Optimizing Workload Category for Adaptive Workload Prediction in Service Clouds

Chunhong Liu[1](✉), Yanlei Shang[1], Li Duan[1,2], Shiping Chen[2], Chuanchang Liu[1], and Junliang Chen[1]

[1] State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China
{liuchunhong2012,shangyl,duanli,lcc3265,chjl}@bupt.edu.cn
[2] DATA61, Commonwealth Scientific and Industrial Research Organization, Sydney, Australia
Shiping.Chen@csiro.au

Abstract. It is important to predict the total workload for facilitating auto scaling resource management in service cloud platforms. Currently, most prediction methods use a single prediction model to predict workloads. However, they cannot get satisfactory prediction performance due to varying workload patterns in service clouds. In this paper, we propose a novel prediction approach, which categorizes the workloads and assigns different prediction models according to the workload features. The key idea is that we convert workload classification into a 0–1 programming problem. We formulate an optimization problem to maximize prediction precision, and then present an optimization algorithm. We use real traces of typical online services to evaluate prediction method accuracy. The experimental results indicate that the optimizing workload category is effective and proposed prediction method outperforms single ones especially in terms of the platform cumulative absolute prediction error. Further, the uniformity of prediction error is also improved.

Keywords: Cloud computing · Resource provisioning · Workload prediction · 0–1 programming

1 Introduction

Service clouds are a kind of service platform using cloud computing technology, on which lots of application systems are running. These applications are designed according to Service-Oriented Architecture (SOA), and it encapsulates business processes into services. Adoption of the new platform paradigm by service provider could make energy and cost to reduce obviously due to the resource provisioning advantages of cloud computing.

With the ability of dynamic provisioning resources, cloud computing platform can adjust the resource to meet the demand of application, which is an important characteristic and is different from traditional platform. By using auto scaling technology, service clouds provide on-demand resource to users. Service providers

A. Barros et al. (Eds.): ICSOC 2015, LNCS 9435, pp. 87–104, 2015.
DOI: 10.1007/978-3-662-48616-0_6

are free from considering over-provisioning or under-provisioning for a service. In order to enable the scalability, it is necessary to develop a mechanism to scale up or down virtual machine (VM) automatically. Many cloud providers, such as AWS EC2 [1] and Google the App Engine [2], focus on the scalability, that means providing on-demand computing power and storage capacities dynamically. In these cases, cloud computing is a good choice to liberate service providers from deploying physical infrastructures.

In order to take advantage of scalability, service clouds need to support auto scaling technology in a way that service clouds manage VM instances automatically [3]. Therefore, it is of important significance to realize the resources predistribution via predicting workload. Workload prediction is a key step to realize predistribution and improve the accuracy of distribution in service clouds.

Currently, most workload prediction methods generally utilize a single prediction method [3–18], which choose special predicting model aiming at a certain workload feature. However, these methods could not get satisfactory prediction performance for service clouds. Because the service is different in service clouds and the service workload changes in a variety forms.

Service workloads have various patterns influenced by service type. As key characteristic of workloads, burstiness and self-similarity [19–21] have been reported for some kinds of application in cloud. There are also research works [22–24] analyze that the actual cloud computing workloads are highly time-varying in nature. Wang [21] classified the workloads into slow time-scale data and fast time-scale data according to the speed of load change in form of time-series employing. They described the relationship of two kinds of workload and characterized the impact of the workload on the value of dynamic resizing.

Focus on the variation pattern of workloads, a prediction method based on feature discriminations could be used to obtain a more accurate prediction effect in service clouds. In the project, the common workload classification method is to extract the feature of workload, classify the workload by comparing the feature value with a threshold value. Therefore, the determination of threshold value greatly influence the workload category. However, the workloads are dynamically changing in service clouds. The threshold need dynamic change with the service numbers floating on account of the service adding or reducing at any time in service clouds. However, dynamic adjustment of the threshold needs a lot of historical statistics or experience, and this increases the computation and management difficulty. The key problem of workload classified prediction is how to catalog workload effectively, which is the crux of the matter to obtain more accurate total prediction results.

In this paper, we propose a prediction approach based on feature discrimination. The key idea is that we transform the workload classification threshold problem into the task assignment one. By modeling and solving the integer programming, the service is allocated a suitable forecasting method automatically. We formulate an optimization problem to maximize prediction precision. The problem is then converted to a 0–1 programming problem. Our objective is to design an intelligent workload prediction management architecture with more robust characteristics to adapt various workload change modes and minimize the predicting error of service clouds. Our contributions are summed up as follows:

1. We propose a categorical prediction approach according to different change mode of workload in order to get higher platform prediction accuracy. Our work applies feedback from the latest observed workloads to a prediction model and update the model parameter on the run. Our method has well robustness and adaptive ability.
2. We present the optimal way to classify workloads in order to allocate prediction models adaptively. We establish a 0–1 programming model and use the sum of l_2-norm of workload average rate to trade off the prediction accuracy and time. The optimizing solution can dynamically determine the type of service workloads effectively.

The rest of the paper is organized as follows. Section 2 describes the design of our adaptive management architecture for workload prediction. The workload classification optimize model is established and solution is given online in Sect. 3. Section 4 shows the evaluation. We present the related work in Sect. 5, and conclude this paper with Sect. 6.

2 System Architecture

We propose the architecture of workload prediction based on feature discrimination in service clouds. In the proposed architecture, adaptive workload management can automatically allocate a prediction model to each service by solving the optimal problem.

