International Journal of Machine Learning and Cybernetics (2019) 10:3285–3300
https://doi.org/10.1007/s13042-019-01017-1

ORIGINAL ARTICLE

An ensemble multiscale wavelet-GARCH hybrid SVR algorithm for mobile cloud computing workload prediction

Saeed Sharifian[1] · Masoud Barati[1]

Received: 11 May 2018 / Accepted: 13 September 2019 / Published online: 20 September 2019

Abstract
Dynamic resource allocation and auto scalability are important aspects in mobile cloud computing environment. Predicting the cloud workload is a crucial task for dynamic resource allocation and auto scaling. Accuracy of workload prediction algorithm has significant impact on cloud quality of service and total cost of provided service. Since, existing prediction algorithms have competition for better accuracy and faster run time, in this paper we proposed a hybrid prediction algorithm to address both of these concerns. First we apply three level wavelet transform to decompose the workload time series into different resolution of time–frequency scales. An approximate and three details components. Second, we use support vector regression (SVR) for prediction of approximate and two low frequency detail components. The SVR parameters are tuned by a novel chaotic particle swarm optimization algorithm. Since the last detail component of time series has high frequency and is more likely to noise, we used generalized autoregressive conditional heteroskedasticity (GARCH) model to predict it. Finally, an ensemble method is applied to recompose these predicted samples from four multi scale predictions to achieve workload prediction for the next time step. The proposed method named wavelet decomposed 3 PSO optimized SVR plus GARCH (W3PSG). We evaluate the proposed W3PSG method with three different real cloud workload traces. Based on the results, the proposed method has relatively better prediction accuracy in comparison with competitive methods. According to mean absolute percentage error metric, in best case W3PSG method achieves 29.93%, 29.91%, and 24.53% of improvement in accuracy over three rival methods: GARCH, artificial neural network, and SVR respectively.

Keywords Workload prediction · Cloud computing · Multi-scale wavelet decomposition

1 Introduction

Improvements in smart mobile devices technology, cause to new kind of applications such as mobile augmented reality (MAR), voice recognition, object recognition and natural language processing (NLP) has emerged among mobile users; these applications require powerful computational resources. Mobile devices are developing constantly to meet the applications requirements, but they yet have some limitations in regards with their processing speed, memory, bandwidth and battery life time. It should be noted that these shortcomings are mostly because of mobile devices weight and size constraints [1]. On the other hand, users of mobile smart devices desire to use these applications on their laptops, smart phones, and wearable computers while moving in any place at any time [2]. Due to resource constraints in mobile devices, mobile users may face to some issues regarding low power batteries or computational power limitations. To solve these problems, MCC is presented [3].

MCC provisions cloud resources for use of mobile devices. So, these devices can overcome their weaknesses in bandwidth, security, low process power and energy. In this computing model, complex processes and computation expensive tasks of mobile applications are sent to cloud datacenters to provide resources like storage, memory, computing power and energy for them [4]. Mobile devices can connect to clouds through standard communication channels such as 3G and 4G mobile networks or WIFI access points and use the cloud services in a pay-as-use manner.

One of the most important features of MCC is its elasticity and scalability which are results of dynamic resource

✉ Saeed Sharifian
sharifian_s@aut.ac.ir

Masoud Barati
masoud_barati@aut.ac.ir

[1] Department of Electrical Engineering, Amirkabir University of Technology, Tehran 15914, Iran

allocation according to users' requirements. Dynamic resource allocation needs control and intelligent management of cloud resources because the number of requests varies at any moment. A lot of reports indicate that in a day, most of the machines in cloud are not being fully loaded and work under load [5]. In practice, it is observed that physical systems consume about 60% of full load energy when they are at idle mode [6, 7]. It should be noted that energy saving results in huge reduction of financial expenses. For instance, 3% reduction in annual energy consumption of Google clouds causes cost saving more than a million dollars [8]. In addition, turning the resources on and off consumes a relatively large amount of energy too [9]. Because during shutting down process of a system, its memory contents should be stored in a permanent storage such as storage area network (SAN) and after turning the system back on, that memory contents should be restored and loaded to the memory. Hence, dynamic resource allocation in accordance with user requirements is very beneficial since it allocates resources in a way that reduces number of idle machines and so results in costs reduction.

In general, resource allocation can be done in three ways: static, reactive and proactive. In static method, fixed resources are provisioned to users, regardless of any changes in workload. The problem is that in the static mode, sometimes there is over-provisioning or under-provisioning which results in waste of resources and increases in total cost of cloud management. If there is under-provisioning, some of the requests cannot be responded and will be dropped and it results in extra costs because of SLA violations. If resources are over-provisioned, some of the physical machines will remain idle and cause extra costs because of their intrinsic cooling and energy consuming [10] and if under-provisioned, some potential users may be missed. The other approach is using feedback in reactive mode. In this method, whenever the system encounters load increase, amount of allocated resources to system will increase and if load decreases extra resources will be released. However, given that resource allocation takes up 1–5 min [11], this method always gets behind the rush of requests. So, it is great that it is possible to predict number of future requests so the resources can be allocated accordingly in advance. This can be possible using the last approach which is proactive workload predicting. In this method, if workload is predicted, service provider resources will be increased accordingly in time, so users can receive service with low delay. If workload reduction is predicted, extra resources can be released so other applications can use them. In some cases, a combination of both reactive and proactive methods is used so resources are provisioned to users in the best possible way [12].

In mobile cloud environments number of requests may change dramatically since the mobile users' requests are composed of small tasks which may consume little time to be processed. The users are always moving and involved in their real life and may enter or exit mobile cloud environment at any time or place. Aggregation of past request as cloud workload in a specific time duration forms a time series and is used for predicting future values. Such a workload time series is very complex with high frequency of variation and volatility nature which makes its prediction a hard task. On the other hand, higher accuracy of workload prediction results to better resource allocation and better resource allocation improves QoS and reduces SLA violations caused by dropped requests and also decreases energy consumption. Hence workload prediction is an important task in MCC. Statistic approaches like autoregressive moving average (ARMA), GARCH, and artificial intelligence (AI) ones like ANN and SVR which are presented in previous works did not provide an acceptable accuracy for cloud workload prediction, an even more accurate algorithm in this subject is required. In this paper we proposed W3PSG algorithm for cloud workload prediction in MCC environments. The novelties of the proposed algorithm are as follows:

- Since the cloud workload is very complex with high frequency of variation and volatility nature, we used wavelet transform to decompose the time series into different time–frequency sub scales. Each sub scale has homogenous characteristic and less complex nature, so can be predicted more accurately.
- We propose a hybrid ensembles of SVR/GARCH predictor in each subscale and train it for prediction reconstructed components of workload in that subscale.in addition we propose CPSO algorithm to adjust SVR parameters and increase its prediction accuracy. Also, GARCH predictor is suggested for predicting high frequency and high volatile sub scales of the time series.
- The proposed W3PSG algorithm is used to predict three real cloud workload traces as baseline benchmarks. The results indicate that the proposed prediction algorithm outperforms rival algorithms especially in prediction accuracy.

Rest of the paper is organized as follows: In Sect. 2, related work presented. In Sect. 3, the architecture, problem formulation and theory of the proposed W3PSG algorithm is explained. In Sect. 4, simulation results and discussion are provided. Finally in Sect. 5 conclusion and suggested future works are explained.

2 Related work

Resource allocation in MCC environments attracted a lot of researches in recent years. It is known that there are so many factors that determine a resource allocation algorithm as a good one, like cloud provider's financial costs and users' QoS. For instance, authors in [13] have proposed a novel hybrid resource allocation algorithm in MCC environments for reducing offload time. They have benefitted two meta-heuristic algorithms for load balancing and both2 mobile users waiting time and servers' response time are reduces. In Ref. [14] authors provide an optimized resource allocation which only considers energy efficiency in MCC. The authors have developed a priority-based algorithm according to users' channel gain and energy consumption. All the above mentioned works do not follow the proactive resource allocation scheme, which we used in the proposed algorithm. In Ref. [15], an adaptive approach is proposed. The authors propose a resource allocation algorithm using learning automata technique and the results show that it outperforms other algorithms in term of QoS. Learning Automata is a lazy learning approach and its adaptive learning rate is too slow in order to pursue cloud workload variations. Hence it become inefficient in cloud workload prediction.

Recently, there has been a lot of valuable works done in the field of time series prediction. A classic and most famous models for predicting time series is ARMA. This model usually includes two sections named; autoregressive (AR) and moving average (MA). ARMA models time series as a static stochastic process. So, it has a good accuracy in stationary and less complex time series with linear dependency between samples [16]. Authors in [17] propose a runtime QoS prediction algorithm for cloud workload, which uses modified versions of ARMA. Simulation results confirm that using long history of QoS data in ARMA models can reduce prediction error. In Ref. [18] similar work presented and real traces are used for the simulation.

GARCH is used for time series prediction in recent works [19]. GARCH is similar to ARMA model with a difference that in GARCH model, variance of predicted term is not constant but is a function of values and variances of previous prediction error terms. This model acts precisely upon data with variable conditional variance and highly volatile nature. This model also acts very well in predicting noisy parts of time series. In [20], authors have proposed a novel hybrid approach which is a combination of adaptive neuro-fuzzy inference system (ANFIS) and GARCH to predict the network packet flow traffics. In this paper we extend this idea and use GARCH for predicting high frequency components of cloud workload time series.

