mathematics

MDPI

Article

Optimal Resource Allocation Model and Algorithm for Elastic Enterprise Applications Migration to the Cloud

Shiyong Li †, Yue Zhang † and Wei Sun *,†

School of Economics and Management, Yanshan University, Qinhuangdao 066004, China; shiyongli@ysu.edu.cn (S.L.); 18712797729@163.com (Y.Z.)

* Correspondence: wsun@ysu.edu.cn; Tel.: +86-335-8057025

† These authors contributed equally to this work.

Received: 31 July 2019; Accepted: 26 September 2019; Published: 1 October 2019

Abstract: Cloud computing has been widely used in various industries in recent years. However, when migrating enterprise applications into the cloud, enterprise users face a problem with minimizing migration time and cloud resource providers face a dilemma of resource allocation problem, with the objective of maximizing the migration utility of enterprise users while minimizing the cost of cloud resource providers. In order to achieve them, this paper considered cloud migration objectives including cloud migration time, cloud migration utility, and cloud data center cost, and proposed a resource allocation model for enterprise applications migration into the cloud. The model is divided into two stages: the bandwidth allocation for enterprise applications migration to the cloud and the physical resource allocation of cloud resource providers for enterprise applications deployment into the cloud. In the first stage, we aim to minimize the cloud migration time for enterprise applications, and propose a scheme of bandwidth allocation for each component of applications. In the second stage, we present the resource allocation of cloud resource providers and propose a gradient-based algorithm which can achieve optimal resource allocation. Finally, we give some numerical simulation results to illustrate the performance of the proposed algorithm.

Keywords: cloud migration; enterprise applications; resource allocation; utility functions; gradient-based algorithm

1. Introduction

Traditional enterprise business management is to select special servers to respond to customer services, or establishing their own computer rooms, or renting independent servers. In recent years, with the rapid growth of enterprise business data, cloud computing technology emerged. Following Li and Yuan (2012) [1], cloud computing integrates IT resources into a large scalable resource pool with virtualization technology, and provides services in the form of software as a Service (SaaS), platform as a Service (PaaS), and infrastructure as a Service (IaaS). Cloud computing has attracted the attention and application of enterprises because of many advantages such as simple and convenient use and low implementation cost.

Cloud migration is the migration of applications or services from traditional platforms to cloud platforms. The cloud platforms have the advantages of powerful storage capacity, computing capacity, diversification of services and high cost-effectiveness. With the rise of cloud computing, more and more enterprises choose to migrate their applications or services to the cloud so as to achieve the purpose of saving internal network resources and costs. Therefore, cloud data centers will host a variety of services and applications, and how to achieve optimal resource allocation in cloud data

centers is particularly important. Reasonable resource allocation schemes can improve computing utilization and reduce energy consumption and network load.

From the point of view of cloud data center, the purpose of resource allocation is to allocate resources rationally for users, so as to improve resource utilization, reduce cost, and maximize revenue under the premise of certain computing power (CPU, memory, storage, etc.) and bandwidth resources. Meanwhile, from the enterprise users' point of view, the purpose of enterprise applications migration to the cloud is to maximize the utility of applications and save local network resources.

At present, the research on resource allocation in cloud migration focuses more on resource allocation (CPU, memory, storage) when applications are deployed in the cloud, and seldom considers the bandwidth allocation of virtual machines when applications layer access to host servers. Access links to the cloud are used for enterprise to transmit data from local data centers to cloud data centers, and bandwidth allocation will affect the migration time which is an important factor to measure enterprise users' satisfaction. In the research of resource allocation for enterprise applications deployment into the cloud, most scholars consider how to allocate resource reasonably to maximize their own benefits or to maximize users' utility from the single point of view of cloud resource providers or users. They seldom consider to minimize the cost of cloud resource providers in resource allocation, and at the same time, to maximize the effectiveness of enterprise users' application migration into the cloud, which is the main motivation and purpose of our work.

Our contributions are summarized as follows: we analyze the resource allocation of enterprise applications migration into the cloud, and construct the resource allocation which is divided into two stages: the bandwidth resource allocation for enterprise application migration to the cloud and physical resources allocation for enterprise applications deployment into the cloud. In the first stage, the bandwidth resource allocation is modeled as an optimization problem of allocating corresponding bandwidth resources for each component of applications. In the second stage, the physical resource allocation is formulated as an optimization problem of how to allocate the computing resources to each component of applications. The physical resource allocation model is decomposed into two independent sub-problems and interpreted from an economic point of view. In addition, a gradient-based resource allocation algorithm is designed for solving the model at each stage. Finally, numerical simulation results are given to illustrate the convergence of the proposed algorithms.

The rest of this paper is organized as follows: Section 2 gives related work on resource allocation in cloud migration; Section 3 introduces the architecture of enterprise applications migration to the cloud; Section 4 presents the bandwidth resource allocation model, gives the optimal bandwidth allocation analysis and proposes a gradient-based bandwidth allocation algorithm; Section 5 proposes the physical resource allocation model, gives the optimal resource allocation analysis and proposes a gradient-based resource allocation algorithm; Section 6 gives the simulation results and illustrates the convergence of the proposed algorithm; finally, Section 7 is the concluding remarks.

2. Related Work

At present, the research on cloud resource allocation can be divided into two categories: (i) the allocation of resources (CPU, storage and other cloud resources) in cloud deployment, and (ii) dynamic allocation of cloud bandwidth resource.

Barshan et al. (2017) [2] proposed a network-aware optimal resource allocation for application components with different characteristics. They also proposed centralized and hierarchical resource allocation with integer linear programming of application components, which maps application components to physical machines. Andrikopoulos et al. (2013) [3] proposed challenges and solutions for each layer when an enterprise migrates different parts of its applications to the cloud. The main solutions focus on migrating the whole application stack by means of virtulization and deployment on the cloud. Zhao and Zhou (2014) [4] divided the existing migration approaches into three types based on the cloud service models. Different migration processes need to be considered for different migration types, and different tasks will be involved accordingly. The goal of this work is to provide

an overall review for legacy system migration to the cloud. Hajjat et al. (2010) [5] proposed a migration model that uses hybrid migration approach, where enterprise operations are partly hosted on-premise and partly in the cloud. The migration model considers enterprise-specific constraints, cost savings, and increased transaction delays and wide-area communication costs due to the migration. This paper also articulates the importance of ensuring assurable reconfiguration of security policies as enterprise applications are migrated to the cloud. Hwang (2016) [6] proposed a model to tackle the migration challenges that transforms one resource into the same or another resource in hybrid clouds. Huang et al. (2014) [7] considered a problem of determining the optimal migrated components set of an enterprise application and address it in a way that is both scalable and deals inherently with network dynamicity. The authors also proposed the migration policies which are analytically shown to be capable of moving an enterprise application between local data center and remote cloud in a cost-efficient way. Their aim is to achieve the optimal set of migrated components of enterprise application, and at the same time, to realize the migration cost minimization and the migration utility maximization.

Hosseini et al. (2012) [8] proposed a cloud adoption tool that is evaluated using a case study of an organization that is considering the migration of some of its IT systems to the cloud. The decision makers have to be able to model the variations in resource usage and their systems deployment options to obtain accurate cost estimates in this way. Pantazoglou et al. (2016) [9] presented a decentralized method towards scalable and energy-efficient management of virtual machine (VM) instances that are provisioned by large enterprise clouds. Each compute operates autonomously and manages its own workload by applying a set of distributed load balancing schemes which reduce their power consumption. Mehfuz and Sahoo (2012) [10] presented challenges included in many types of enterprises, e.g., educational institutions, government, or large corporations. This paper aims to provide an understanding of migration challenges. It also seeks to propose a new cloud migration model that various type of organizations could follow to migrate to the cloud. In order to reduce cost of cloud data center, Marquez et al. (2015) [11] presented Cloud Migration Orchestrator (CMO), a framework for automation and coordination of large-scale cloud migration based on the IBM Business Process Management (BPM) technology with pre-migration analytics. At the same time, they presented a taxonomy of network challenges based on experience with migration of legacy environments, and discuss how to automate and optimize network configurations.

As for resource allocation of virtual machines, Shieh et al. (2011) [12] presented a proportional network sharing method based on virtual machine weight. Soudan et al. (2009) [13] proposed a fair resource allocation mechanism in cloud computing based on game theory. Guo et al. (2013) [14] modeled the bandwidth allocation problem of cloud data center as a bargaining game, and proposed a new bandwidth allocation protocol to achieve proportional and fair sharing in the network of virtual machine data center. Wilcox et al. (2011) [15] proposed a multi-objective genetic algorithm for VM deployment based on the classic NSGA-II algorithm to ensure minimization time of cloud migration. Hui et al. (2019) [16] presented SpongeNet, a bandwidth allocation solution, to ensure tenants' application performance.