Figure 1 shows the architecture of the classified workload prediction in service clouds. There are Admission Controller module, Resource Manager module, Predictor module and Data Warehouse module in the proposed architecture. Infrastructures include compute, storage and network resources. The services

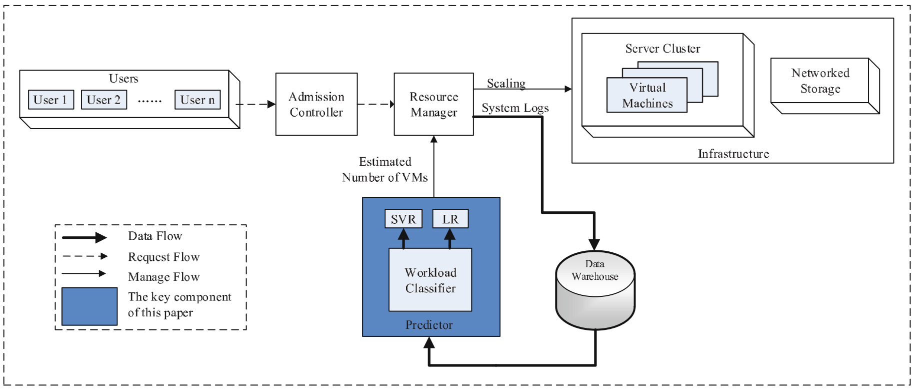

Fig. 1. Adaptive management architecture for workload prediction in service clouds

request is scaled in response to change infrastructures at fixed interval. The fixed interval is denoted as reconfiguration interval. The Admission Controller module determines the user's access. The Resource Manager module adjusts the allocation of resources according to the change of user requests dynamically. All of the service workload logs are transferred to workloads database, named as Data Warehouse. The historical workload information is used to distinguish workload type. It is also used to train and test the prediction model.

The Predictor module is the key module of this paper, it is highlighted in the figure. Once a new service is added, or an old service is down, the Workload Classifier is triggered. Then service workloads could be divided into different data types according to the change pattern of workload. Different prediction method is allocated to appropriate workload based on the optimal solution provided by Workload Classifier. The output of the Predictor is the total workloads of the platform. The total predicting workloads are estimated as the number of VMs and the Resource Manager module takes the number as input. Then the Resource Manager makes the decision to scale up or down VMs for the next reconfiguration interval.

The Workload Classifier automatically allocates suitable prediction model according to the change pattern of each service. In service clouds, there are web service, blog service, audio service, video service, e-mail service and so on. We classify workloads of these services into fast time-scale data or slow time-scale data according to the speed of load change in form of time-series employing. The fast time-scale data is stochasticity and nonlinearity, in which the burstiness of arrivals was high. The slow time-scale data is similar to linearity, in which the peak-to-mean ratio was low. The major difference between fast time-scale data and slow time-scale data is the intensity and speed of change. Therefore, we consider workload average change rate in the workload classification model. The result of workload classification is affected by the behavior of each load recently. Compared with slow time-scale workload, the fast time-scale workload changes violent and its average change rate is larger. This classification well distinguishes the load from the perspective of the change rate.

A constant prediction model cannot cover two kinds of data sets which are with contrary change pattern. In this paper, we utilize linear regression (LR) model and support vector regression (SVR) model to predict the workload, because they are naturally efficient and effective in the forecasting paradigm. Especially SVR is very suitable for the prediction of small sample nonlinear data. Therefore, they are suitable to these change patterns in this paper.

Workload Classifier component is the key to realize adaptive classification prediction. More detailed description can be seen in Sect. 3.

3 Model Formulation and Solution

In this section, the workload category problem is transformed into task assignment one which has a good solution. A 0–1 mixed integer programming of minimum assignment model is established, and an online solution is given in this section.

3.1 The Workload

We consider a discrete-time model that there is a time interval which is evenly divided into "frames" $t \in \{1, \ldots, K\}$. In practice, the length of a frame could be 5–10 min. Given a set of time series data w_t, $t \in \{1, \ldots, K\}$, $t = 1$ is the first frame of the time series, w_t stands for the actual data in frame t.

3.2 The Workload Classification Problem

Supposed there are i kinds of services $A_i(i = 1, \cdots, n)$ in service clouds. There are j kinds of prediction models $M_j(j = 1, 2)$ for A_i, which are fit for forecasting workload according to the characteristic of each kind of application workload provided by service clouds. Assign M_j to A_i, the predicting time is denoted by t_{ij}, the predicting error is denoted by ε_{ij} $(i = 1, \cdots, n, j = 1, 2)$. Here, ε_{ij} expresses Mean Absolute Percentage Error (MAPE) of a service workload i by model j at the frame t, defined as $\varepsilon_{ij} = \frac{1}{n} \sum_{s=1}^{n} \left| {}^{(\hat{w}_s - w_s)}\!/_{w_s} \right|$, where w_s is the actual output, and $\hat{w}_s$ is the prediction output, s is the observing time point, n is the data number of observation dataset at frame t. Cumulative Absolute Error at frame t denoted by $c\varepsilon_t$, and expresses the sum of MAPE of all services loads in service clouds, defined as $c\varepsilon_t = \sum_{i=1}^{n} \varepsilon_{t_i}$.

In order to achieve the minimum $c\varepsilon_t$ and satisfy the time requirement meanwhile, it is the key to improve the platform prediction performance that assigns suitable prediction models to each service in service clouds. While the workload feature is hard to extract and dynamic changing threshold is difficult to determine, we transform that problem into an assignment problem by establishing the optimization model.

Supposed x_{ij} expressed as distributing prediction model M_j to service A_i.

$$x_{ij} = \begin{cases} 0 & A_i \textit{ is not assigned prediction model } j \\ 1 & A_i \textit{ is assigned prediction model } j \end{cases} \tag{1}$$

When a new service is adding, or an old service is down at frame t, we need to reorganize the service type and reassign the prediction model. Then we need to determine x_{ij}.