Previous works mentioned that the main problem of time series prediction as high degree of nonlinearity which cannot be modeled by stochastic approaches. Hence suggested to use machine learning approaches instead. One of the main stream line machine learning methods that has attracted a lot of attentions for predicting time series is Neural Network (NN) [21, 22]. Actually, NN's ability to models very complex nonlinear relationships between inputs and output has made it a proper choice for prediction applications. NN consists of several layers: input layer, hidden layer, and output layer and there are several neurons in each layer. Number of neurons and hidden layers are structural features of a NN and can be determined based on the type of problem [23]. NN have so many parameters that should be adjusted. Hence the computational cost for training them is high. In Ref. [24] authors use NN coupled with singular spectrum analysis in order to predict rainfall-runoff modeling. In Ref. [25] authors propose an enhanced extreme learning machine (ELM) neural network model for river flow forecasting.

Another popular machine learning approach for prediction is SVR. The idea of SVR is based on calculating a linear regression function in a multi-dimensional feature space resulted by nonlinear mapping of input vector to feature space. SVR has less parameters in comparison to NN. Also, with regards to nonlinear mapping and operational risk function that is defined for it, if the parameters are set properly, SVR has more accuracy than NN. We have suggested a new meta-heuristic optimization method for tuning its parameters to improve its performance. In Ref. [26], a novel hybrid algorithm based on genetic algorithm is presented to improve load predicting using SVR method. The simulation results reveal that the proposed method excels ARIMA in prediction accuracy. A short-term workload prediction using SVR is proposed in [27]. In this paper the authors have tried to benefit memetic algorithm and PSO to improve SVR. In Ref. [28] authors use firefly algorithm to tune SVR parameters to predict evaporation in northern Iran.

In recent years, hybrid prediction algorithms have attracted much attention. Time series which are challenging problems are usually very complex, highly none linear and volatile. Hence, a single method could not perform well. Unlike the previous mentioned works, we addressed a hybrid prediction algorithm for the case of cloud workload prediction. We used wavelet transform to decompose workload time series and then each sub-scale component can be predicted regarding its characteristics. With a close examination of cloud workload time series characteristics, we have suggested using GARCH model for the noisy sub-scale and SVR for the rest of sub scales. The details of the proposed prediction algorithm is provided in Sect. 3.

3 The proposed method

In this section we present the proposed method. First the MCC architecture is presented. Second, we explain the dynamic resource allocation mechanism in this environment. Finally, we describe the proposed workload prediction algorithms and explain related sub modules.

3.1 MCC architecture

Generally, MCC means executing computing parts of resource-hungry applications (for example "Google translation") of smart mobile devices on powerful servers as depicted in Fig. 1. In Fig. 1 a mobile device acts as a thin client and connects to servers of datacenters via 3G, Wi-Fi or cloudlets. As presented in Fig. 1, cloudlet is a rack of computers which is located near mobile devices (for example in base station) and receives users' requests, processes them or sends them to a remote cloud [29]. In comparison to public clouds, cloudlets have limited CPU and memory resources but provides less expensive services. Users in public places can send their requests to clouds (huge datacenters) via cloudlets. This method helps mobile devices to overcome their bandwidth limitation and its resulting delay. In this paper we used such architecture, because it is a popular architecture in recent years.

3.2 Dynamic resource allocation in MCC

In Fig. 2 dynamic resource allocation for MCC environment is presented. In this architecture, mobile users send their requests to the remote cloud through internet or via cloudlets and the cloud provider provisions resources VM (virtual machines) based on users' requests. There is a contract between each user and cloud provider called Service Level Agreement (SLA) that determines user's maximum acceptable waiting time to receive a service from cloud. For each service which response time exceeds SLA, the cloud provider will drop the request and pay penalty to user. In this regard, the main problem of cloud provider is that when a request arrives, a VM should be provisioned immediately for service in order to reduce the user service time. Unfortunately starting a virtual machine typically takes 1–5 min which is beyond the user's acceptable service time. The solution is to predict number of future requests in advance and provide the VM, so the VMs become ready when the new requests arrive. As it is illustrated in Fig. 2, the monitoring unit always saves the number of requests in any time to produce request time series. Monitoring unit sends current number of requests to the resource controller. Request time series is fed into the predicting unit so it can forecast future number of requests. The proposed prediction method which is described in Sect. 3.3 is used in this module. The output of the predicting unit is sent to the resource controller module.

The main task of resource controller is deciding whether to add or remove VM resources to the cloud resource pool based on its input information which is current number of tasks, predicted number of requests and current state of resources in the cloud resource pool. After predicting the number of upcoming requests, the amount of needed resources should be adopted by the Resource controller, so cloud resource pool is increased or decreased by ΔCPU and ΔMemory according to VM resources in a way that it matches with the upcoming requests requirements. It is task of the resource allocator unit. By using this provisioning mechanism, SLA can be guaranteed and the cloud provider can always be ready to receive upcoming requests.

3.3 Proposed workload prediction algorithms

In MCC environments, user requests are submitted by very diverse mobile applications, hence the multiplexed request time series in cloud is always nonlinear, highly variable and very stochastic. Predicting such a stochastic time series is a tricky issue because of its high frequency components. As a solution, we proposed three prediction algorithms which are gradually improved one after the other and finally reached to the proposed W3PSG algorithm. First of all; workload time series is decomposed to different time–frequency scales, using wavelet transform and then properly predicted using combinations of SVR and GARCH algorithms in an ensemble manner. The main difference between three proposed algorithms is in selecting which one of SVR or GARCH algorithms is selected to predict each scale of time–frequency transformed time series. Since GARCH algorithm have more stochastic nature; it better estimate time series