It can be seen that optimal resource allocation in cloud data centers has received much attention and some interesting research results have been obtained. However, they do not consider both bandwidth allocation and physical resource allocation during migration and deployment into the cloud, which is the main aim of our work. In this paper we formulate resource allocation model for enterprise applications migration and deployment into the cloud and present resource allocation algorithm to achieve the optimum of resource allocation model, which is also illustrated through some numerical examples.

3. Architecture of Enterprise Applications Migration into Cloud

Enterprise applications typically consist of three layers: the front-end layer, the business logic layer, and the back-end layer. Generally, the front-end servers layer and business logic layer components are

allowed to partially or whole migrate to the cloud, but some enterprises need to keep back-end database servers which may involve security business data and their related components locally. The physical machines of cloud data center can host several virtual machines for components, which migrate on different physical machines according to their needs. The virtual machines on the same physical machine share the physical links of the underlying network, and the data streams generated by different virtual machines compete for network bandwidth resource. So access links to the cloud are used for enterprises to transmit data from local data centers to cloud data centers. The architecture for enterprise applications migration into the cloud is shown in Figure 1.

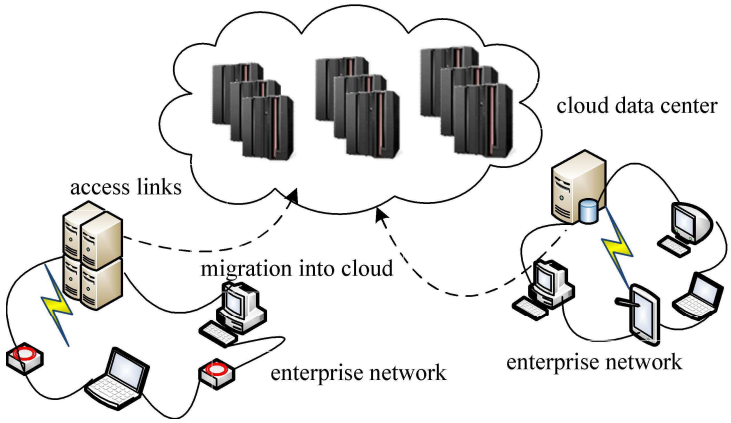


Figure 1. Architecture of enterprise applications migration into cloud.

The notations used in this paper are summarized in Table 1.

Table 1. Notation list.

Notations	Meanings
S	the set of applications, each element is application s
R	the set of components, each element is component r
P	the set of physical machines, each element is physical machine p
L	the set of access links, each element is link l
$R(s)$	the set of components for application s
$S(p)$	the set of applications using physical machine p
$P(r)$	the set of physical machines hosting component r
$S(l)$	the set of applications using access link l
$L(r)$	the set of links that component r uses
x_{sr}^{l}	bandwidth for component r of application s from link l
z_s	the total bandwidth for application s
C_l	the bandwidth capacity of link l
x_{sr}^{p}	resource for component r of application s from physical machine p
y_s	the total physical resource for application s
C_p	the resource capacity (e.g., CPU, memory, storage) of physical machine p
D_s	the load of application s

An application is composed of several components and a physical machine can host multiple virtual machines. Each virtual machine runs one component of the application, sharing physical machine resources at the granularity of virtual machine. The bandwidth resource in the cloud data center is shared with all virtual machines hosted by the same physical machine. Assuming that the set of applications to be migrated is S, each element is application $s \in S$; the set of components that make up an application is R, each element is component $r \in R$; the set of physical machines is P, each element is physical machine $p \in P$; the set of links is L, each element is link $l \in L$. $R(s)$ is the set of components for application s; $S(p)$ is the set of applications which use physical machine p; $P(r)$ is

the set of physical machines which host component r. $S(l)$ is the set of applications which use access link l; $L(r)$ is the set of links that component r uses for data transmission.

Let $x_{sr}^{l} \geq 0$ be the bandwidth resource that component r of application s obtains from link l. Then each application s obtains the total bandwidth resource $z_s = \sum_{r \in R(s)} \sum_{l \in L(r)} x_{sr}^{l}$. The bandwidth capacity of each link l is C_l, then the total bandwidth resource allocated by link l should not exceed the capacity of the access link, i.e., $\sum_{s \in S(l)} \sum_{r \in R(s)} x_{sr}^{l} \leq C_l$. Let $x_{sr}^{p} \geq 0$ be the physical resource that component r of application s obtains from physical machine p. Here, $x_{sr}^{\min} \leq x_{sr}^{p} \leq x_{sr}^{\max}$, where $x_{sr}^{\min}$ and $x_{sr}^{\max}$ are the minimum and maximum resource constraints that component r of application s should satisfy, respectively. Then each application s obtains the total physical resource $y_s = \sum_{r \in R(s)} \sum_{p \in P(r)} x_{sr}^{p}$. Let C_p denote the resource capacity (e.g., CPU, memory, storage) of physical machine p. Then the total physical resource offered to components should not exceed the capacity C_p.

4. Bandwidth Allocation for Enterprise Applications Migration

4.1. Model Description

A reasonable bandwidth resource allocation strategy can help cloud service providers avoid link congestion and improve enterprise applications performance. The more bandwidth resource enterprise users obtain, the shorter time required for cloud data center to complete parallel computing tasks. If enterprise users do not get enough bandwidth resource, it will cause delay of transmission time or longer task completion time. So the more bandwidth resource users obtain, the shorter time of cloud migration and the higher satisfaction of enterprise users.

In the bandwidth allocation of enterprise applications migration to the cloud, the corresponding resource is allocated to the virtual machines of cloud data center, and each virtual machine hosted by the physical machine runs one component of the application, that is to say, the bandwidth resource is allocated to each component of the application. So every application $s \in S$ occupies the total bandwidth resources $z_s = \sum_{r \in R(s)} \sum_{l \in L(r)} x_{sr}^{l}$. The capacity of each link is limited so each link can not carry more tasks than its capacity, which satisfies the inequality $\sum_{s \in S(l)} \sum_{r \in R(s)} x_{sr}^{l} \leq C_l$. The load of each application i is D_s. Enterprise users aim at completing their local applications migration to the cloud in a certain transmission time, so as to satisfy the migration satisfaction of enterprise users. Cloud data center has a certain deadline to complete the task of applications transmission, represented with T, the actual transmission time is $\sum_{s \in S} D_s / z_s$. Therefore, the bandwidth allocation for enterprise applications migration into the cloud can be modelled as the transmission time optimization problem **M** as follows:

$$\begin{aligned} \textbf{M:} \quad & \max \quad T - \sum_{s \in S} \frac{D_s}{z_s} \\ & \text{subject to} \quad \sum_{r \in R(s)} \sum_{l \in L(s)} x_{sr}^{l} = z_s, \forall s \in S \\ & \qquad\qquad \sum_{s \in S(l)} \sum_{r \in R(s)} x_{sr}^{l} \leq C_l, \forall l \in L \\ & \text{over} \qquad\quad x_{sr}^{l} \geq 0, s \in S, r \in R, l \in L. \end{aligned} \tag{1}$$

The goal of bandwidth allocation problem **M** for application components is to maximize transmission time difference, which depends on the total bandwidth allocation obtained by each application. So the problem **M** is about the allocation of bandwidth resource for the total resource z_s of applications.

The model (1) is regarded as the primal problem with primal variables $x = (x_{sr}^{l}, s \in S, r \in R, l \in L)$, and $z = (z_s, s \in S)$. Then we can obtain the following result.

Theorem 1. *The bandwidth allocation model (1) is a convex programming problem. The optimal bandwidth allocation for each application, i.e., z_s^*, exists and is unique. However, the optimal bandwidth allocation of each component of an application, i.e., x_{sr}^{l*}, is not necessarily unique.*

Proof of Theorem 1. From the nonlinear programming theory in Bertsekas (2003) [17], the objective of model (1) is concave with respect to primal variables z_s. The constraints of (1) are linear, so the feasible set of this optimization problem is compact. We can obtain that the bandwidth allocation model (1) for applications migration to the cloud is a convex programming problem. Further, the optimization problem is strictly convex with respect to primal variable $z = (z_s, s \in S)$, then the optimal bandwidth allocation $z^* = (z_s^*, s \in S)$ for applications exists and is unique as a consequence of strict convexity. However, the objective of (1) is not strictly concave with respect to primal variable $x = (x_{sr}^l, s \in S, r \in R, l \in L)$, thus the optimal bandwidth allocation of each component of an application can be not unique. □

4.2. Model Analysis

In order to investigate the optimum of model (1), we firstly give the Lagrangian of this nonlinear optimization problem

$$\begin{aligned} L(\mathbf{x}, \mathbf{z}; \omega, \mu) = & \left(T - \sum_{s \in S} \frac{D_s}{z_s} \right) + \sum_{s \in S} \omega_s \left(\sum_{r \in R(s)} \sum_{l \in L(r)} x_{sr}^l - z_s \right) \\ & + \sum_{l \in L} \mu_l \left(C_l - \sum_{s \in S(l)} \sum_{r \in R(s)} x_{sr}^l - \delta_l^2 \right), \end{aligned} \tag{2}$$

where $\omega = (\omega_s, s \in S)$ is the price vector with element $\omega_s \geq 0$, which can be considered as the price per unit bandwidth resource paid by the enterprise for its application s; $\mu = (\mu_l, l \in L)$ is the price vector with element μ_l, which can be considered as the price per unit bandwidth resource charged by link l; δ_l^2 is slack variable, which can be respectively considered as the spare capacity of link l.