3.3 The Workload Classification Optimization

Given the workload classification component above, the platform has one control decision vector x_{ij}, i.e. the allocation of prediction models when service number is changing at each frame. Given the limited error vector and time vector for A_i that satisfy the SLA requirements are $\varepsilon = (\varepsilon_1, \cdots \varepsilon_i \cdots, \varepsilon_n)$ and $T = (T_1, \cdots, T_i, \cdots, T_n)$. The goal of the service clouds is to determine x_{ij} to minimize the predicting error during [0, K]:

$$\begin{aligned} min \quad & \sum_{i=1}^{n}\sum_{j=1}^{2} \varepsilon_{ij} x_{ij} \\ s.t. \quad & 0 < t_{ij} < T_i \quad \mathrm{i} = 1, \cdots, \mathrm{n}, \quad \mathrm{j} = 1, 2 \\ & 0 < \varepsilon_{ij} < E_i \quad \mathrm{i} = 1, \cdots, \mathrm{n}, \ \mathrm{j} = 1, 2 \\ & \sum_{i=1}^{n} x_{ij} = 1 \qquad \mathrm{j} = 1, 2 \\ & x_{ij} \in \{0, 1\}^n \end{aligned} \tag{2}$$

where:

T_i is the upper limit of predicting time for A_i. It equals to configuration time - VM restart time - service deployment time.

ε_{ij} is the predicting error of A_i under current predicting model scheme j, E_i is the maximum error limit of A_i.

This model generalizes the service cloud optimization problem by accounting the total error of the platform. However, there is an issue in the model: the object of minimum platform error leads to the results that the optimal solution tends to be fast time-scale type. If it is assigned as fast data, it will use SVR model to forecast workload. Compared with LR model, SVR model has small predicting error and long predicting time. Predicting time of platform at frame t defined as the maximum predicting time of each service used in frame t. This would lead to more fast items and a long predicting time in the optimal solution for the workload classification.

With regard to this issue, we consider time factor, instead of only considering minimum platform error in the model. There is a significant difference in the average change rate of workload between fast time-scale and slow time-scale. The fast time-scale workload changes violent and the average change rate of workload is larger. Defined workload average change rate as $\overline{\omega} = \sum_{j=1}^{n} (y_{j+1} - y_j)/N$. There is obviously difference between fast time-scale and slow time-scale in term of $\overline{\omega}$. The $\overline{\omega}$ value of fast type is greater than that of slow type. In a classification result, the more fast type number, the larger sum of l_2-norm for $\overline{\omega}$. Therefore, to deal with the above issue, the Eq. (2) can be more balanced by introducing the sum of l_2-norm for $\overline{\omega}$ to tradeoff the overall prediction performance including prediction accuracy and time. Constant λ trades off a priori knowledge as throughout on the influence degree of the model. It needs to be determined before solving the optimization problem. Then the problem (2) may be written as Eq. (3). The influence of λ value on the prediction results will be discussed in Sect. 4.4.

$$\begin{aligned} min \quad & \sum_{i=1}^{n}\sum_{j=1}^{2} \varepsilon_{ij} x_{ij} + \lambda \sum_{i=1}^{n}\sum_{j=1}^{2} \|\overline{\omega_i} \cdot x_{ij}\|_2 \\ s.t. \quad & 0 < t_{ij} < T_i \quad \mathrm{i} = 1, \cdots, \mathrm{n}, \ \mathrm{j} = 1, 2 \\ & 0 < \varepsilon_{ij} < E_i \quad \mathrm{i} = 1, \cdots, \mathrm{n}, \ \mathrm{j} = 1, 2 \\ & \sum_{i=1}^{n} x_{ij} = 1 \qquad \mathrm{j} = 1, 2 \\ & x_{ij} \in \{0, 1\}^n. \end{aligned} \tag{3}$$

3.4 The Optimal Solution

Equation (3) is a 0–1 integer programming, branch and bound method is a primary algorithm to solve this kind of problem. The classical branch and bound algorithm works for the low efficiency when the scale of the problem is bigger. The branch and sub-problem choice strategies are important factors of the algorithm efficiency [26,27]. We adopt a search approach to solve Eq. (3). In order to achieve the better time performance, we choose the minimum cost priority and feasibility pruning strategies in searching, pruning and bounding process in this section. There is a detailed description in Fig. 2.

Given integrate programming as shown in Eq. (3), S(P) is the feasible solution set of Eq. (3), $\overline{x}$ is the feasible solution, F_u is the upper bound of the optimal value. Programming (3) can be divided into sub-programmings, denoted by $(P_1), (P_2) \cdots (P_i)$, and each sub-programming has corresponding relaxation programming, denoted by $(\overline{P_1}), (\overline{P_2}), \cdots, (\overline{P_i})$, $S(\overline{P_i})$ is the feasible solution set of $\left(\overline{P_i}\right)$, $x^{(i)}$ is the optimal solution and f_i is the optimal value. *NF* is the subscript set of the detecting problem P_i.

All the children nodes are produced from the current node in branch process. Those nodes that are impossible to generate feasible solutions are abandon, and other children nodes are adding to activated nodes set *NF*. Then new expansion node is chosen from the activated nodes set. In branch process, the minimum $c\varepsilon_k$ is the strategy to choose the branching node. We select the appropriate subscript $k \in \{1, 2, \cdots, n\}$ according to Eq. (4).

$$k = \arg\min\{f(\lfloor x_j^0 \rfloor), f(\lceil x_j^0 \rceil) | x_j^0 \ \ is \ \ fraction\} \tag{4}$$

After k is determined, $x_k = 0$ or $x_k = 1$, and the original problem is divide into two sub-problems.

The boundary of objective function for the sub-problems is established in a bounding process. If a sub-problem value is outside the boundary, this sub-problem will be pruned. To solve the relaxed programming $\left(\overline{P_i}\right)$ can obtain the optimal solution $x^{(i)}$ and the optimal value f_i.