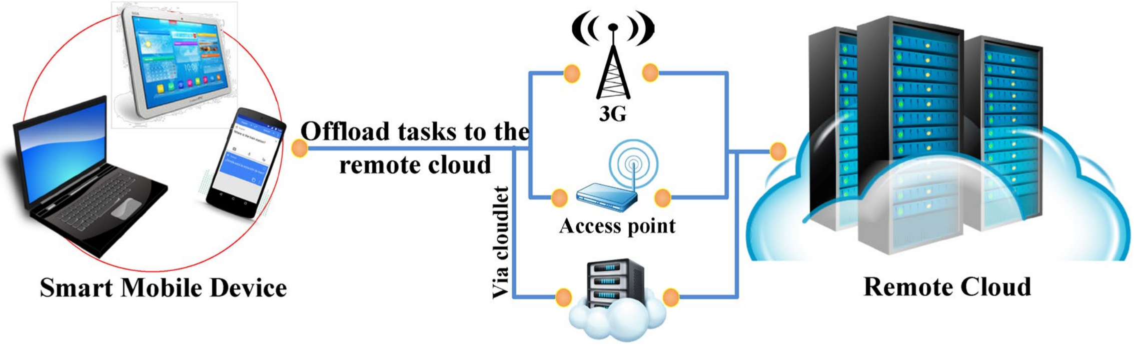


Fig. 1 Architecture of MCC

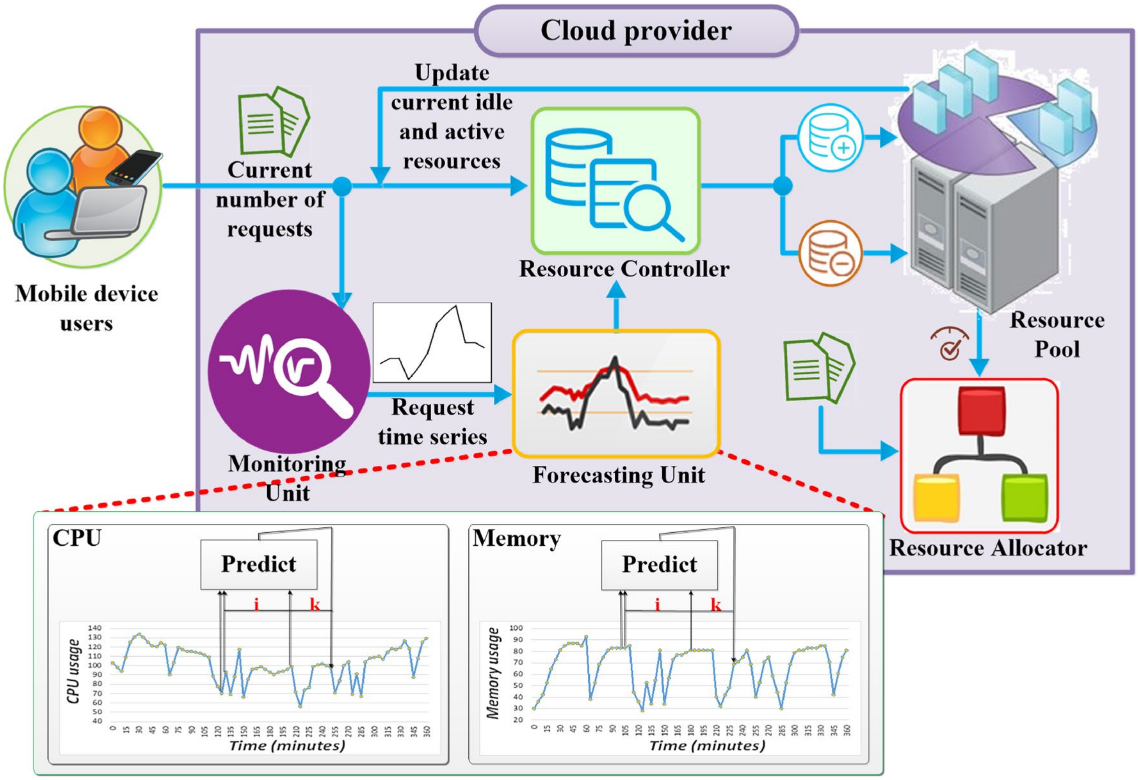


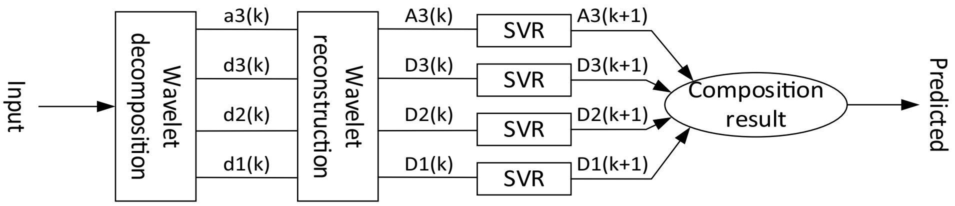
Fig. 2 Dynamic resource allocation in MCC

with high degree of Volatility. Hence it should be used for higher frequency of time series components which is related to details in wavelet transform. On the other hand SVR is suitable to predict time series with high degree of nonlinearity, so we used it to predict lower frequency components of time series which is related to the approximate in wavelet transformation. To verify the proposed hypothesis we develop three version of the prediction algorithm which are described in more details as follows".

As illustrated in Fig. 3 after three level wave let decomposition of input cloud workload time series; the transformed wavelet time series for each time–frequency scale are reconstructed to have the same number of samples and equal length to the original workload time series. The results are three different details components named as D1(k), D2(k) and D3(k) respectively. Where k is the kth sample of time series and D1(k) has the highest frequency component among details and also an approximate component named as A3(k) which has low frequency components. The details of wavelet transform and reconstruction is provided in Sect. 3.4.

For prediction in first algorithm; we used SVR for all the wavelet components as depicted in Fig. 3. The details of SVR algorithm is provided in Sect. 3.5. In order to improve the performance of the SVR model for each component we used chaotic PSO algorithm to find SVR parameters which is described in details in Sect. 3.5.1. Chaotic number generator is used instead of typical random number generation for better and faster convergence of PSO algorithm. Finally, after prediction of each sub-scale, their results are added together in order to obtain future value of workload time series. Since in this algorithm four sub-scales used PSO optimized SVR (PSVR) for prediction, we have named it wavelet 4 PSVR (W4PS). Each of the sub scale time series has its specific statistics and signal characteristics, so each of them is trained via a separate PSVR.

Fig. 3 Block diagram of W4PS



In the next step of improving the algorithm we decide to use GARCH method for three detail sub scales and a PSVR for approximate sub scale as depicted in Fig. 4. The idea behind this improvement is the ability of GARCH method to model and predict time series with high frequency, noisy and high volatility nature which is the same specification as details sub scales. Hence we have named this algorithm wavelet one stage PSVR plus 3 stage GARCH (WPS3G). This algorithm enables us to benefit from both methods advantages simultaneously. So, WPS3G outperforms W4PS method since it has benefited GARCH method superiority for predicting noisy parts of the signal. This feature of WPS3G method causes model accuracy to improve but increases execution time of algorithm.

The main problem of WPS3G method is its high computational cost which is related to GARCH algorithm. To solve this problem we proposed the third algorithm as depicted in Fig. 5, which is named wavelet 3 stage PSVR plus a GARCH (W3PSG). Like the previous algorithm, it is a combination of PSVR and GARCH, however, in this method, only the sub-scale with highest frequency and noisy nature (D1) is predicted by GARCH method and the rest of the sub-scales (D2, D3 and A3) are predicted by PSVR method which has a better trade of accuracy and computation cost. In addition to computation cost reduction, this method has higher total prediction accuracy as shown in simulation results which is described in more details in Sect. 4.

3.4 Multi scale decomposition using wavelet transform

Cloud workload time series usually have much noise and fluctuations due to many factors. Wavelet transform is a multi-resolution time–frequency analysis that decomposes input workload time-series to several sub-scales based on mother wavelet type. Preprocessing input time series in this way causes reduction of disordered and irregular characteristics of input time series; this results to simplification of modeling each sub-scales [30]. Nowadays, it can be applied to a lot of applications such as compression, regeneration, simplification and noise reduction [31, 32]. Wavelet transform has the ability of decomposition of time series to a few sub-scales with different frequency bandwidth just like a filter bank in signal processing. Wavelet transform can decompose an input signal to an approximation time series (overall shape of the signal) and a few high frequency time series (noise and details of the signal).

In this paper, we have used wavelet transform for increasing accuracy of workload prediction in MCC. Since workload in MCC environments is very noisy and the cloud workload time series is stochastic with high frequency components, wavelet transform is used to divide the input time series to several time–frequency sub scales. Each of those sub-scales is modeled using one of the prediction algorithms. Similar to Fourier transform, continues wavelet transform of a function is defined as aggregation of multiplying the function with the scaled wavelet function which is shifted over a time interval. Therefore, wavelet coefficients, C, can be written as in Eq. (1). Multiplication of each of these coefficients by its related scaled and shifted wavelet, determines its portion in construction of the main signal.

$$C(scale,\ \text{position}) = \int_{-\infty}^{+\infty} f(t)\psi(scale,\ \text{position})dt$$
$$C(a,b) = \int_{-\infty}^{+\infty} f(t)\frac{1}{\sqrt{a}}\psi\left(\frac{t-b}{a}\right)dt \tag{1}$$

It should be noted that a wavelet with large scale property incorporates in low frequencies and small scale ones incorporates in high frequencies. Any function that is used as a wavelet has a zero mean and its energy value equals one. In addition, to make sure that the wavelet-based transformed signal has the capability to be reconstructed in the

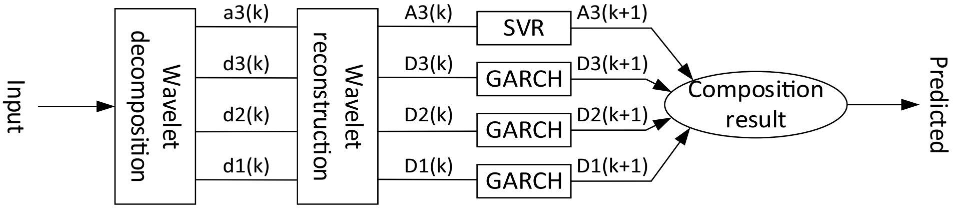


Fig. 4 Block diagram of WPS3G

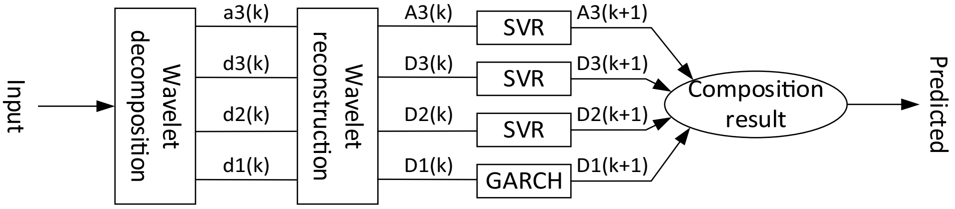


Fig. 5 Block diagram of W3PSG

time domain, the selected wavelet should meet the admission condition formulated as in Eq. (2).

$$0 < \int_0^{\infty} \frac{|\Psi(f)|^2}{f} df < \infty \tag{2}$$

Since the signal that we are mostly interested in analyzing them are usually discrete, wavelet transform discretization is inevitable. For simplifying working with wavelet transform, its discretization should be done in binary. Thus, scale and shift are integer exponents of two. Hence Eq. (3) can be obtained by substituting $a = 2^j$ and $b = ka$ in Eq. (1).

$$\begin{aligned} C(j,k) &= \sum_{n \in Z} f(n)\psi_{j,k}(n) \\ \psi_{j,k}(t) &= 2^{-\frac{j}{2}}\psi\left(2^{-j}t - k\right) \end{aligned} \tag{3}$$

Although discretized version of wavelet transform can be calculated by computer systems, but it is not really a discrete transform. Actually, the discretized version of wavelet transform is several wavelets which are taken as samples of a continuous wavelet transform. So, the information hidden in it is in vain and causes extra computational load. In order to decrease the computational load, we have used the discrete wavelet transform (DWT) as sub-band coding. Principles of DWT refer to a method called sub-band coding which is implemented using digital filters. In discrete mode, filters with different cutoff frequencies are used for analyzing signals in different scales. Different frequencies of signals are analyzed when they pass through high-pass and low-pass filters. In discrete mode, signal resolution is controlled by filter functions and their scale is changed via down sampling and up sampling procedure. DWT processing begins with passing the signal through a low-pass digital filter. Filter output equals convolution of input and filter impulse response. As the result, all frequency components which are bigger than half of the biggest existing frequency in the signal will be omitted. Since the biggest existing frequency in the signal equals π/2 rad, half of the components can be omitted. So, if the samples are omitted every other sample, signal length will be half without losing any information. Same procedure can be done using a high-pass digital filter.