The Lagrangian (2) can be rewritten as

$$L(\mathbf{x}, \mathbf{z}; \omega, \mu) = \left(T - \sum_{s \in S} \left(\frac{D_s}{z_s} + \omega_s z_s \right) \right) + \sum_{s \in S} \sum_{r \in R(s)} \sum_{l \in L(r)} x_{sr}^l \left(\omega_s - \mu_l \right) + \sum_{l \in L} \mu_l (C_l - \delta_l^2). \tag{3}$$

Notice that the first term in (3) is separable in z_s, and the second term is separable in x_{sr}^l. Thus the objective function of the dual problem is

$$D(\omega, \mu) = \max_{\mathbf{x}, \mathbf{z}} L(\mathbf{x}, \mathbf{z}; \omega, \mu) = \left(T + \sum_{s \in S} G_s(\omega_s) \right) + \sum_{s \in S} \sum_{r \in R(s)} \sum_{l \in L(r)} H_{sl}(\omega_s, \mu_l) + \sum_{l \in L} \mu_l (C_l - \delta_l^2), \tag{4}$$

where

$$G_s(\omega_s) = \max_{z_s} \left(- \frac{D_s}{z_s} - \omega_s z_s \right), \tag{5}$$

$$H_{sl}(\omega_s, \mu_l) = \max_{x_{sr}^l} x_{sr}^l \left(\omega_s - \mu_l \right). \tag{6}$$

Then, the optimal bandwidth allocation can be denoted as

$$z_s^*(\omega_s) = arg \max_{z_s} \left(- \frac{D_s}{z_s} - \omega_s z_s \right), \tag{7}$$

$$x_{sr}^{l*}(\mu_l) = arg \max_{x_{sr}^l} x_{sr}^l(\omega_s) \left(\omega_s - \mu_l \right), \tag{8}$$

where $\sum_{r \in R(s)} \sum_{l \in L(r)} x_{sr}^l(\omega_s) = z_s^*(\omega_s)$. Hence, the dual problem is

$$\begin{aligned} \mathbf{N:} \quad & \min \quad D(\omega, \mu) \\ & \text{over} \quad \omega_s \geq 0, \mu_l \geq 0, s \in S, l \in L. \end{aligned} \tag{9}$$

Assuming the optimal solution of bandwidth allocation model **M** and its dual problem **N** is $(x^*, z^*, \varpi^*, \mu^*)$, we can obtain the following result by analyzing the price paid by an application and those charged by links.

Theorem 2. *For the bandwidth allocation problem* **M**, *assume that* l_1 *and* l_2 *are the two links that component* r *of application* s *uses for data transmission. At the optimum of model (1), if the optimal bandwidth allocation are non-zero, then the prices charged by the links are equal, that is, if* $x_{sr}^{l_1*} \geq 0$ *and* $x_{sr}^{l_2*} \geq 0$, *then* $\mu_{l_1}^* = \mu_{l_2}^* = \varpi_s^*$.

Proof of Theorem 2. Indeed, at the optimum of the bandwidth allocation model (1) for applications migration to the cloud, from the Karush–Kuhn–Tucker (KKT) conditions for optimality of an optimization problem, we can deduce

$$\frac{D_s}{z_s^{*2}} - \varpi_s^* = 0, z_s^* > 0, \forall s \in S, \tag{10}$$

$$\varpi_s^* - \mu_l^* = 0, x_{sr}^* > 0, \forall s \in S, x_{sr}^{l*} > 0, \forall s \in S, l \in L(r), r \in R(s). \tag{11}$$

Here, Equations (10) and (11) are the necessary conditions for the existence of the optimal solution of bandwidth resource allocation problem **M**. Hence, two links that offer bandwidth resource for an application, e.g., $L_1, L_2 \in L(r), r \in R(s)$, the optimal prices charged by the two links are both equal to

$$\mu_l^* = \varpi_s^* = \frac{D_s}{z_s^{*2}}. \tag{12}$$

The result is obtained. □

4.3. Optimal Solution of the Model

The bandwidth resource of each link in the cloud center is limited. From the point of view of cloud resource providers, the objective is to maximize the resource utilization of each link and to reduce applications migration time. In the model **M**, δ_l^2 is the remaining bandwidth resource of link l. Obviously, according to KKT condition when $\delta_l^2 = 0$, the resource constraint corresponding to link l becomes active constraint. When $\delta_l^2 > 0$, the resource constraint corresponding to link l is non-active constraint. In the latter analysis, assume that all links satisfy $\delta_l^2 = 0, l \in L$. Otherwise, the non-active constraints can be omitted and only active constraints are considered. We rewrite the formulation (2) and obtain

$$L(\mathbf{z}; \mu) = \left(T - \sum_{s \in S} \left(\frac{D_s}{z_s} + \varpi_s z_s \right) \right) + \sum_{l \in L} \mu_l C_l. \tag{13}$$

Because of $\mu_{l_1} = \mu_{l_2} = \mu$, that is to say, there are two links offering bandwidth resource for one component of an application, then the prices charged by the two links are equal. Then substituting (12) into (13), we can rewrite (13) as follows

$$\widetilde{L}(\mu) = \left(T - \sum_{s \in S} 2\mu^{\frac{1}{2}} D_s^{\frac{1}{2}} \right) + \mu \sum_{l \in L} C_l. \tag{14}$$

Setting $d\widetilde{L}(\mu)/d\mu = 0$, we can obtain

$$\mu = \left(\frac{\sum_{s \in S} D_s^{\frac{1}{2}}}{\sum_{l \in L} C_l} \right)^2, \tag{15}$$

and

$$z_s = \frac{D_s^{\frac{1}{2}} \sum_{l \in L} C_l}{\sum_{s \in S} D_s^{\frac{1}{2}}}. \tag{16}$$

If $D_s = D$, that is, each application has the same workload, then $z_s = \frac{1}{|S|} \sum_{l \in L} C_l$, where $|S|$ is the number of applications which are migrated to the cloud.

4.4. Bandwidth Allocation Algorithm

In order to obtain the optimal bandwidth allocation of access links for components of applications when these applications are migrated into the cloud, we present the following bandwidth allocation algorithm

$$x_{sr}^l(t+1) = \left[x_{sr}^l(t) + \vartheta x_{sr}^l(t)(\varpi_s(t) - \mu_l(t)) \right]_{x_{sr}^l(t)}^+, \tag{17}$$

$$\varpi_s(t) = \frac{D_s}{(z_s(t))^2}, \tag{18}$$

$$z_s(t) = \sum_{r \in R(s)} \sum_{l \in L(r)} x_{sr}^l(t), \tag{19}$$

$$\mu_l(t+1) = \left[\mu_l(t) + \varsigma \frac{\rho_l(t) - C_l}{C_l} \right]_{\mu_l(t)}^+, \tag{20}$$

$$\rho_l(t) = \sum_{s \in S(l)} \sum_{r \in R(s)} x_{sr}^l(t), \tag{21}$$

where $\vartheta > 0, \varsigma > 0$ are step sizes of the algorithm, $a = [b]_c^+ = b$ if $c > 0$ and $a = [b]_c^+ = \max\{0, b\}$ if $c \leq 0$.

4.5. Basic Steps

The steps in the implementation of the algorithm are described as follows:

At the time $t = 1, 2, \ldots$

Step 1: Initialize variables and parameters. Choose the appropriate step sizes ϑ, ς. Initialize bandwidth allocation $x_{sr}^l(t)$ for component r of application s by access link l to the cloud data center.

Step 2: Calculate the prices paid by the enterprise for its applications. Compute the bandwidth allocation $z_s(t)$ obtained by each application i according to (19), and compute the price $\varpi_s(t)$ paid for access link l to the cloud data center according to (18).

Step 3: Calculate the prices charged by access links to the cloud data center. The cloud data center obtains the total bandwidth resource $\rho_l(t)$ occupied by multiple components of applications on each access link l according to (21), and update its price $\mu_l(t+1)$ according to (20).

Step 4: Update the bandwidth allocation of each component of an application. At the time $t+1$, update bandwidth allocation $x_{sr}^l(t+1)$ for component r of application s according to (17).

Step 5: Set the stop criterion.