In step $i \geq 0$, the low boundary of the optimal value f_i for (P_i) can be obtained, denoted as:

$$LF_{Li} = \min\{f_i \,| i \in NF\}$$

In step $i \geq 0$, the upper boundary of the optimal value f_i for (P_i) can be obtained, denoted as:

$$UF_{Ui} = \min\{LF(f_i) \left| f_i \in S(\overline{P}) \right.\}$$

To repeat branching, bounding and pruning process until there is no node in *NF*, the best feasible solution is the optimal solution of the original problem as shown in Eq. (3).

Through the above discussion, the branch and bound algorithm for the prediction model assignment program solving is described as Fig. 2.

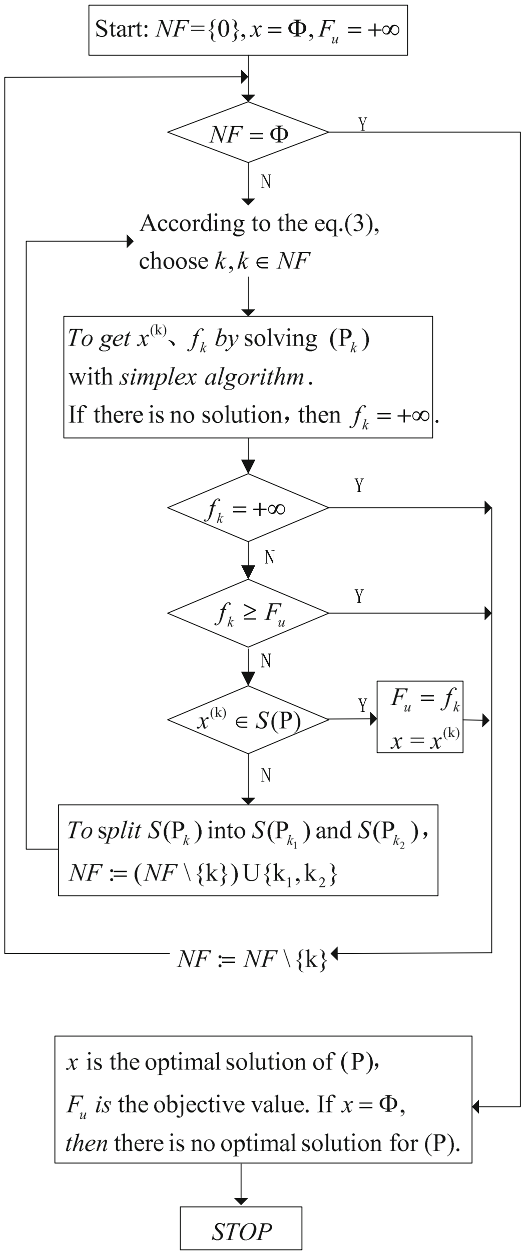
Start: $NF=\{0\}, x=\Phi, F_u=+\infty$
$NF=\Phi$
Y
N
According to the eq.(3), choose $k, k \in NF$
To get $x^{(k)}$、f_k by solving (P_k) with simplex algorithm. If there is no solution, then $f_k=+\infty$.
$f_k=+\infty$
$f_k \geq F_u$
$x^{(k)} \in S(P)$
$F_u=f_k$
$x=x^{(k)}$
To split $S(P_k)$ into $S(P_{k_1})$ and $S(P_{k_2})$, $NF:=(NF\setminus\{k\})\cup\{k_1,k_2\}$
$NF:=NF\setminus\{k\}$
x is the optimal solution of (P), F_u is the objective value. If $x=\Phi$, then there is no optimal solution for (P).
STOP

Fig. 2. Flow chart of branch and bound algorithm

4 Experimental Analysis

In order to illustrate the effectiveness of the proposed approach in service clouds, we conduct experiments with the data set built from some typical services of the application system developed by our lab. First, each service is allocated a suitable prediction model according to the optimal solution. With this assignment, we compare our method with single prediction methods in term of the platform cumulative absolute predicting error proposed in this paper. In order to obtain more accurate workload classification by optimal solution, we discuss the effect of parameter λ in optimization model on optimal solution followed.

4.1 Setup of the Experiment

There are some application systems in the experimental service clouds, such as Social Network Sites(SNS), Multimedia Conference System(Video System), Online Learning System(Learning System) and so on. Each of them is composed of more than one services. Table 1 lists some of the services of these systems. They are running in service clouds which are based on OpenStack, using two IBM x3650 servers as control nodes and three IBM x3650 servers as computing nodes.

Table 1. Services of the application system in service clouds

Application	Service
SNS	Web service
	Picture service
	Blog service
	Comment service
	E-mail service
Video system	Video service
	White board service
	Discuss service
	Cache service
Learning system	Web service
	White board service
	Video service
	Discuss service
	E-mail service

The data set of optimal classification experiment is built from the real trace of requests to service servers. We sample workloads of these services every 10 min time interval for 48 h in service clouds. The time interval is set by the time spending on booting a VM. The data set is composed of 30 groups workload traffic samples include 20 groups fast time-scale data plus 10 groups slow time-scale data to evaluate the workload classification optimization result. We would

