.

Performance. In the e-learning private cloud, user-perceived response time is a major concern since the workers are voluntary and can disconnect spontaneously. The servers' responsiveness directly depends on the ratio between the number of online learners and the number of online volunteers, which can be controlled via the restrict or non-restrict promotion policies mentioned above. We introduce some high capability official workers and develop a new scheduling algorithm to integrate the official and volunteer workers efficiently.

Security. Unlike normal volunteer computing platform where tasks are normally generated in authenticated servers, the tasks in the e-learning system are initiated by those in-trained learners. Executing the e-learning tasks may lead to unexpected results in the volunteer's computer. In addition, to prevent plagiarism, volunteers must be prohibited from inspecting the code delegated to their computers.

To the best of our knowledge, the notion of gathering computing resources from e-learning users to maintain system responsiveness is original. There are several existing web-based e-learning systems that provide automated and distributed online judge functionalities [11–13]. These systems employ more servers as the system scale increases. In this paper, the e-learning desktop grid achieves scalability by involving user computers. Most cloud computing platforms assume that VM hosts are directly controlled by a front-end server, which is not applicable in volunteer computing environment. In [14,15], the authors developed the tools for automated deployment of VMs to the computers owned by anonymous volunteers. Compared to their works, our deployment component is more complex since the system has to maintain a membership mapping between the e-learning system and the volunteer computing platform.

The proposed architecture allows participants not only to donate their resources but also to use others' as well. When employing a *broker*, the participants form a desktop grid community that helps members to solve overloaded large-scale tasks such as computational science projects or artificial intelligence game tree searches. Developing a resource broker that can fairly and efficiently assign resource for the desktop grid community will be a future work.

The remaining of this paper is organized as follows. Section 2 introduces the backgrounds of e-learning systems. Section 3 discusses the system architecture and implementation issues. Section 3.4 discusses the scheduler for assigning tasks among official and volunteer workers. Finally, conclusion remarks are made in Section 5.

2. Blackbox-oriented e-learning

This work models an e-learning system as a *blackbox-oriented* system—a platform representing the learning contents as a composition of *blackboxes*, which are software components whose internals cannot be viewed. The term "blackbox" indicates that the system only considers the input/output of learner behaviors and ignores completely other details. This model has been significantly adopted in many organizations and is referred by different terminologies, such as outcome-based education or performance-based education, *etc.* [16]. In this paper, we adopt the name "blackbox" in terms of software technologies.

An invocation of blackbox b with input data I generates the output referred to by invoke(b, I). We assume that blackboxes are *deterministic*. That is, a blackbox always produces the same result for the same input data. Therefore, the behaviors of a blackbox cannot be dependent on time-related functions or random-number functions.

In the perspective of computer language education, interactive code review can conduct learning context more effectively. This approach is referred to as the *whitebox-oriented e-learning*. However, it is also considered difficult and costly to be implemented. In fact, the whitebox-oriented e-learning can be emulated by using blackboxes with adapted granularity (fine grain vs. coarse grain). In the extreme case, given a sequence of program instructions, the system can create one blackbox for each single instruction to force users to examine the program line-by-line. On the other hand, constructing blackboxes with coarse grain leads to a result-oriented learning approach.

Fig. 1 illustrates the architecture of a blackbox-oriented LMS (Learning Management System). Each learning object in the DAG is associated with an *instructor blackbox* (or *T-box*) as well as a set of input data. To complete the lesson, a user has to construct a *user blackbox* (or *U-box*) and try to make it act like the T-box. A *checker* is a program to determine whether two blackboxes act like each other for the same input data. In this work, we use an elementary checker which simply uses the string comparison functions to judge the equality of outputs of two blackboxes. The checker can be customized to evaluate blackboxes based on exact textual matches or approximate pattern matches, depending on individual e-learning application requirements.

When navigating the learning contents, a user is not allowed to advance to object w in the DAG unless all the prerequisite objects of w (*i.e.,* all the objects with edge to w) are completed. Theoretically, if the cardinality of input data is large enough and the U-box always generates expected answers that can be approved by the checker, then user k has a high probability of successfully completing the topic w. For each user, the *learning progress* is the collection of the objects completed by the user.

This paper implements the LMS shown in Fig. 1 as a web-based e-learning system [17]. In this system, each T-box represents the instructional materials of a topic of computer language (C++ or Java), including the textual and illustrative explanations, the source code of the standard answer, and a set of input data for learning evaluation. The teacher can upload the instructional materials and maintain their precedence relationships up to the LMS via the web-based administrative interface. To complete a course, a user has to upload a U-box which contains the source code of the answer for the topic. The server then compiles, executes, and invokes the checker to validate the answer. The user learning progress is updated if the quality of the U-box is approved. In this research paper, the implementation of the e-learning web site is beyond the scope of this article and will be omitted.

3. System design and development

3.1. System architecture

This subsection gives an overview and depicts the specific requirements for the e-learning desktop grid. The system contains an e-learning web server, a BOINC server and the worker control