When the equilibrium of the algorithm is achieved, the iteration procedure can be stopped and the optimum of bandwidth resource allocation can be obtained. The flowchart of bandwidth allocation scheme for enterprise applications migration is shown in Figure 2.

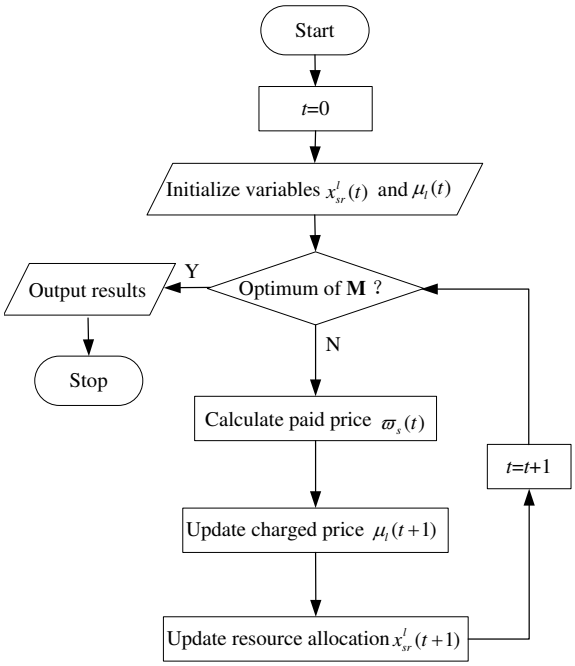


Figure 2. The flowchart of bandwidth allocation scheme for enterprise applications migration.

5. Resource Allocation for Enterprise Applications Deployment

5.1. Model Description

When the enterprise completes the applications migration to the cloud, each application s obtains a deployment utility function $U_s(\cdot)$ which quantifies the enterprise user's perceived satisfaction by obtaining a certain amount of resources from cloud resource providers. There are two types of applications that enterprises can offer for users [18]. One corresponds to the traditional data applications, such as file transfer, web services and e-mail. This type of applications are known as elastic applications. The other type of applications are delay and resource sensitive real-time applications, such as real-time streaming video and audio applications. They are known as inelastic applications. In this paper we main consider the elastic applications migration and deployment into the cloud. The deployment utility function of each application depends only on the physical machine resource y_s occupied by all components of the application, which is concave with respect to its obtained resource. Meanwhile, the resource provider which offers resource to application s occurs an operation cost $E_s(\cdot)$ which is a convex function during this application deployment into the cloud.

The process of enterprise applications migration to the cloud is migrating and deploying all components that make up an application into the physical machines in the cloud data center, so each application $s \in S$ occupies the total resource that is the aggregation of all its component, i.e., $y_s = \sum_{r \in R(s)} \sum_{p \in P(r)} x_{sr}^{p}$. Recall that every physical machine's resource capacity is limited, so the resources occupied by the components hosted by each physical machine should not exceed its maximal practical computing capacity, which satisfies the inequality $\sum_{s \in S(p)} \sum_{r \in R(s)} x_{sr}^{p} \leq \beta_p C_p$, where β_p is the physical machine resource threshold parameter. Then the reserved resource of physical machine p is $(1-\beta_p)C_p$, which is used for management of physical machines, for example, integrating and managing the virtual machines on this physical machine.

So, resource allocation for enterprise applications deployment into the cloud can be modelled as the problem of maximizing user deployment utility of enterprise applications into the cloud meanwhile minimizing the cost undertaken by cloud resource providers, which can be written as the following problem **Q**:

$$\begin{aligned} \mathbf{Q}: \quad & \max \quad \sum_{s\in S}\left(U_s(y_s)-E_s(y_s)\right) \\ & \text{subject to} \quad \sum_{r\in R(s)}\sum_{p\in P(r)} x_{sr}^{p}=y_s, \forall s\in S \\ & \qquad\qquad \sum_{s\in S(p)}\sum_{r\in R(s)} x_{sr}^{p}\le \beta_p C_p, \forall p\in P \\ & \text{over} \quad\quad x_{sr}^{\min}\le x_{sr}^{p}\le x_{sr}^{\max}, s\in S, r\in R, p\in P. \end{aligned} \tag{22}$$

The goal of cloud resources allocation problem **Q** for application components is to maximize the deployment utility of the applications minus the cost of the cloud resource providers, which depends on the total cloud resource allocation obtained by each application that consist of several components.

The resource allocation model **Q** with deployment utility functions and cost functions is regarded as the primal problem with primal variables $x=(x_{sr}^{p}, s\in S, r\in R, p\in P)$ and $y=(y_s, s\in S)$. It is not hard to find that the objective is strictly concave with respect to primal variables $y=(y_s, s\in S)$, but not strictly concave with respect to primal variables $x=(x_{sr}^{p}, s\in S, r\in R, p\in P)$. Then we can obtain the following result.

Theorem 3. *The cloud resources allocation model (22) is a convex programming problem. The optimal total cloud resources allocation for each application, i.e., y_s^*, exists and is unique. But the optimal cloud resources allocation of each component of an application, i.e., x_{sr}^{p*}, is not necessarily unique.*

Proof of Theorem 3. Recall that the deployment utility function $U_s(y_s)$ is concave with respect to its obtained resource y_s, and the operation cost function $C_s(y_s)$ is convex with respect to the offered resource y_s, then based on the nonlinear programming theory in Bertsekas (2003) [17], the objective of cloud resources allocation model (22) is concave with respect to primal variable y_s. Meanwhile, the constraints of (22) are linear, thereby the constraint set is convex. Therefore, the cloud resources allocation model (22) is a convex programming problem. Furthermore, the optimization problem is strictly convex with respect to primal variable $y=(y_s, s\in S)$, then the optimal resource allocation $y^*=(y_s^*, s\in S)$ for applications exists and is unique as a consequence of strict convexity. However, the objective of (22) is not strictly concave with respect to primal variable $x=(x_{sr}^{p}, s\in S, r\in R, p\in P)$, thus the optimal cloud resources allocation of each component of an application, i.e., x_{sr}^{p*}, is not necessarily unique. □

5.2. Model Analysis

In order to investigate the optimum of model (22), we firstly give the Lagrangian of this nonlinear optimization problem

$$\begin{aligned} L(\mathbf{x},\mathbf{y};\lambda,\mu)= & \sum_{s\in S}\left(U_s(y_s)-E_s(y_s)\right)+\sum_{s\in S}\lambda_s\left(\sum_{r\in R(s)}\sum_{p\in P(r)} x_{sr}^{p}-y_s\right) \\ & +\sum_{p\in P}\mu_p\left(\beta_p C_p-\sum_{s\in S(p)}\sum_{r\in R(s)} x_{sr}^{p}-\delta_p^2\right), \end{aligned} \tag{23}$$

where $\lambda=(\lambda_s, s\in S)$ is the price vector with element $\lambda_s\ge 0$, which can be considered as the price per unit cloud resources paid by the enterprise for its application s; $\mu=(\mu_p, p\in P)$ is the price vector with element $\mu_p\ge 0$, which can be considered as the price per unit cloud resource charged by physical machine p; δ_p^2 is the slack variable, which can be respectively considered as the spare capacity of physical machine p.

Then the Lagrangian (23) can be rewritten as

$$L(\mathbf{x},\mathbf{y};\lambda,\mu)=\sum_{s\in S}\left(U_s(y_s)-E_s(y_s)-\lambda_s y_s\right)+\sum_{s\in S}\sum_{r\in R(s)}\sum_{p\in P(r)} x_{sr}^{p}\left(\lambda_s-\mu_p\right)+\sum_{p\in P}\mu_p\left(\beta_p C_p-\delta_p^2\right) \tag{24}$$

Notice that the first term in (24) is separable in y_s, and the second term is separable in x_{sr}^p. Thus the objective function of the dual problem is

$$D(\lambda,\mu)=\max_{\mathbf{x},\mathbf{y}} L(\mathbf{x},\mathbf{y};\lambda,\mu)=\sum_{s\in S} A_s(\lambda_s)+\sum_{s\in S}\sum_{r\in R(s)}\sum_{p\in P(r)} B_{sp}(\lambda_s,\mu_p)+\sum_{p\in P}\mu_p\left(\beta_p C_p-\delta_p^2\right), \tag{25}$$

where

$$A_s(\varpi_s)=\max_{y_s} U_s(y_s)-E_s(y_s)-\lambda_s y_s, \tag{26}$$

$$B_{sp}(\lambda_s,\mu_p)=\max_{x_{sr}^p} x_{sr}^p\left(\lambda_s-\mu_p\right). \tag{27}$$

Then, the optimal cloud resource allocation can be denoted as

$$y_s^*(\lambda_s)=arg\max_{y_s} U_s(y_s)-E_s(y_s)-\lambda_s y_s, \tag{28}$$

$$x_{sr}^{p*}(\mu_p)=arg\max_{x_{sr}^p} x_{sr}^p(\lambda_s)\left(\lambda_s-\mu_p\right), \tag{29}$$

where $\sum_{r\in R(s)}\sum_{p\in P(r)} x_{sr}^p(\lambda_s)=y_s^*(\lambda_s)$.