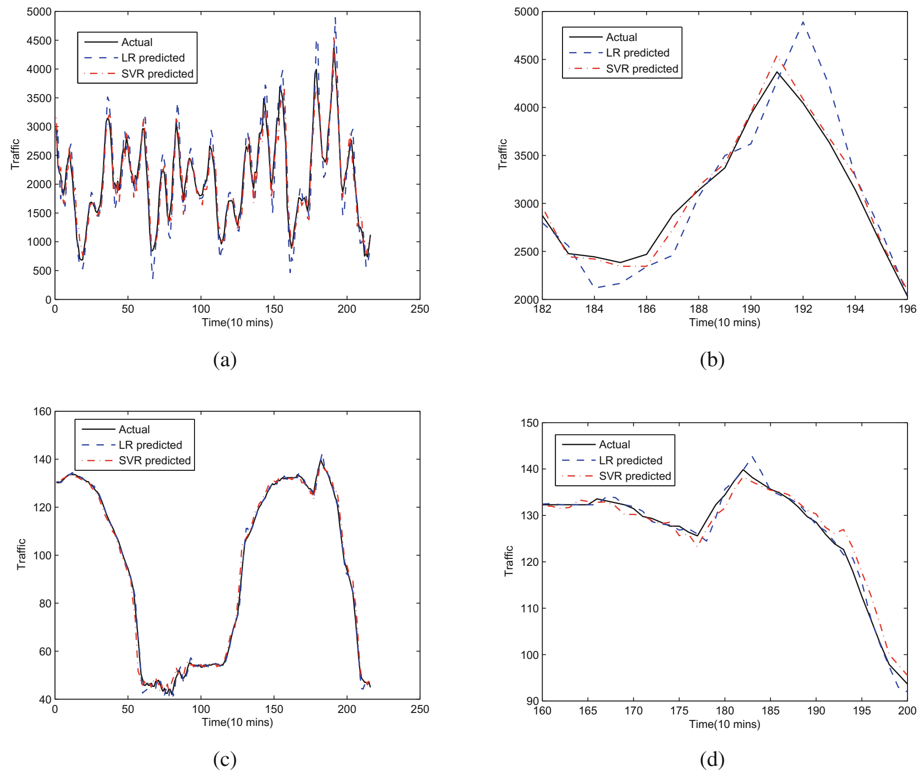

Fig. 3. Comparative prediction results on two kinds of workload: Web Service and Video Service. Figures on the left column illustrate comparative results using LR and SVR algorithm; figures on the right column enlarge the results for more clear illustration. (a) Web Service workload, (b) Enlarged part, (c) Video Service workload, (d) Enlarged part.

conduct our prediction experiment with two typical workloads to evaluate the proposed prediction approach. Workload variation pattern is a stochastic feature, closely related to the characteristics of service applications. For example, the traffic workload of web service is greatly influenced by a web content and change drastically with the arrivals burstiness. Video service mainly provides video transmission and decoding process for multimedia conference system. Its users mainly in the campus tend to use the video service in the daily work time. The traffic curve appears certain regularity and takes the shape of a flat curve. The ratio of peak to average is low and has cyclical change. Then we choose these two service workloads to verify the prediction result.

4.2 Workload Prediction Methods Analysis

The prediction models used in the prediction experiment are LR and SVR. LR [28] models the relationship between one or more input variables z and a dependent output variable y by using a linear equation. We take the following form.

$$y_t = \beta_1 + \beta_2 z_t \tag{5}$$

where y is the target variable, here is prediction workload. z is the explained variable, here is the time. t indexes the sample interval. The coefficients β_1, β_2 are determined by solving a linear regression equation based on previous workloads y_{t-1}, y_{t-2}, y_{t-3} and so on. β_1, β_2 change with different previous workloads, that is to say, this model can change with the workload trend. We use the Ordinary Least Squares to solve the Eq. (5).

Before using the SVR to predict time series, sample space reconstruction is required. We analyze our time series data, and give the reconstruction process as follow. Given time series x_t $(t = 1, 2, \cdots, T)$, from Takens phase space delay reconstructing theory, we use the m-dimensional vector, it is defined as

$$x(t-1) = [x(t-\tau), x(t-2\tau), \cdots, x(t-m\tau)] \tag{6}$$

where m is an embedding dimension and τ is delay constant. The prediction model can be described as $x(t) = g(x(t-1))$, g is a non-linear map. For our time series data, the input variables should be reconstructed using Eq. (6) of all data sets in the beginning, the values of embedding dimension and delay constant for the data set are set as follows: $m = 3$, $\tau = 1$.

We used the Gaussian RBF kernel and defined as $K(x_i, x_j) = exp(-\|x_i - x_j\|^2)$ $/(d\delta^2))$, where d is the dimensionality of x, $d = 2$, δ^2 is the variance of the kernel. We used the leave - one out cross validation approach to select model parameters, which chose the test and training sets randomly [29]. This method divided sample data into two parts, 70 % of the data is processed as a training set and 30 % of the data is processed as a test set. We used LIBSVM toolbox [30] to implement SVR algorithm. SVR parameter Settings are as follows: $m = 3$, $c = 100$, $g = 0.5$, $s = 3$, $p = 0.001$, $t = 2$.

Because of adopting cross validation, we use another half of the data set to test the prediction effect. As shown in Fig. 3, both SVR and LR make a good prediction effect for slow time-scale data. For fast time-scale data, the traffic changes drastically, more burstiness of arrivals come out near the sudden change. LR could not adapt the change and it produces a greater predictive error and phase deviation. SVR could adapt the dramatic change tendency well, because it conducts nonlinear kernel transforms. SVR has better prediction effect for fast time-scale data. But SVR needs more predicting time than LR. Therefore, we allocate the linear regression model for slow time-scale data as well as the support vector machine model for fast time-scale data in classified prediction approach.

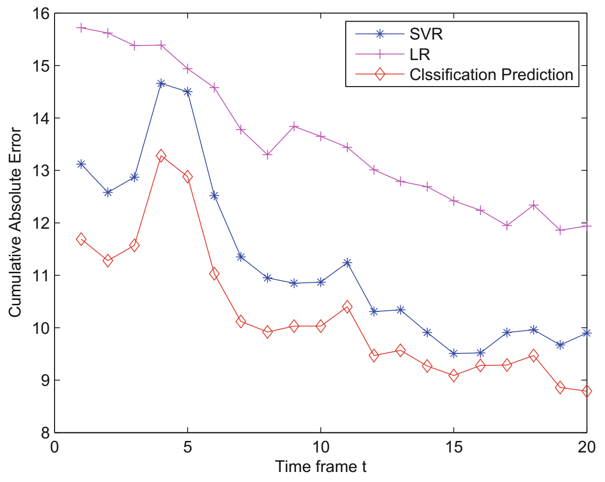

Fig. 4. Comparison of cumulative absolute error

4.3 Classified Prediction Effect Analysis

According to the above prediction model allocation scheme, we compare the prediction effect of classified prediction approach with SVR and LR prediction model. We evaluate the accuracy of the prediction approaches based on a number of metrics: Mean Absolute Percentage Error (MAPE), Cumulative Absolute Error at the frame t ($c\varepsilon_t$).