Preforming this method, time resolution is halved and in return, frequency resolution will be doubled. This procedure can be applied again on the low-passed version of the signal and in every iteration, with halving time resolution of the previous signal, frequency resolution will be doubled. This idea is known as filter bank which is used for calculating DWT. It can be seen that output coefficients of the low-pass filter follow initial formation of the signal, so they are called Approximations. Also, output coefficients of the high-pass filter contain high frequency details of the signal so they are called Details. With the increase of number of transform iterations, the amount of details decreases. DWT using filter bank is shown in Fig. 6. In the left side of the figure, wavelet transform is presented and H_0 and H_1 are low-pass and high-pass filters respectively. Downward arrows indicate down-sampling. In the right side of the figure, reverse wavelet transform is shown in which an up-sample occurs at first and then low-pass and high-pass filters are applied.

In this paper, we have used three level filter banks in order to decompose the input time series to four sub-scales each with the same number of samples as the original input time series. We examine different levels of wavelet transform and 3 level achieves the best performance. Approximation part of the input time series (low frequency) is indicated as a3 which length is $\frac{1}{8}$ of the main signal because it was down-sampled three times. Details part of the signal (high frequency) are shown using d1, d2 and d3 which length are $\frac{1}{2}$, $\frac{1}{4}$ and $\frac{1}{8}$ of the input time series respectively. Since the sub-scale components obtained via filter bank method do not have equal lengths, an up-sampling procedure followed by a filter should be applied on them as illustrated in the right side of the Fig. 6, so all of sub-scale components became the same length as the original time series. By combining A3 (approximation) and D1, D2 and D3 (details), the input time series P can be rebuilt according to Eq. (4).

$$P = A3 + D3 + D2 + D1 \tag{4}$$

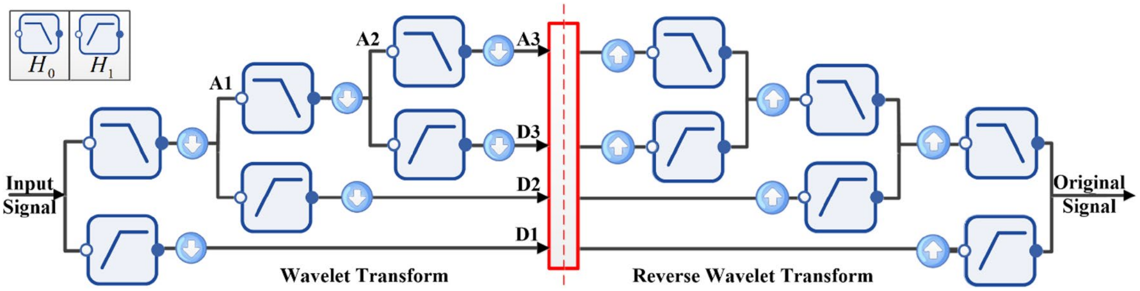


Fig. 6 Wavelet transform and reverse wavelet transform

For better clarification, Worldcup98 [33] time series which is used as baseline cloud workload is decomposed using wavelet transform (db4 wavelet) and then reconstructed by the Eq. (4). The result shown in Fig. 7 indicated that D1 and A3 represent high frequency and low frequency variation of the workload respectively.

3.5 Support vector regression (SVR)

SVR is an extension of binary classification of support vector machines (SVM) with a difference that outputs can take infinite values. SVR can be used in function estimation, curve fitting and time series prediction. In the following we used the notations in Table 1.

Table 1 Notations for SVR parameters

Symbol	Description
ti	The input vector
Yi	The prediction values
N	The total number of data set
w	Weight coefficients of the SVR function
b	Constant coefficient of the SVR function
ε	The value of epsilon in the insensitive loss function
σ	The value of sigma in the Gaussian kernel
C	The trade-off between the empirical risk and the model flatness
$\xi_i^+ \xi_i^-$	The distance from actual values to the corresponding boundary values of ε-tube
w^T	The optimal weight vector of the regression hyperplane
K(xi, xj)	The kernel function

In SVR inputs $x_i \in R^m$ and outputs $t_i \in R$ are both continuous variables. It is assumed that the SVR should be estimated in a way that $t_i \simeq y_i$ and y_i can have a linear model as $y_i = w^T x_i + b$ or any other nonlinear models. If assume a nonlinear model, a nonlinear mapping can transfer the input space to a linear space with more dimensions. The less the w norm is, the simpler model results and if it's become zero, y turn to a constant value. Estimation error of the model is acceptable if its value remains under ε. If our model meets this condition and all estimations are included in the above range, our model is acceptable. If the difference of any data with its estimation is more than ε, a penalty ξ will be considered for it. This penalty is called loss function L_ε as in Eq. (5).

$$L_\varepsilon = \begin{cases} 0 & \left|t_i - y_i\right| \le \varepsilon \\ \left|t_i - y_i\right| - \varepsilon & \text{otherwise} \end{cases} \tag{5}$$

To construct the SVR model we consider two objectives as formulated in Eq. (6): first empirical risk which is the average

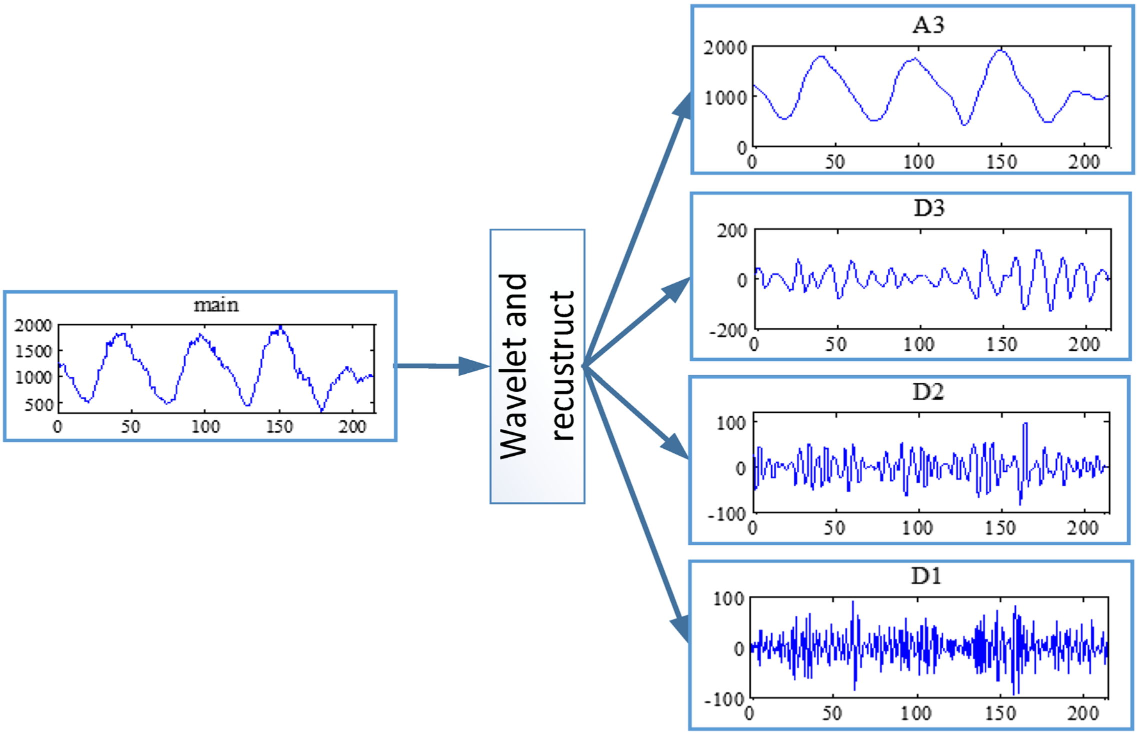


Fig. 7 Decomposition of Worldcup98 signal using wavelet transform

of penalty functions should be minimized. Second, the w norm should be minimized in order to simplify the model.

$$Min\left\{\text{Remp}=\frac{1}{N}\sum_{i=1}^{N}L_{\varepsilon}\left(t_i,y_i\right)\right\} \qquad Min\,\|w\|\rightarrow Min\frac{1}{2}w^{T}w \tag{6}$$

where model conditions can be written as (7). ξ_i^- and ξ_i^+ are violation costs.

$$\begin{aligned}-\varepsilon-\xi_i^- &\le \left|t_i-y_i\right|\le\varepsilon+\xi_i^+\\ \xi_i^- &\ge 0\\ \xi_i^+ &\ge 0\end{aligned} \tag{7}$$

Then with applying Karush–Kuhn–Tucker (KKT) conditions on Eq. (6) and performing quadratic function in the conjugate minimization problem, Eq. (6) can be solved and substituted with Eq. (8).

$$y=\sum_i\left(\alpha_i^+-\alpha_i^-\right)k\left(x_i,x_j\right)+b \tag{8}$$

In Eq. (8) we have used radial basis function (RBF) Kernel to map input data into a higher dimensional space which is shown in Eq. (9). RBF kernel is selected for its better accuracy due to flexible hyper plane decision boundary and fewer dimension space in compare to linear, polynomial kernels.

$$k\left(x_i,x_j\right)=\exp\left(-\frac{1}{2\sigma^2}\left\|x_i-x_j\right\|^2\right) \tag{9}$$

The problem can be considered as a neural network as shown in Fig. 8. In Fig. 8, workload prediction by SVR is illustrated.