We can interpret the sub-problems (26) and (27) from an economic point of view as follows. The sub-problem (26) is regarded as a system problem for enterprise users and cloud resource providers. In this problem, the enterprise user wants to maximize its own deployment utility of application s which depends on the total cloud resource y_s granted by cloud resource providers. Meanwhile, the enterprise has to pay a price for its using cloud resources. Since λ_s is the price per unit cloud resource paid by the enterprise user, then $U_s(y_s)-\lambda_s y_s$ indicates the benefit of the enterprise user when deploying application s into the cloud, $E_s(y_s)$ is the operation cost of cloud resources provider for application s. Thus, the problem (26) is a nonlinear optimization problem which is to maximize the migration utility of enterprise users minus the cost of cloud resource providers.

The sub-problem (27) is considered as the cloud data center problem. In this problem, $\lambda_s x_{sr}^p$ is the payment by the enterprise user for its application s when using cloud resource x_{sr}^p, μ_p is the price per unit resource charged by physical machine p, then $\mu_p x_{sr}^p$ is the charge of physical machine p. Hence the problem (27) is a linear optimization problem that each cloud resource provider is to maximize its own revenue.

Hence, the dual problem is

$$\begin{array}{lll}\textbf{D:} & \min & D(\lambda,\mu)\\ & \text{over} & \lambda_s\geq 0,\mu_p\geq 0, s\in S, p\in P.\end{array} \tag{30}$$

Assuming that the optimal solution of cloud resource allocation model **Q** and its dual problem **D** is $(\mathbf{x}^*,\mathbf{y}^*,\lambda^*,\mu^*)$, we can find the following result by analyzing the price paid by the enterprise user for an application and those charged by its physical machines

$$\mu_p^*=\lambda_s^*=U_s'(y_s)-E_s'(y_s). \tag{31}$$

Then, we can obtain the optimal resource allocation y_s^* of application s after substituting the specific deployment utility function and cost function into (31). Generally, we rewrite y_s^* as a function of price μ_p which is charged by physical machine p, i.e.,

$$y_s^*=\psi_s(\mu_p). \tag{32}$$

5.3. Optimal Resource Allocation

The objective of cloud resource providers is to maximize the resource utilization of each physical machine while satisfying the deployment satisfaction of enterprise applications. In the resource allocation model **Q**, δ_p^2 is the remaining cloud resource of physical machine p. Obviously, according to KKT conditions, when $\delta_p^2 = 0$, the resource constraint corresponding to the physical machine p is active. When $\delta_p^2 > 0$, the resource constraint corresponding to the physical machine p is non-active. In the following analysis, assume that all physical machines satisfy $\delta_p^2 = 0$. Otherwise, the non-active constraints can be omitted and only active constraints are considered. Then we rewrite the Lagrangian (23) and obtain

$$\begin{aligned}\overline{L}(\mathbf{x},\mu) = & \sum_{s\in S}\left(U_s\left(\sum_{r\in R(s)}\sum_{p\in P(r)} x_{sr}^p\right) - E_s\left(\sum_{r\in R(s)}\sum_{p\in P(r)} x_{sr}^p\right)\right) \\ & + \sum_{p\in P}\mu_p\left(\beta_p C_p - \sum_{s\in S(p)}\sum_{r\in R(s)} x_{sr}^p\right).\end{aligned} \tag{33}$$

Because of $\mu_{p_1} = \mu_{p_2} = \mu$, we can rewrite (33) as follows

$$\begin{aligned}\overline{L}(\mathbf{x},\mu) & = \sum_{s\in S}\left(U_s\left(\sum_{r\in R(s)}\sum_{p\in P(r)} x_{sr}^p\right) - E_s\left(\sum_{r\in R(s)}\sum_{p\in P(r)} x_{sr}^p\right)\right. \\ & \quad \left. -\mu\sum_{r\in R(s)}\sum_{p\in P(r)} x_{sr}^p\right) + \mu\sum_{p\in P}\beta_p C_p, \\ & = \sum_{s\in S}\left(U_s(\psi_s(\mu)) - E_s(\psi_s(\mu)) - \mu\psi_s(\mu)\right) + \mu\sum_{p\in P}\beta_p C_p.\end{aligned} \tag{34}$$

Setting $\partial\overline{L}(\mathbf{x},\mu)/\partial\mu = 0$, we can obtain

$$\sum_{s\in S}\left[\frac{\partial\psi_s(\mu)}{\partial\mu}\left(\frac{\partial E_s(\psi_s(\mu))}{\partial\psi_s(\mu)} - \frac{\partial U_s(\psi_s(\mu))}{\partial\psi_s(\mu)} + \mu\right) + \psi_s(\mu)\right] = \sum_{p\in P}\beta_p C_p. \tag{35}$$

When selecting a specific form of utility function and cloud data center cost function, we can obtain specific expression of cloud resource provider price μ^* and optimal resource allocation y_s^*. Next, we analyze the optimal solution of the resource allocation model for enterprise application deployment into the cloud when choosing specific forms of utility function.

5.4. Further Discussions

In above analysis we only assume the deployment utility function is concave. Indeed there are many specific forms of utility function for enterprise applications. For example, we choose the following form of utility function for elastic applications proposed in Vo et al. (2012) [19] and also discussed in Li and Sun [20] and Li et al. [18]

$$U_s(y_s) = w_s\log(y_s + 1), \tag{36}$$

where w_s is the payment for cloud resource providers when the enterprise user migrates its application s to the cloud.

We then choose the following cost function proposed in Tso, Jouet, and Pezaros (2016) [21],

$$E_s(y_s) = \sigma y_s + \phi, \tag{37}$$

where σ represents the cost undertaken by the cloud resource providers when an application occupies per unit resource, ϕ represents the fixed cost undertaken by the cloud resource providers to maintain their normal operation in cloud data center.

Next, we substitute utility function (36) and cost function (37) into the resource allocation model and its corresponding analysis. In particularly, we rewrite (31) as follows

$$\mu_p^* = \lambda_s^* = U_s'(y_s) - E_s'(y_s) = \frac{w_s}{y_s^* + 1} - \sigma = \frac{w_s}{1 + \sum\limits_{r \in R(s)} \sum\limits_{p \in P(r)} x_{sr}^{p*}} - \sigma. \tag{38}$$

Then we obtain

$$\mu_p^* = \frac{\sum\limits_{s \in S} w_s}{|S| + \sum\limits_{p \in P} \beta_p C_p} - \sigma, y_s^* = \frac{w_s}{\sum\limits_{s \in S} w_s} \left(|S| + \sum_{p \in P} \beta_p C_p \right) - 1. \tag{39}$$

We can find that when the enterprise user migrates its applications into the cloud, the optimal total cloud resource offered to each application depends on the number of applications, the total resource of physical machines in the cloud center, and the total payment weighted by the payment for the application. Obviously, the optimal total cloud resource granted for each application is unique. If we choose other form of utility functions, we can also obtain the optimal resource allocation for each application.

5.5. Resource Allocation Algorithm

5.5.1. Basic Algorithm

In order to obtain the optimal resource allocation for components of enterprise applications on physical machines in the cloud data center, we present the following resource allocation algorithm

$$x_{sr}^p(t+1) = [x_{sr}^p(t) + \kappa x_{sr}^p(t)(\lambda_s(t) - \mu_p(t))]_{x_{sr}^{\min}}^{x_{sr}^{\max}}, \tag{40}$$

$$\lambda_s(t) = U_s'(y_s(t)) - E_s'(y_s(t)), \tag{41}$$

$$y_s(t) = \sum_{r \in R(s)} \sum_{p \in P(r)} x_{sr}^p(t), \tag{42}$$

$$\mu_p(t+1) = \left[\mu_p(t) + \tau \frac{\xi_p(t) - \beta_p C_p}{\beta_p C_p} \right]_{\mu_p(t)}^{+}, \tag{43}$$

$$\xi_p(t) = \sum_{s \in S(p)} \sum_{r \in R(s)} x_{sr}^p(t), \tag{44}$$

where $\kappa > 0, \tau > 0$ are step sizes of the algorithm, and $a = [b]_c^d = \min\{d, \max\{b, c\}\}$.