We select 20 points uniformly within prediction period and compute the MAPE and $c\varepsilon_t$. Then we compare the prediction effect of three approaches in terms of the mean value, the square error, and the mean square deviation of MAPE.

Figure 4 shows $c\varepsilon_t$ of service clouds at 20 prediction points. It illustrates that the MAPE of the classified prediction is minimum, SVR comes second, and LR is the highest.

In regard of the mean value, the square error and the mean square error at 20 prediction points with three methods, the statistical results are listed in Table 2. The mean $c\varepsilon_t$ of three prediction methods (e.g. classified prediction, SVR, LR) are 11.227, 13.544 and 10.266 respectively. Therefore, compared with SVR and LR, the classified prediction method reduces the platform cumulative absolute predicting error by 8.56 % and 24.20 % respectively. For different service load predictions, the cumulative error of a single prediction method is far bigger than the adaptive classification prediction method depending on the load characteristics.

Table 2 shows the $c\varepsilon_t$ statistical results of service clouds at 20 prediction points. The mean value of the classification prediction method is the smallest, LR is the highest. It illustrates that the $c\varepsilon_t$ of the classification prediction distributes more concentrated and forecasts more accurately. Variance and standard deviation of $c\varepsilon_t$ at each time are consistent. Variance and standard of classifica-

Table 2. Statistical results of cumulative absolute error

	Mean value	Square error	Mean square error
SVR	11.227	2.5997	1.6124
LR	13.544	1.7377	1.3182
Classified prediction	10.266	1.6702	1.2924

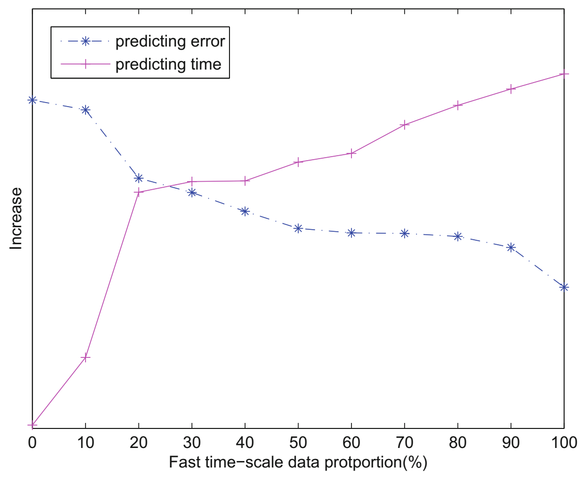

Fig. 5. Influence of fast time-scale data proportion on error and time.

tion prediction method are the smallest, and that of SVR are the biggest. This illustrates that the $c\varepsilon_t$ of the classification prediction method distribute more stably, compared with single prediction methods.

4.4 Influence of the Parameters λ on Workload Classification Optimization

As analysis in the previous section, the proportion of fast time-scale data in workload classification produces an effect on platform prediction accuracy and platform prediction time in classified predicting approach.

We measure the platform prediction error and platform prediction time by classified prediction approach, which the proportion of fast time-scale data increases from 0 to 100 % with 30 groups of load respectively. Figure 5 illustrates the influence of fast time-scale data proportion on accuracy and time. Obviously, with the increase of proportion, the platform prediction error decrease distinctly. Meanwhile, this would increase the platform prediction time. Therefore, finding the optimal ratio of fast time-scale workload is the key to improve the platform prediction result and keeping the balance of the forecasting accuracy and time performance to satisfy the response time of SLA.

In the workload classification optimization model, parameter λ plays the role to control the ratio of fast time-scale data in the solution of optimization. λ trades off a priori knowledge of workload on the influence of the model. It

Table 3. The relation between λ and optimal classification results

λ	Accuracy of classification	Average cumulative absolute error	Average predicting time (ms)
0	73.33 %	3.456	72.026
[0.11,1)	93.33 %	3.47	54.053
$+\infty$	33.33 %	8.028	0.0525

needs to be determined before solving the optimization problem. Thus the value of parameter λ on the optimal solution and predicting accuracy performance is studied in this section.

Table 3 illustrates the effect of λ. Here, accuracy of classification is defined as the correct classification number divided by the total number of load. Average cumulative absolute error equals to $c\varepsilon_t$ divided by the number of services. Average predicting time equals to the sum of predicting time divided by the number of services.

From Table 3, we analyze the effect of λ range where global optimal solution can be found.

(1) $\lambda = 0$ that is only the error term, no regular item to play a role. In optimal solution X, there are 26 workloads identified as fast time-scaled data, 4 workloads identified as slow time-scaled data. Classification accuracy is 73.33 %.
(2) As λ value is gradually increasing from 0.11 to 1, regular item plays more important role. In optimal solution X, there are 22 workloads identified as fast time-scaled data, 8 workloads identified as slow time-scaled data. 2 slow time- scaled workloads identified as fast time-scaled data on account of severe changing in the test interval. Classification accuracy is 93.33 %. Compared with (1), workload predicting model scheme, average cumulative absolute error only increase 0.4 % but average predicting time reduce 33.25 % in this scheme.
(3) $\lambda = +\infty$ that is regular item to play an important role, the error term nearly play no function. In optimal solution X, 30 workloads are identified as slow time-scaled data. Because in this optimal model, the objective is only time performance and the predicting time of slow time-scaled is much better than fast time-scaled. Classification accuracy is only 33.33 %. The model with no information of workload average rate of change leads to poor performance.

5 Related Work

At present, the approaches for workload or resource prediction in cloud can be classified in two categories according to the feature of a prediction method.