SVR algorithm has three parameters (c, ε and σ). There isn't a closed form method to adjust these parameters and their desirable values could be obtained by searching or Meta heuristic optimization methods. Proper adjustment of these parameters has a direct effect on accuracy of SVR. These three parameters have features. Where C is called adjustment parameter and establishes a balance between minimizing error and minimizing complexity of the model. If we consider high value for C, the goal will only be minimizing empirical risk (cost function) and it will result to a complex model. Other parameter called ε and is defined in order to decrease noise and stabilize predictions [34]. It determines insensitive area of operational risk and also specifies the number of vectors used in the regression [35, 36]. A high value of ε causes the number of support vectors to be decreased and the regression function gets more flat and became simpler. The last parameter σ is RBF kernel parameter which is specifies the kernel structure. Hence three parameters of SVR model should be set with high accuracy. In order to adjust these three parameters, we have used PSO algorithm as described in the following section.

3.5.1 Chaotic particle swarm optimization algorithm

We select PSO algorithm to optimize SVR parameters due to its fast convergence and simple computation. We modified the baseline PSO algorithm to use chaotic random generator and named it CPSO. Usually simple random function generator is used in optimization algorithms for generating their first population, crossover and mutation procedures. In this paper, chaotic recursive function is used for generating random numbers due to its ability to help faster convergence of algorithm. Diversity and proper distribution as well as its simplicity have made chaotic sequence a proper choice for us to use it as a random function generator [37]. There

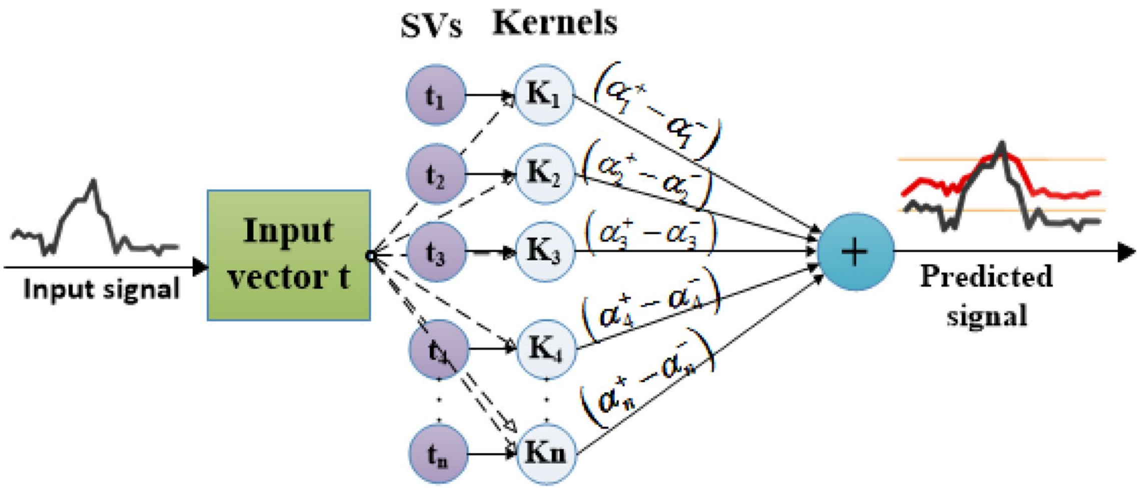

SVs
Kernels
t_1
t_2
t_3
t_4
t_n
K_1
K_2
K_3
K_4
Kn
$(\alpha_1^+-\alpha_1^-)$
$(\alpha_2^+-\alpha_2^-)$
$(\alpha_3^+-\alpha_3^-)$
$(\alpha_4^+-\alpha_4^-)$
$(\alpha_n^+-\alpha_n^-)$
+
Input signal
Input vector t
Predicted signal

Fig. 8 SVR model for cloud workload load prediction

are many chaotic sequence generators in the literatures. We achieves better results in our experiments by using the one which is in Eq. (10).

$$x^{(i+1)} = \mu x^{(i)}\left(1 - x^{(i)}\right) \quad x^{(i)} \in (0,1), \quad i = 0,1,2,\ldots \tag{10}$$

where $x^{(0)}$ is the initial value of the sequence (or its seed) and its value has effects on the sequence behavior; a little variation in its value changes the sequence behavior completely. μ is another parameter called Bifurcation that determines type of the sequence. For example, if μ is in range [2.8, 4] interval, the sequence will behave chaotically. We have chosen 4 for its value via trial and error. Optimization of SVR model parameters is done in offline learning procedure, and unlike neural network learning, SVR does not need to be learned online. In Fig. 9, flowchart of the proposed CPSO algorithm is illustrated. In the first step, optimization algorithm parameters are initialized. Particle range, particle velocity range, population and number of iterations are set to [0, 1], [−0.1, 0.1], 10 and 100 respectively as defined in [38, 39]. Then the first population is generated using a chaotic sequence. In each iteration, the search space gets explored in order to converge to the optimal solution.

We consider mean absolute percentage error (MAPE) as cost function and search space of C, ε and σ parameters to [0, 1000], [0, 1] and [0, 20] respectively [38].

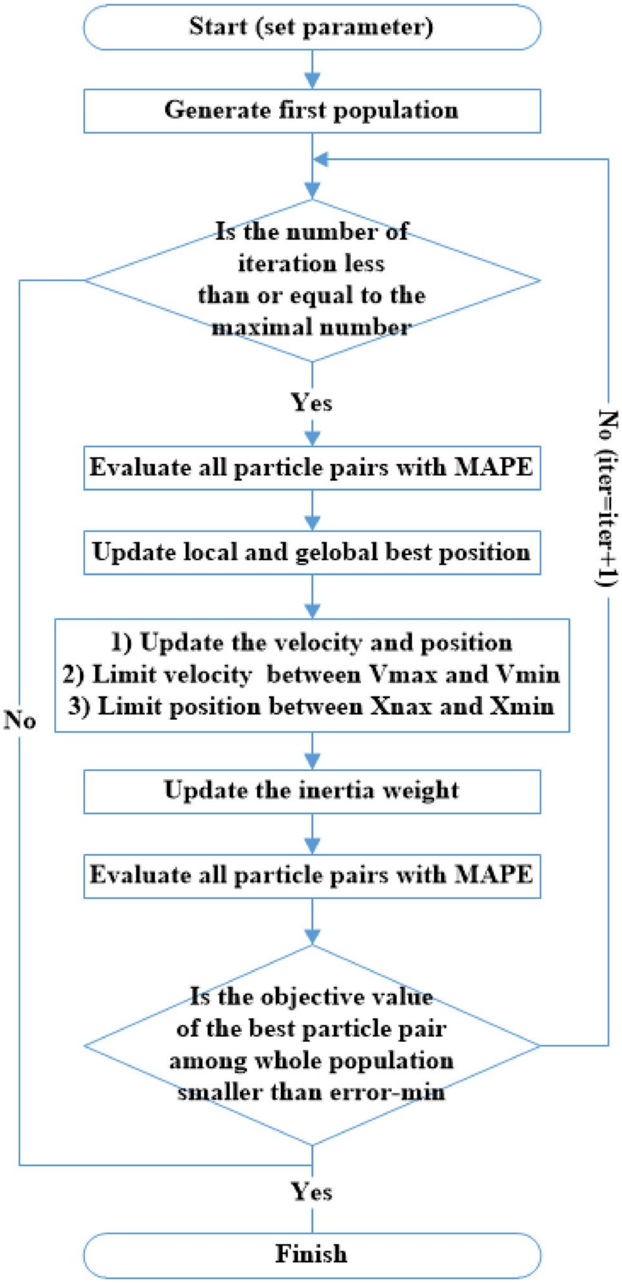

Start (set parameter)
Generate first population
Is the number of iteration less than or equal to the maximal number
Yes
Evaluate all particle pairs with MAPE
Update local and gelobal best position
1) Update the velocity and position
2) Limit velocity between Vmax and Vmin
3) Limit position between Xnax and Xmin
Update the inertia weight
Evaluate all particle pairs with MAPE
Is the objective value of the best particle pair among whole population smaller than error-min
No (iter=iter+1)
No
Yes
Finish

Fig. 9 Flowchart of CPSO optimization algorithm

4 Simulation results and discussion

In this section we introduce three real cloud workload traces which are used as baseline benchmark in the previous works [38, 40]. In order to compare our proposed algorithm with rival works in workload prediction which are based on ANN [21, 22], SVR [35–38] and GARCH [19, 41] standard metrics which are used to evaluate accuracy is explained. Also computational cost of each algorithm is reported for better comparison of results. It should be noted that the simulation is done on an Intel dual core T3400 with 2 GB memory.