The cloud physical machine p computes resource $\xi_p(t)$ occupied by multiple components of the enterprise applications hosted by physical machine p according to (44), and updates its price $\mu_p(t+1)$ according to (43). Meanwhile, each application s computes its cloud resource $y_s(t)$ offered by physical machines according to (42), and obtains the price $\lambda_s(t)$ paid for cloud resource providers according to (41). Then physical machine p updates cloud resource allocation $x_{sr}^p(t)$ for component r of application s according to (40). Obviously, resource allocation $x_{sr}^p(t)$ of physical machine p for each component r of application s only depends on resource price $\lambda_s(t)$ that the enterprise user pays for cloud resource providers and resource price $\mu_p(t)$ which is charged by the cloud physical machine p.

The algorithm is an iteration process based on gradient method. According to convex optimization theory, the algorithm can converge to the equilibrium point within a limited number of iterations,

that is, the optimal solution of resource allocation model for enterprise applications migration into the cloud.

Due to nonuniqueness of optimal resource allocation for each component of an application, the proposed algorithm above may oscillate around an optimum. In order to overcome it, we will improve the algorithm by applying the low-pass filtering approach which does not change the optimal solution.

5.5.2. Improved Algorithm

In order to eliminate the oscillation and also improve the convergence speed, we introduce an augmented variable $\widetilde{x}_{sr}^{p}(t)$, which is regarded as the optimal estimation of resource allocation $x_{sr}^{p}(t)$ for component r of application s. Then rewrite the Equation (40) as following

$$x_{sr}^{p}(t+1) = [(1-\theta)x_{sr}^{p}(t) + \theta\widetilde{x}_{sr}^{p}(t) + \theta\kappa x_{sr}^{p}(t)(\lambda_s(t) - \mu_p(t))]_{x_{sr}^{\min}}^{x_{sr}^{\max}}, \tag{45}$$

$$\widetilde{x}_{sr}^{p}(t+1) = [(1-\theta)\widetilde{x}_{sr}^{p}(t) + \theta x_{sr}^{p}(t)]_{x_{sr}^{\min}}^{x_{sr}^{\max}}, \tag{46}$$

where θ is the parameter for low-pass filtering in the improved algorithm. We can obtain from the filtering theory that at the optimum, i.e., $x_{sr}^{p*} = \widetilde{x}_{sr}^{p*}$. The oscillation is eliminated by augmented variable $\widetilde{x}_{sr}^{p}(t)$ without changing the optimal solution.

The proposed resource allocation algorithm above applies the first-order Lagrangian method and low filtering theory, and can be proven to be convergent. Meanwhile, due to the model is not strictly concave and optimal resource allocation is not unique, the basic algorithm may have an oscillation problem around the equilibrium. Therefore, the low-pass filtering method in the improved scheme is used to eliminate the oscillation and also improve the convergence speed. In fact, the step sizes have an significant impact on the convergence speed. The step sizes used in the algorithm should be small enough to guarantee convergence, but not so small such that the convergence becomes unnecessarily very slow. Thus we need to choose appropriate step sizes to ensure the optimum is realized within a certain number of iterations.

5.5.3. Basic Steps

The steps in the implementation of the algorithm are described as follows:

At time $t = 1, 2, \ldots$

Step 1: Initialize variables and parameters.

Choose the appropriate step sizes κ and τ, and filtering parameter θ. Initialize resource allocation $x_{sr}^{p}(t)$ for component r of application s by cloud physical machine p.

Step 2: Calculate the price paid by the enterprise user for each application.

Compute the cloud resource $y_s(t)$ occupied by each application s according to (42), and obtain the price paid for cloud resource providers according to (41).

Step 3: Calculate the price charged by each physical machine in the cloud data center.

The cloud physical machine p obtains the resource $\xi_p(t)$ offered to multiple components of applications hosted by physical machine p according to (44), and updates its charged price $\mu_p(t+1)$ according to (43).

Step 4: Update the resource allocation for each component of an application.

At time $t+1$, update cloud resource allocation $x_{sr}^{p}(t+1)$ for component r of application s according to (45) and (46).

Step 5: Set the stop criterion.

When the equilibrium of the algorithm is achieved, the iteration procedure can be stopped and the optimum of resource allocation can be obtained. The flowchart of resource allocation for enterprise applications deployment is shown in Figure 3.

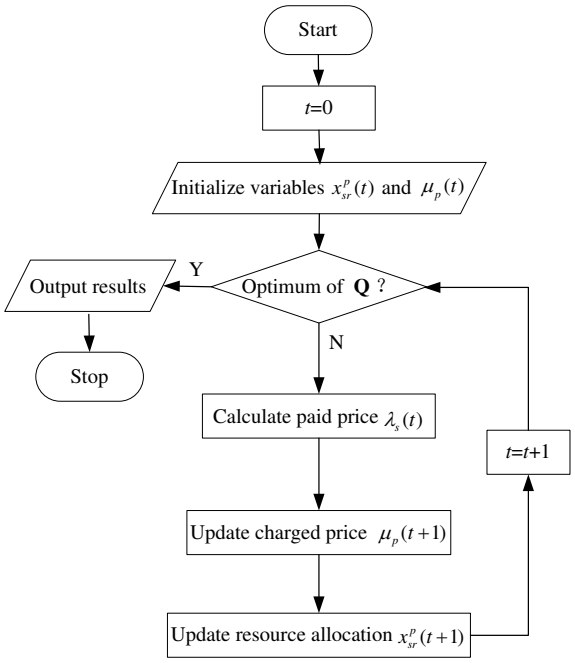


Figure 3. The flowchart of resource allocation scheme for enterprise applications deployment.

6. Simulation and Analysis

6.1. Cloud Migration and Deployment Description

Assuming there is an enterprise who intends to migrate its applications into the cloud. There are three disposable physical machines in the cloud center to host the enterprise applications. The enterprise migrates its three applications with eight components into the cloud center, i.e., application 1 with two components, application 2 with four components, and application 3 with two components. Each component is run by a virtual machine, and the virtual machines deployment in the cloud center is shown in Figure 4. Each physical machine in the cloud data center accesses a separate link for application migration.

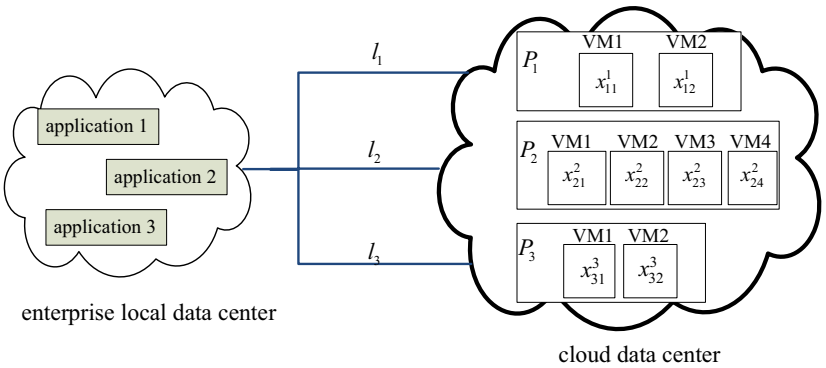


Figure 4. An example of enterprise applications migration and deployment into cloud center.

6.2. Bandwidth Allocation for Enterprise Applications Migration

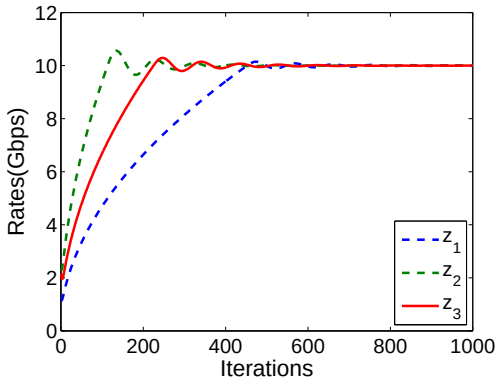
In the phase of enterprise applications migration, our aim it to achieve optimal bandwidth allocation. Assume the capacity of the access links is $C = (C_1, C_2, C_3) = (10, 10, 10)$ Gbps. In the bandwidth allocation algorithm we choose parameters $\vartheta = 0.1$ and $\varsigma = 0.1$, and the load of these applications is $D = (D_1, D_2, D_3) = (10, 10, 10)$ G. The initial transmission rate of each application on each link is 0.05Gbps. The optimum obtained by the proposed algorithm is listed in Table 2.

The optimal solution solved by nonlinear programming software LINGO is also presented in the table. We can find that the algorithm is convergent and the optimal bandwidth allocation is obtained.

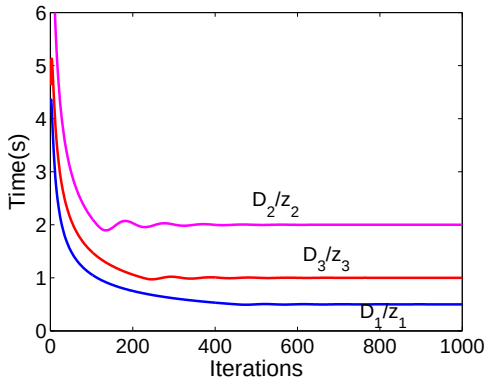
Table 2. The optimal bandwidth allocation for applications migration.