The first category prediction methods employed classical prediction models by studying the application characteristic in the cloud platform. To satisfy upcoming resource demands, Islam et al. [4] used Neural Network (NN) and Linear Regression (LR) to get new prediction-based resource measurement and provisioning strategies. Markov method is also used to predict workload. Studies [5–7] show that web and data center workloads tend to present behavior that can be effectively captured by time series-based models. Regard workload as time series, the Autoregressive Integrated Moving Average (ARIMA) model was applied to estimate the future need of applications. Roy et al. [8] utilized linear prediction method such as the exponential moving average, a second order autoregressive moving average and moving average method, to forecast the time series workloads. Sapankevych et al. [9] provided a survey of time series prediction applications using support vector machines (SVM) approach. It points out that the motivation for using SVMs is the ability of this methodology to accurately forecast time series data when the underlying system processes are typically nonlinear, nonstationary and not defined a priori knowledge. SVMs have also been proven to outperform other non-linear techniques including neural-network based non-linear prediction techniques such as multi-layer perceptrons. The work of [10] shows SVM provides the best prediction model among SVM, NN and LR with the SLA metrics for Response Time and Throughput. John et al. [11] analyzed the cloud workloads and used Markov modeling and Bayesian modeling to predict workload. Mathias et al. [12] developed a Markov framework to model the capacity variability of a service cluster for VM provisioning.

The second category prediction methods tended to propose a new prediction approach according to the feature of workload. They usually adopted the single prediction strategy at the same time to predict all the services resources. As typical ones, Caron et al. [13] proposed a Pattern Match algorithm to predict workload taking advantage of the self-similar feature. The main idea is to find out the historical load data similar to the load in current stage by using the feature of web traffic self-similar. Meanwhile, Gong et al. [14] presented a predictive elastic resource scaling scheme named as PRESS, which monitored CPU usage of VM, and derived signature-driven resource demand prediction by employing a Fast Fourier Transform. For applications with no repeating patterns, a discrete-time Markov chain based method was proposed to obtain state-driven resource demand prediction. Ghorbani et al. [15] introduced a fractal operator to account for the time-varying fractal and bursty properties of the cloud workloads.

However, these methods mentioned above usually adopt the single prediction strategy, that is, at the same time using the same method to predict all the services resources. Although many works were dedicated to resource prediction in clouds, there are few researches related to the application scenarios in which a variety of businesses run on the cloud platform at the same time. Considering the influence of cross correlation between servers' workloads, Khan et al. [16] analyzed the interaction between these workloads from the perspective of multiple time series, and then used hidden Markov chain model to predict the workload. Zhang et al. [17] designed a workload factoring service for proactive workload management. It segregated flash crowd workload from base workload with a data item detection algorithm.

Finally, Kupferman et al. [18] recommended a set of scoring metrics to measure the effectiveness and efficiency of dynamic scaling algorithms in terms of availability and cost. Copil et al. [31] presented a framework for estimating and evaluating cloud service elasticity behavior to estimate the expected elasticity behavior in time. It advise elasticity controllers about cloud service behavior to improve cloud service elasticity. Islam et al. [4] discussed a set of predicting metrics to evaluate the accuracy of the prediction algorithms, including Mean Absolute Percentage Error(MAPE), PRED(25), Root Mean Squared Error(RMSE) and R^2 Prediction Accuracy.

6 Conclusions and Future Work

To facilitate auto scaling resource management in service clouds, one of the crucial technologies is to predict resources in advance. In service clouds, it is difficult to predict accuracy due to the variation pattern of workloads.

In order to solve the above problem, we have presented an adaptive workload prediction approach in this paper. The approach categorizes the workloads and assigns different prediction models according to the speed of workload change. The key idea is that we formulate an optimization problem to maximize prediction precision, then convert workload classification into a 0–1 programming problem. It is solved quickly by employing an improved branch and bound algorithm.

We evaluated its accuracy using real traces from some typical services of the application system developed by our lab. We also analyzed the parameters value and the influence of the fast time-scale data on workload classification optimization. The experiment results demonstrate that our approach predicts more accurately in terms of the platform cumulative absolute predicting error. Moreover, the predicting error is well-distributed.

Our approach is also generic, and it can be used well in most service-cloud scenarios. In future, we plan to implement and evaluate our classification prediction approach for a wide variety of workloads by integrating the existing service platforms of our laboratory. We also intend to introduce more prediction models for these workload patterns. We need to modify the optimal model accordingly. This strategy will facilitate the prediction framework to make business-level SLAs constraint (such as, response time, accuracy performance and cost etc.) for adaptive and optimal resource provisioning in the cloud.

Acknowledgement. The work is supported by National Key Technology Research and Development Program of China (Grant No. 2012BAH94F02); National Natural Science Foundation of China (Grant No. 61132001); National High-tech R&D Program of China (863 Program) under Grant No. 2013AA102301; Project of New Generation Broad band Wireless Network (Grant No. 2014ZX03006003);The National Grand Fundamental Research 973 Program of China (Grant No. 2011CB302506).