4.1 Real cloud workload traces for benchmark

Cloud workload contains a lot of fluctuations. As described in [41], cloud workload is almost 20 times noisier than grid computing workload. So we like the other works [38, 40] consider three real cloud workload traces as baseline benchmarks which are as follows.

Worldcup98 [33]: This workload trace is related to website of world cup 1998. The load represents a set of users' requests logged into the website in every half an hour. Its variations and volatilities are dependent on many factors such as day time, week day and the time of the year that the games are held. The period of its fluctuations is 1 day. The workload is recorded since 30 April to 26 July 1998 and contains 1,352,804,107 requests.

CPU and memory usage traces in Google cluster [40]: This workload is related to the amount of used CPU and memory in the Google cluster. This workload is sampled every 5 min. The whole sampling time is 6 h and 15 min and it contains 75 samples. Due to security reasons these two workloads are normalized via an unknown linear mapping [40]. This workload contains 9218 jobs and 176,580 tasks.

Table 2 Prediction accuracy metrics

Criteria	Formula
Mean square error (MSE)	$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(\hat{y}_i - y_i\right)^2$
Root-mean-square error	$RMSE = \sqrt{\frac{\sum_{i=1}^{n}\left(\hat{y}_i - y_i\right)^2}{n}}$
Mean absolute error	$MAE = \frac{1}{n}\sum_{i=1}^{n}\left\lvert\hat{y}_i - y_i\right\rvert$
Mean absolute percentage error	$MAPE = \frac{1}{n}\sum_{i=1}^{n}\frac{\left\lvert\hat{y}_i - y_i\right\rvert}{y_i}$

4.2 Prediction accuracy metrics

In order to assess prediction accuracy of each algorithm we used standard metrics which are used in previous works [21, 38]. Each of these metrics somehow indicates difference between the actual value and the predicted value. In all metrics, n, $\hat{y}_i$ and y_i represent number of predicted samples, predicted value and actual value of the i'th sample. In Table 2, four popular assessment metrics for assessing prediction accuracy is depicted.

4.3 Workload prediction results

In this paper, three workloads; CPU, memory and Worldcup98 in conjunction with their corresponding predicted values and errors which are obtained via proposed W3PSG prediction algorithm are shown in Figs. 10, 11 and 12 respectively. In these three figures, workloads are shown with blue color and their predicted values are presented as red dash lines. Section (a), (b), and (c) show predicted values and the amount of user requests, predicting error value based on MAPE metric, and the error's histogram graph respectively. 15 first iterations of (a) sections are not predicted because of lack of initial training data. As it can be inferred from the W3PSG algorithm results (Figs. 10, 11, 12), predicted values (red dash line) follow load values (blue line) properly with little error. So W3PSG method has high accuracy and low prediction error. As it can be seen, W3PSG has a good performance even in swinging points and has small deviation from cloud workload. In all the simulations, a window with 15 sample length is used for training. This procedure continues with the movement of prediction window.

Prediction accuracy metrics for three workloads are shown in Tables 3, 4 and 5 respectively. As it can be inferred from the simulation results, hybrid wavelet transform based prediction models, W4PS, WPS3G and W3PSG, possess higher prediction accuracy in comparison whit other rival methods. The W3PSG has the highest prediction accuracy considering MAPE metric. W3PSG method has respectively 14.605%, 5.987% and 16.27% improvement over CPSO based SVR model and 24.529%, 15.788% and 55.631% improvement over basic SVR model. Best results in each table is marked as bold.

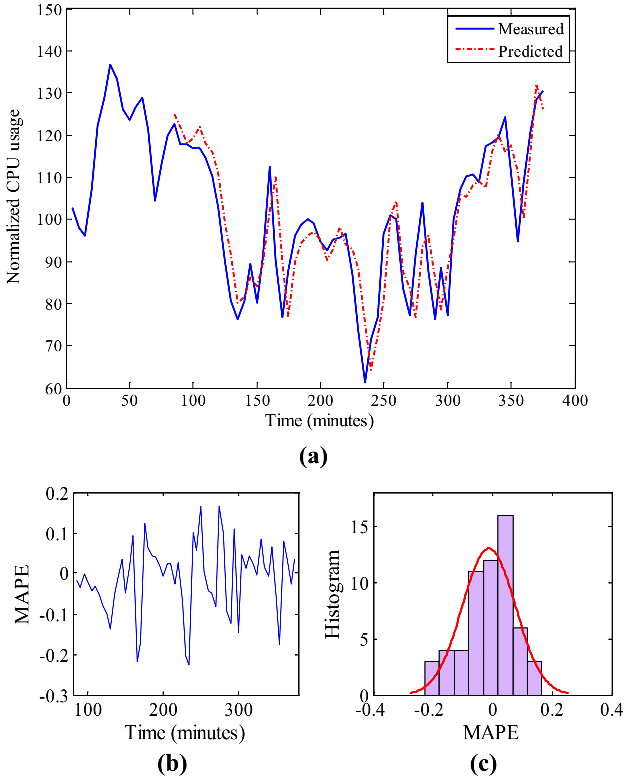


Fig. 10 **a** Google CPU workload prediction using W3PSG method, **b** along with prediction error, **c** error histogram

In Table 6, standard deviation and mean error of prediction for three workloads are illustrated. Notice to error histogram graph, it can be understood that prediction error is almost fallow Gaussian distribution. As it can be seen in Table 5, W3PSG has a smaller mean and standard deviation in comparison with other methods.

4.4 Impact of CPSO algorithm on SVR performance

In this section we compare the effect of optimization algorithm on convergence speed and final solution of SVR parameters and its impact on prediction accuracy. We consider chaotic Genetic algorithm (CGA) and CPSO to optimize SVR parameters. Simulation results indicate that the CPSO optimization method increases SVR accuracy more than CGA method which fails to find best solution in equal situation. MAPE cost function convergence regime of the best member for each workload is shown in Fig. 13; Fig. 13a–c show convergence speed of the CPSO which reduces SVR prediction error for CPU, memory and worldcup98 workloads respectively. By several experiments we select 10 population members and 100 iterations for CPSO algorithm. The final SVR parameters

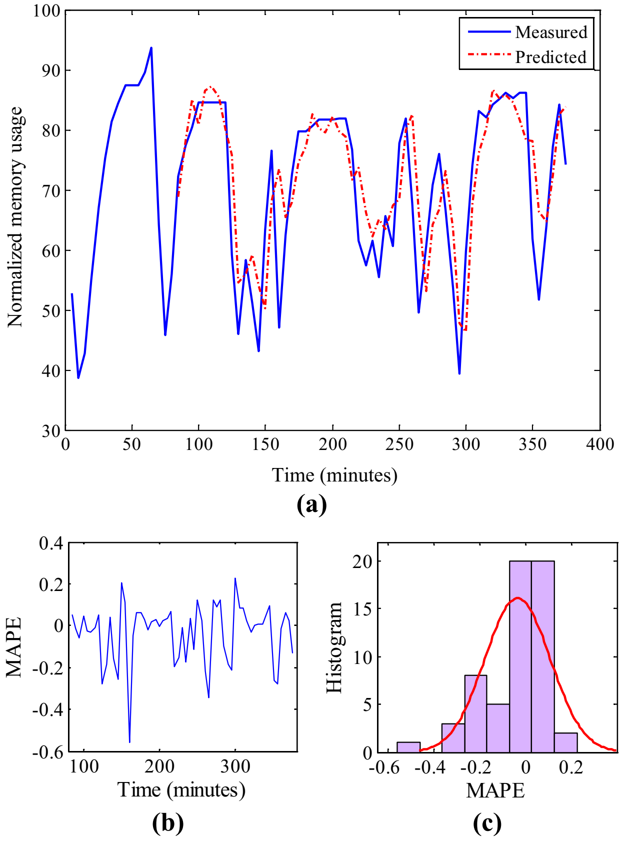


Fig. 11 **a** Google memory workload prediction using W3PSG method, **b** along with prediction error, **c** error histogram

using CPSO optimization algorithm are compared by the one achieved by CGA in Table 7.

4.5 Determine number of wavelet decomposition sub scales

One of the important aspects of the proposed algorithm is how to determine the number of sub scales which the workload time series should be decomposed to. It determines the level of workload details that we want to model and has direct impact on the accuracy of prediction. As described in previous works [12] Daubechies mother wavelet (db) is the best selection for fractal time series which is the case of cloud workload. Hence we select it as the best mother wavelet. The next step is to determine the number of wavelet transform sub scales. We conduct an experiment and test 1–8 sub scales (db1 to db8 wavelet transforms) in W3PSG to predict three baseline workloads. The results are shown in Fig. 14. Considering the results, db4 has the best performance and less prediction error.