Variable	x_{11}^{1*}	x_{12}^{1*}	x_{21}^{2*}	x_{22}^{2*}	x_{23}^{2*}	x_{24}^{2*}	x_{31}^{3*}	x_{32}^{3*}
algorithm	5.0	5.0	2.5	2.5	2.5	2.5	5.0	5.0
LINGO	5.0	5.0	2.5	2.5	2.5	2.5	5.0	5.0

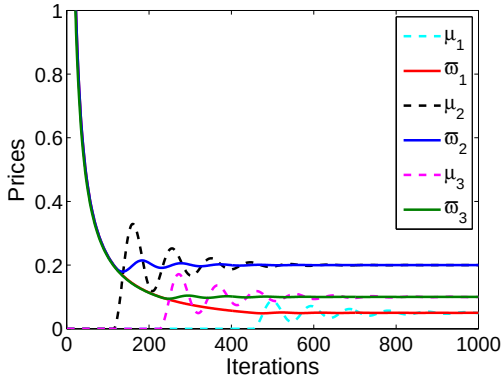
The simulation results of the bandwidth allocation algorithm (17)–(19) are also given in Figure 5, which shows the bandwidth allocation for each application on the access links, the migration time of three applications, and the price paid by the enterprise user along with the prices charged by the access links. The algorithm is gradually driven to a steady state where the utilization of the access links to the cloud data center is approximately 100% as we observed in the simulation. We can also observe that the optimal bandwidth allocation is achieved within a limited number of iterations.



(**a**) Bandwidth allocation for applications migration



(**b**) Migration time of applications

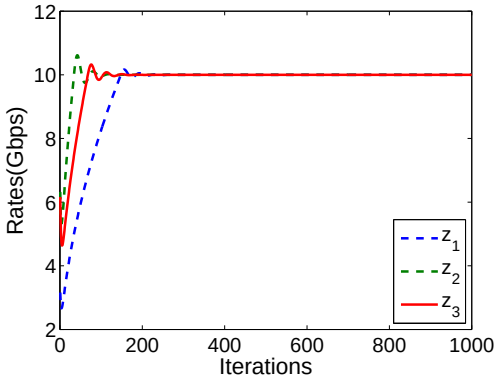


(**c**) Prices of links

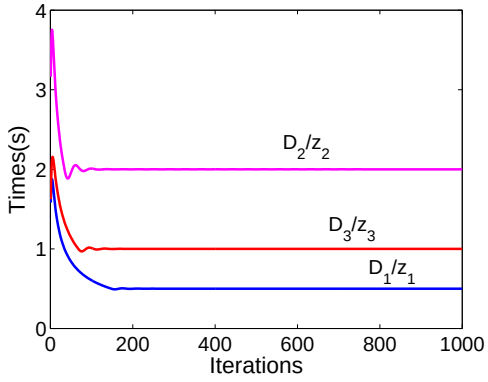
Figure 5. Performance of the bandwidth allocation algorithm for application migration with $\vartheta = 0.1$ and $\varsigma = 0.1$.

Now we investigate the convergence speed of the bandwidth allocation algorithm for application migration. The simulation setup is identical with the previous one except that we choose different parameters. For example, we choose larger step sizes $\vartheta = 0.4$ and $\varsigma = 0.6$ and depict the performance results in Figure 6. We find that the convergence speed is obviously improved with the larger step sizes. Indeed, the bandwidth allocation algorithm is a subgradient-based scheme. The convergence speed

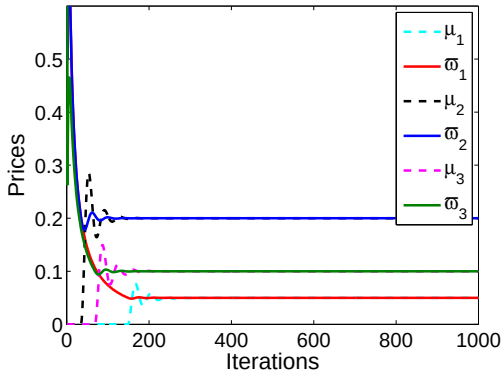
mainly depends on parameters such as step sizes other than workload of applications or capacity of the access link to the cloud. Generally, the step sizes used in the algorithm should be small enough to guarantee convergence, but not so small such that the convergence becomes unnecessarily very slow. Meanwhile, the step sizes should also be not so big that the algorithm may oscillate around the optimum. It is very necessary to choose appropriate step sizes so ensure the optimum can be achieved within reasonable convergence times.



(**a**) Bandwidth allocation for applications migration



(**b**) Migration time of applications



(**c**) Prices of links

Figure 6. Performance of the bandwidth allocation algorithm for application migration with $\vartheta = 0.4$ and $\varsigma = 0.6$.

6.3. Resources Allocation for Enterprise Applications Deployment

In the phase of enterprise elastic applications deployment into the cloud, the physical machines allocate their resources such as CPU, memory and storage to virtual machines to host the components of enterprise applications. Here, we choose CPU resource as an example to learn the performance of the proposed resource allocation algorithm. However, the scheme can be applied into the scenario where memory and/or storage resources are considered. Assume the capacity of the three physical machines is $C = (C_1, C_2, C_3) = (1200, 2000, 1600)$ MIPS, where MIPS (million instructions per second) is an indicator of CPU operation speed.

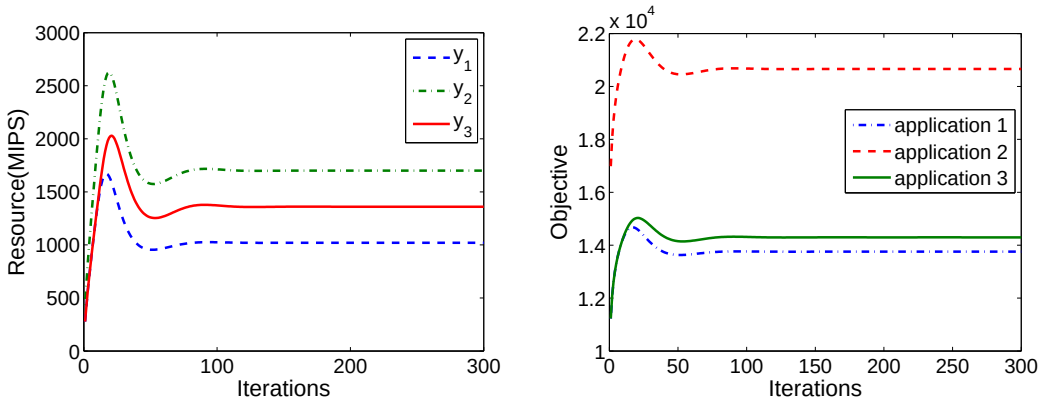
The payment of the enterprise user for its applications when deploying into the cloud is $w = (w_1, w_2, w_3) = (2000, 2800, 2000)$. In the resource allocation algorithm we choose filtering parameter $\theta = 0.2$, step sizes $\kappa = 0.2$ and $\tau = 0.5$, and parameters $\sigma = 0.1$, $\beta = (\beta_1, \beta_2, \beta_3) = (0.85, 0.85, 0.85)$. Initialize resource allocation for each component of applications $x_{sr}^p = 50$ MIPS. We list the optimum obtained by the proposed algorithm in Table 3. Also, we present the optimal solution solved by

nonlinear programming software LINGO in the table. It is not hard to find that the proposed algorithm is finally convergent to the optimum.

We also depict the simulation results of the proposed algorithm (40)–(46) in Figure 7, which shows the resource allocation for each application offered by physical machines, migration utility of three applications, and the price paid by the enterprise user along with the price charged by cloud physical machines. We can find that the resource allocation scheme can converge to the optimum within a certain number of iterations.

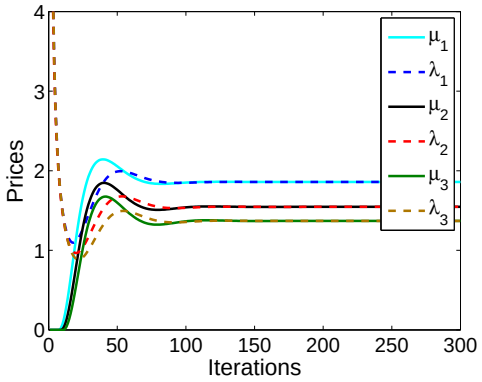
Table 3. The optimal resource allocation for applications deployment.

Variable	x_{11}^{1*}	x_{12}^{1*}	x_{21}^{2*}	x_{22}^{2*}	x_{23}^{2*}	x_{24}^{2*}	x_{31}^{3*}	x_{32}^{3*}
algorithm	510	510	425	425	425	425	680	680
LINGO	510	510	425	425	425	425	680	680



(**a**) Resource allocation for applications deployment

(**b**) Deployment objectives of applications



(**c**) Prices of physical machines

Figure 7. Performance of the resource allocation algorithm for application deployment in a simple scenario.