References

1. Amazon elastic compute cloud. http://aws.amazon.com/ec2/

2. Google app engine. http://developers.google.com/appengine/
3. Jingqi, Y., Chuanchang, L., Yanlei, S., et al.: A cost-aware auto-scaling approach using the workload prediction in service clouds. Inf. Syst. Front. **16**(1), 7–18 (2014)
4. Islam, S., Keung, J., Lee, K., et al.: Empirical prediction models for adaptive resource provisioning in the cloud. Future Gener. Comput. Syst. **28**(1), 155–162 (2012)
5. Calheiros, R.N., Masoumi, E., et al.: Workload prediction using ARIMA model and its impact on cloud applications' QoS. IEEE Trans. Cloud Comput. **2**(8), 1–11 (2014)
6. Tran, V.G., Debusschere, V., Bacha, S.: Hourly server workload forecasting up to 168 hours ahead using seasonal ARIMA model. In: Proceedings of the 13th International Conference on Industrial Technology (ICIT 2012), pp. 1127–1131 (2012)
7. Jiang, Y., Perng, C.S., Li, T., et al.: Cloud analytics for capacity planning and instant vm provisioning. IEEE Trans. Netw. Serv. Manage. **10**(3), 312–325 (2013)
8. Roy, N., Dubey, A., Gokhale, A.: Efficient autoscaling in the cloud using predictive models for workload forecasting. In: Proceedings of the 2011 IEEE 4th International Conference on Cloud Computing, pp. 500–507, IEEE (2011)
9. Sapankevych, N.I., Sankar, R.: Time series prediction using support vector machines: a survey. IEEE Comput. Intell. Mag. **4**(2), 24–38 (2009)
10. Bankole, A.A., Ajila, S.A.: Cloud client prediction models for cloud resource provisioning in a multitier web application environment. In: Proceedings of 2013 IEEE 7th International Symposium on Service Oriented System Engineering (SOSE), pp. 156–161, IEEE (2013)
11. Panneerselvam, J., Liu, L., Antonopoulos, N., et al.: Workload analysis for the scope of user demand prediction model evaluations in cloud environments. In: 2014 IEEE/ACM 7th International Conference on Utility and Cloud Computing (UCC), pp. 883–889 (2014)
12. Björkqvist, M., Spicuglia, S., Chen, L., Binder, W.: QoS-aware service VM provisioning in clouds: experiences, models, and cost analysis. In: Basu, S., Pautasso, C., Zhang, L., Fu, X. (eds.) ICSOC 2013. LNCS, vol. 8274, pp. 69–83. Springer, Heidelberg (2013)
13. Caron, E., Desprez, F., Muresan, A.: Forecasting for Cloud computing on-demand resources based on pattern matching. Technical report, INRIA, pp. 1–23 (2010)
14. Gong, Z., Gu, X., Wilkes, J.: Press: Predictive elastic resource scaling for cloud systems. In: 2010 International Conference on Network and Service Management (CNSM), pp. 9–16, IEEE (2010)
15. Ghorbani, M., Wang, Y., Xue, Y., et al.: Prediction and control of bursty cloud workloads: a fractal framework. In: Proceedings of the 2014 International Conference on Hardware/Software Codesign and System Synthesis, pp. 12–21, ACM (2014)
16. Khan, A., Yan, X., Tao, S., et al.: Workload characterization and prediction in the cloud: a multiple time series approach. In: Network Operations and Management Symposium (NOMS), 2012, pp. 1287–1294, IEEE (2012)
17. Zhang, H., Jiang, G., Yoshihira, K., et al.: Proactive workload management in hybrid cloud computing. IEEE Trans. Netw. Serv. Manage. **11**(1), 7–18 (2014)
18. Kupferman, J., Silverman, J., Jara, P., Browne, J.: Scaling into the cloud. Technical report, University of California, Santa Barbara; CS270 - Advanced Operating Systems (2009). http://cs.ucsb.edu/~jkupferman/docs/ScalingIntoTheClouds.pdf

19. Yin Jianwei, L., Xingjian, Z.X.: BURSE: a bursty and self-similar workload generator for cloud computing. IEEE Trans. Parallel Distrib. Syst. **26**(3), 668–680 (2015)
20. Eldin, A.A., Rezaie, A., Mehta, A., et al.: How will your workload look like in 6 years? Analyzing Wikimedia's workload. In: 2014 IEEE International Conference on Cloud Engineering (IC2E), pp. 349–354 (2014)
21. Wang, K., Lin, M., Ciucu, F., et al.: Characterizing the impact of the workload on the value of dynamic resizing in data centers. In: 2013 Proceedings IEEE International Conference on Computer Communications (INFOCOM), pp. 515–519 (2013)
22. Reiss, C., Tumanov, A., Ganger, G.R., et al.: Heterogeneity and dynamicity of clouds at scale: google trace analysis. In: Proceedings of the Third ACM Symposium on Cloud Computing, pp. 7–20, ACM (2012)
23. Di, S., Kondo, D., Cirne, W.: Characterization and comparison of cloud versus grid workloads. In: Proceedings of IEEE International Conference on Cluster Computing, pp. 230–238 (2012)
24. Liu, Z., Cho, S.: Characterizing machines and workloads on a google cluster. In: Proceedings of International Conference on Parallel Processing Workshops, pp. 397–403 (2012)
25. Jorgensen, M.: Experience with the accuracy of software maintenance task effort prediction models. IEEE Trans. Softw. Eng. **21**(8), 674–681 (1995)
26. Chan, D.Y., Ku, C.Y., Li, M.C.: A method to improve integer linear programming problem with branch-and-bound procedure. Appl. Math. Comput. **179**(2), 484–493 (2006)
27. Mingfang, N.: Li Qi.: a surrogate constraint bounding approach to mixed 0–1 linear programming problems. J. Syst. Sci. Math. Sci. **19**(3), 341–347 (1999)
28. Seber, G.A.F., Lee, A.J.: Linear Regression Analysis. Wiley, New York (2012)
29. Mao, W., Xu, J.: Cao Xizheng.: a fast and robust model selection algorithm for multi-input multi-output support vector machine. Neurocomputing **130**, 10–19 (2014)
30. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines. ACM Trans. Intell. Syst. Technol. (TIST) **2**(3), 1–27 (2011)
31. Copil, G., Trihinas, D., Truong, H.-L., Moldovan, D., Pallis, G., Dustdar, S., Dikaiakos, M.: ADVISE – a framework for evaluating cloud service elasticity behavior. In: Franch, X., Ghose, A.K., Lewis, G.A., Bhiri, S. (eds.) ICSOC 2014. LNCS, vol. 8831, pp. 275–290. Springer, Heidelberg (2014)