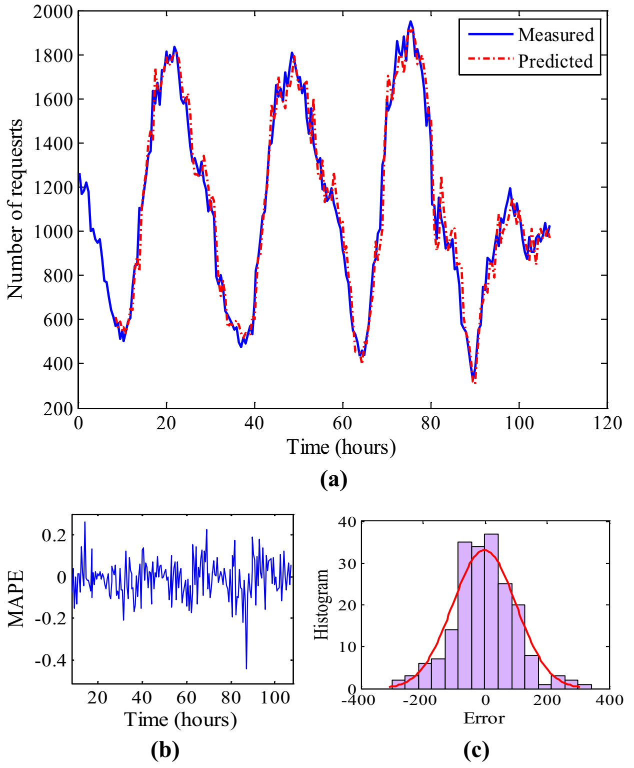


Fig. 12 **a** Worldcup98 workload prediction using W3PSG method, **b** along with prediction error, **c** error histogram

Table 3 Comparison of prediction methods tested on CPU workload

	MAE	MAPE	RMSE	MSE
GARCH	12.7212	13.1196	14.9385	223.1592
ANN	12.6665	13.1178	15.8685	251.8102
SVR	11.4899	12.1823	13.6275	185.7059
CPSO	10.9517	11.5075	14.5647	212.1297
W4PS	9.0065	9.6846	10.8019	116.6800
WPS3G	9.7181	10.4260	11.7972	139.1736
W3PSG	**8.4888**	**9.1941**	**10.2086**	**104.2146**

Table 4 Comparison of prediction methods tested on memory workload

	MAE	MAPE	RMSE	MSE
GARCH	12.1745	18.0194	13.5363	183.2323
ANN	12.5107	18.3430	15.2029	231.1276
SVR	10.6528	16.1286	12.7763	163.2335
CPSO	10.6210	15.7107	12.1230	146.9666
W4PS	9.2842	14.2913	10.8159	116.9831
WPS3G	9.1963	14.2900	11.3868	129.6589
W3PSG	**8.8711**	**13.5822**	**10.3413**	**106.9434**

Table 5 Comparison of prediction methods tested on Worldcup-98workload

	MAE	MAPE	RMSE	MSE
GARCH	341.1153	33.9013	413.5801	1.7105e+5
ANN	241.4541	25.2949	287.5850	8.2705e+4
SVR	168.6784	17.0253	204.2466	4.1717e+4
CPSO	87.1048	9.0311	116.9473	1.3677e+4
W4PS	79.4154	7.6850	103.5331	1.0719e+4
WPS3G	82.2152	8.3431	103.4400	1.0700e+4
W3PSG	**77.9170**	**7.5539**	**101.3441**	**1.0271e+4**

4.6 Comparison of computation cost of algorithms

To have fair comparison between prediction algorithms, we should consider the computational cost of the algorithms. CPSO takes about 19/93 s and it is done only once in training phase (offline). After optimization, optimal parameters are used in the SVR model. Hence the prediction algorithms are adaptive; exclude the CPSO, remaining parts such as ANN, SVR and GARCH have training and prediction computational cost periodically. As it shown in Table 8, the algorithms that use SVR model have almost equivalent training and prediction time durations. GARCH model uses much computing time for training and predicting. Therefore, instead of WPS3G algorithm, W3PSG model is proposed to reduce computing cost. In addition W3PSG have better prediction accuracy.

5 Conclusion

In this paper we propose an accurate workload prediction algorithm for dynamic resource allocation in MCC environments. Our proposed W3PSG method, in addition to gaining the highest accuracy, acts better than its rival algorithm, W3PSG, in terms of computation cost. In W3PSG wavelet transforms methods (mother wavelet db4) are used for decomposing the workload into four high and low frequency sub scales. In these methods, each one of the sub scales is predicted via a combination of SVR or GARCH models. In W3PSG, high frequency sub scales (D1) is predicted using GARCH and low frequency sub scales (A3) and high frequency sub scales (D2, D3) are predicted using a PSVR model in W3PSG. As mentioned in simulation results, W3PSG has the highest accuracy and also has a lower computational cost in comparison with WPS3G model. W3PSG model has 14/605%, 5/987% and 16/270% improvement of MAPE metric in comparison with W4PS and 24/529%, 15/788% and 55/631% compared to the base line SVR model. We also propose a new Meta heuristic parameter optimization algorithm for PSO named CSPO to adjust C, ε, σ parameters. In CSPO, search operation is repeated until the optimal values of the SVR model parameters are achieved. CSPO method, because of considering MAPE metric, excels base SVR model in term of prediction accuracy for CPU, memory and Worldcup98 workloads by 11/621%, 10/425% and 47/01% respectively. By considering MAPE metric, a general comparison of W3PSG method with ANN [21, 22], SVR [35–38] and CPSO optimized SVR methods for three cloud workloads, Google CPU, Google memory and Worldcup98, in terms of accuracy improvement, is given in Table 9.

Since in W3PSG we consider multi scale decomposed analysis and train for each sub scale a separate carefully tuned prediction model (SVR/GARCH); the final prediction which is ensemble of multi scale prediction has higher accuracy. Also, modeling high frequency components of the time series which has noisy and volatile nature by GARCH improved the prediction accuracy. However this method is computationally expensive. Devising CPSO to improve the accuracy of SVR by carefully tuning its parameters is another pillars of the W3PSG algorithm. For the future works we suggest to use other family of standard GARCH algorithm such as EGARCH to improve accuracy.

Table 6 Mean error and standard deviation

	GARCH	ANN	SVR	CPSO	W4PS	WPS3G	W3PSG
Google CPU							
Mean	5.745	5.520	4.2711	4.067	0.975	1.652	**0.852**
STD	13.970	15.072	13.1101	14.168	10.898	11.834	**10.306**
Google memory							
Mean	2.1797	4.1331	0.9577	0.0530	0.310	0.3473	**0.0033**
STD	13.534	14.821	12.9069	12.281	10.957	11.530	**10.476**
Worldcup98							
Mean	27.058	−13.10	−14.4865	2.268	−2.113	−1.034	**−2.010**
STD	413.858	288.097	204.3069	117.255	103.803	103.727	**101.610**

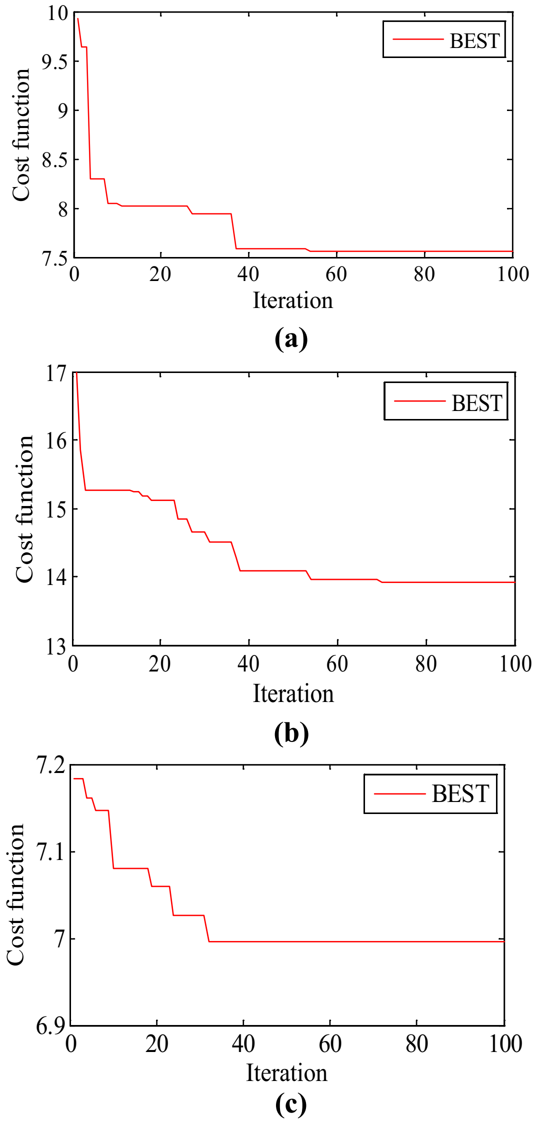


Fig. 13 Cost function of the best member of the CPSO optimization algorithm tested on three workloads **a** Google CPU, **b** Google memory, **c** Worldcup98