Now we consider the performance of the proposed resource allocation algorithm for application deployment into the cloud in different scenarios, e.g., different numbers of applications as well as physical machines. The algorithm has the same parameters as the aforementioned scenario, but is applied into larger scale cloud data centers. We depict the evolution of aggregated deployment utility of applications in Figure 8. We find that the size or scale of the cloud data center has no obvious affect on the convergence speed of the algorithm. The final optimal objective increases with the number of applications or physical machines but, in all scenarios, the value is achieved within almost the same

number of iterations (e.g., 200 iterations). Therefore, the number of applications or physical machines does not change the convergence speed obviously. Actually, the resource allocation algorithm for application deployment into the cloud is also a subgradient-based scheme. The convergence speed mainly depends on algorithm parameters such as step sizes other than the number of applications or physical machines, as we have discussed in Section 5.2.

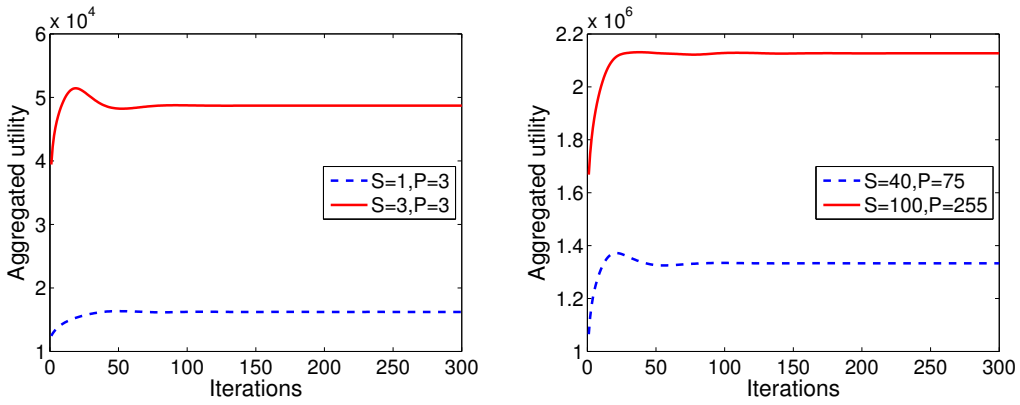


Figure 8. Performance of the resource allocation algorithm for application deployment in different scenarios.

7. Conclusions

This paper discusses the resource allocation of enterprise applications migration and deployment into the cloud, and divides it into two stages which are bandwidth resource allocation of enterprise applications migration to cloud and resource allocation of enterprise applications deployment into the cloud. In the first stage, our aim is to minimize the migration time to the cloud. We propose the bandwidth allocation for applications migration to the cloud and design a bandwidth allocation algorithm to realize the optimal bandwidth allocation for each component of the applications. In the second stage, we present the resources allocation of physical machines for enterprise applications during cloud deployment which is decomposed into two independent sub-problems and interpreted from an economic point of view. We analyze the relationship between the prices charged by the enterprise user and those paid by physical machines of cloud data center and give a gradient-based resource allocation algorithm which can achieve optimal resource allocation. Finally, some numerical simulation results are given to illustrate the effectiveness and convergence of the proposed algorithm. For further research work, we will investigate the resource allocation for applications whose components support elastic and/or inelastic services, and apply the proposed algorithm to achieve optimal resource allocation for inelastic services or even multiclass services.

Author Contributions: These authors have contributed equally to the development of this model and algorithm and its writing.

Funding: This research was supported in part by the National Natural Science Foundation of China (71671159, 71971188), the Humanity and Social Science Foundation of Ministry of Education of China (16YJC630106), the China Postdoctoral Science Foundation (2018T110205), the Natural Science Foundation of Hebei Province (G2018203302), the project Funded by Four Batch of Talents Program in Hebei Province, the project Funded by Hebei Education Department (BJ2017029) and Hebei Talents Program (A2017002108).

Acknowledgments: The authors would like to thank the anonymous reviewers and Associate Editor for very detailed and helpful comments and suggestions to improve this work.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Li, C.; Yuan, L. Optimal resource provisioning for cloud computing environment. *J. Supercomput.* **2012**, *62*, 989–1022. [CrossRef]

2. Barshan, M.; Moens, H.; Latre, S. Algorithms for network-aware application component placement for cloud resource allocation. *J. Commun. Netw.* **2017**, *19*, 493–508. [CrossRef]
3. Andrikopoulos, V.; Binz, T.; Leymann, F.; Strauch, S. How to adapt applications for the Cloud environment: Challenges and solutions in migrating applications to the Cloud. *Computing* **2013**, *95*, 493–535. [CrossRef]
4. Zhao, J.F.; Zhou, J.T. Strategies and methods for cloud migration. *Int. J. Autom. Comput.* **2014**, *11*, 143–152. [CrossRef]
5. Hajjat, M.; Sun, X.; Sung, Y.; Maltz, D.; Rao, S.; Sripanidkulchai, K.; Tawarmalani, M. Cloudward bound: Planning for beneficial migration of enterprise applications to the cloud. *ACM SIGCOMM Comput. Commun. Rev.* **2010**, *40*, 243–254. [CrossRef]
6. Hwang, J. Toward beneficial transformation of enterprise workloads to hybrid clouds. *IEEE Trans. Netw. Serv. Manag.* **2016**, *13*, 295–307. [CrossRef]
7. Huang, D.; Yi, L.; Song, F.; Yang, D.; Zhang, H. A secure cost-effective migration of enterprise applications to the cloud. *Int. J. Commun. Syst.* **2014**, *27*, 3996–4013. [CrossRef]
8. Hosseini, A.; Greenwood, D.; Smith, J.D. The Cloud Adoption Toolkit: supporting cloud adoption decisions in the enterprise. *J. Softw. Pract. Exp.* **2012**, *42*, 447–465. [CrossRef]
9. Pantazoglou, M.; Tzortzakis, G.; Delis, A. Decentralized and energy-efficient workload management in enterprise clouds. *IEEE Trans. Cloud Comput.* **2016**, *4*, 196–209. [CrossRef]
10. Mehfuz, S.; Sahoo, G. A five-phased approach for the cloud migration. *Int. J. Emerg. Technol. Adv. Eng.* **2012**, *2*, 286–291.
11. Marquez, L.; Rosado, D.G.; Mouratidis, H.; Mellado, D.; Medina, E. A framework for secure migration processes of legacy systems to the cloud. *Lect. Notes Bus. Inf. Process.* **2015**, *215*, 507–517.
12. Shieh, A.; Kandula, S.; Greenberg, A.; Kim, C.; Saha, B. Sharing the data center network. In Proceedings of the 8th USENIX Symposium on Networked Systems Design and Implementation, Boston, MA, USA, 30 March–1 April 2011.
13. Soudan, S.; Chen, B.; Primet, P. Flow scheduling and endpoint rate control in grid networks. *Future Gener. Comput. Syst.* **2009**, *25*, 904–911. [CrossRef]
14. Guo, J.; Liu, F.; Tang, H.; Lian, Y.; Jin, H.; Lui, J. Falloc: Fair network bandwidth allocation in iaas datacenters via a bargaining game approach. In Proceedings of the 21st IEEE International Conference on Network Protocols, Goettingen, Germany, 7–10 October 2013.
15. Wilcox, D.; McNabb, A.; Seppi, K. Solving virtual machine packing with a reordering grouping genetic algorithm. In Proceedings of the 2011 IEEE Congress of Evolutionary Computation (CEC), New Orleans, LA, USA, 5–8 June 2011; pp. 362–369.
16. Hui, Y.; Hai, Y.; Hui, W. Towards predictable performance via two-layer bandwidth allocation in cloud datacenter. *J. Parallel Distrib. Comput.* **2019**, *126*, 34–47.
17. Bertsekas, D.B. *Nonlinear Programming*; Athena Scientific: Belmont, MA, USA, 2003.
18. Li, S.; Sun, W.; Tian, N. Resource allocation for multi-class services in multipath networks. *Perform. Eval.* **2015**, *92*, 1–23. [CrossRef]
19. Vo, P.L.; Lee, S.W.; Hong, C.S. The random access NUM with multiclass traffic. *Eurasip J. Wirel. Commun. Netw.* **2012**, *2012*, *242*. [CrossRef]
20. Li, S.; Sun, W. A mechanism for resource pricing and fairness in peer-to-peer networks. *Electron. Commer. Res.* **2016**, *16*, 425–451. [CrossRef]
21. Tso, F.P.; Jouet, S.; Pezaros, D.P. Network and server resource management strategies for data centre infrastructures: A survey. *Comput. Netw.* **2016**, *106*, 209–225. [CrossRef]