Table 7 Final SVR parameters using CPSO and CGA optimization algorithms

	CPU	Memory	Worldcup98
CGA			
E	0.08163	0.2872	0.09277
Σ	0.890	0.9	2
C	686.9272	461.9568	174.2565
CPSO			
E	0.07683	0.2771	0.09472
Σ	0.8858	11.811	5.448
C	407.2239	583.0683	560.1422

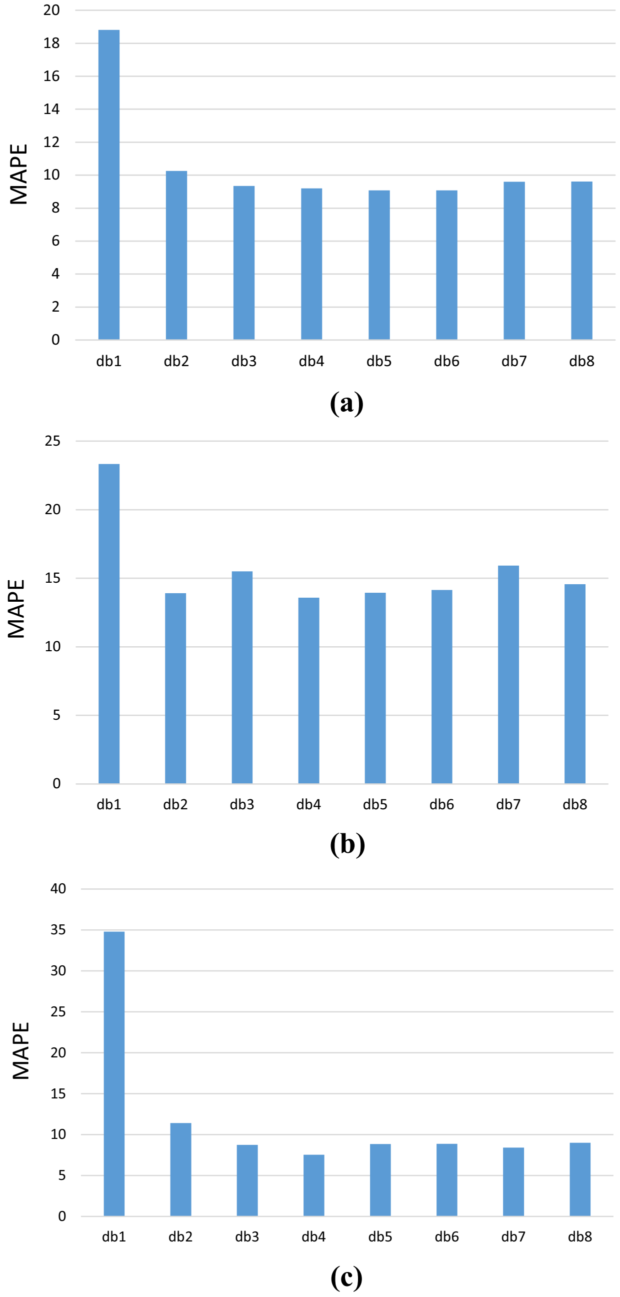


Fig. 14 Prediction error for wavelet transforms db1–db8 for the three baseline workloads, **a** CPU, **b** memory and **c** Worldcup98

Table 8 Training and prediction time in seconds

	GARCH	ANN	SVR	CPSO	W4PS	WPS3G	W3PSG
Train	0/3261	0/2311	0/00039	0/00037	0/0207	1/1698	0/3924
Predict	0/0290	0/0467	0/00017	0/00013	0/00084	0/0770	0/0283

Table 9 W3PSG accuracy improvement percentage over the three rival models

	ANN	SVR	CPSO-SVR
Google CPU	29/911	24/529	14/605
Google memory	25/954	15/788	5/987
Worldcup98	70/137	55/631	16/270

References

1. Korpi D, Tamminen J, Turunen M, Huusari T, Choi Y-S, Anttila L, Talwar S, Valkama M (2016) Full-duplex mobile device: pushing the limits. IEEE Commun Mag 54(9):80–87
2. Li W, Zhao Y, Lu S, Chen D (2015) Mechanisms and challenges on mobility-augmented service provisioning for MCC. IEEE Commun Mag 53(3):89–97
3. Wang Y, Chen I-R, Wang D-C (2015) A survey of mobile cloud computing applications: perspectives and challenges. Wirel Pers Commun 80(4):1607–1623
4. Fernando N, Loke SW, Rahayu W (2013) Mobile cloud computing: a survey. Future Gener Comput Syst 29(1):84–106
5. Zhang Q, Zhani MF, Zhang Sh et al (2012) Dynamic energy-aware capacity provisioning for cloud computing environments. In: Proceedings of the 9th international conference on Autonomic computing. San Jose, California, USA, 18–20 September 2012. https://doi.org/10.1145/2371536.2371562
6. Barroso LA, Hölzle U (2007) The case for energy-proportional computing. Computer 40(12):33–37. https://doi.org/10.1109/MC.2007.443
7. Fu Y, Lu Ch, Wang H (2010) Robust control-theoretic thermal balancing for server clusters. In: IEEE international symposium on parallel and distributed processing (IPDPS). Atlanta, GA, USA, 19–23 April 2010. https://doi.org/10.1109/IPDPS.2010.5470480
8. Qureshi A, Weber R, Balakrishnan H, Guttag J, Maggs B (2009) Cutting the electric bill for internet-scale systems. In: Proceedings of the ACM SIGCOMM 2009 conference on data communication, New York
9. Kusic D, Kephart JO, Hanson JE, Kandasamy N, Jiang G (2009) Power and performance management of virtualized computing environments via lookahead control. Clust Comput 12(1):1–15
10. Lajevardi B, Haapala KR, Junker JF (2015) Real-time monitoring and evaluation of energy efficiency and thermal management of data centers. J Manuf Syst 37(2):511–516
11. Mao M, Humphrey M (2012) A performance study on the vm startup time in the cloud. In: 2012 IEEE 5th international conference on cloud computing (CLOUD). Honolulu, HI, USA, 24–29 June 2012. https://doi.org/10.1109/CLOUD.2012.103
12. Ghorbani M, Wang Y, Xue Y, Pedram M, Bogdan P (2014) Prediction and control of bursty cloud workloads: a fractal framework. In: Proceedings of the 2014 international conference on hardware/software codesign and system synthesis, New Delhi, India
13. Rashidi S, Sharifian S (2017) A hybrid heuristic queue based algorithm for task assignment in mobile cloud. Future Gener Comput Syst 68:31–345
14. You C, Huang K, Chae H, Kim BH (2016) Energy-efficient resource allocation for mobile-edge computation offloading. IEEE Trans Wirel Commun 16(99):1397–1411
15. Karamoozian A, Hafid A, Boushaba M, Afzali M (2016) QoS-aware resource allocation for mobile media services in cloud environment. In: 13th IEEE annual consumer communications & networking conference (CCNC). Las Vegas, NV, USA, 9–12 January 2016. https://doi.org/10.1109/CCNC.2016.7444870
16. Valipour M, Banihabib ME, Behbahani SMR (2013) Comparison of the ARMA, ARIMA, and the autoregressive artificial neural network models in predicting the monthly inflow of Dez dam reservoir. J Hydrol 476(7):433–441
17. Nourikhah H, Akbari MK, Kalantari M (2015) Modeling and predicting measured response time of cloud-based web services using long-memory time series. J Supercomput 71(2):673–696
18. Calheiros RN, Masoumi E, Ranjan R, Buyya R (2015) Workload prediction using ARIMA model and its impact on cloud applications. QoS IEEE Trans Cloud Comput 3(4):449–458
19. Zhang J, Tan Z (2013) Day-ahead electricity price predicting using WT, CLSSVM and EGARCH model. Int J Electr Power Energy Syst 45(1):362–368
20. Chang BR, Tsai HF (2009) Novel hybrid approach to data-packet-flow prediction for improving network traffic analysis. Appl Soft Comput 9(3):1177–1183
21. Chenglei H, Kangji L, Guohai L, Lei P (2015) Predicting building energy consumption based on hybrid PSO-ANN prediction model. In: 34th Chinese control conference (CCC). Hangzhou, China, 28–30 July 2015. https://doi.org/10.1109/ChiCC.2015.7260948
22. Islam S, Keung J, Lee K, Liu A (2012) Empirical prediction models for adaptive resource provisioning in the cloud. Future Gener Comput Syst 28(1):155–162
23. Rashidi S, Sharifian S (2017) Cloudlet dynamic server selection policy for mobile task off-loading in MCC using soft computing techniques. J Supercomput. https://doi.org/10.1007/s11227-017-1983-0
24. Wu CL, Chau KW (2011) Rainfall-runoff modeling using artificial neural network coupled with singular spectrum analysis. J Hydrol 399(3–4):394–409. https://doi.org/10.1016/j.jhydrol.2011.01.017
25. Yaseen ZM et al (2019) An enhanced extreme learning machine model for river flow forecasting: state-of-the-art, practical applications in water resource engineering area and future research direction. J Hydrol 569:387–408
26. Hong WC, Dong Y, Zhang WY, Chen L-Y, Panigrahi BK (2013) Cyclic electric load predicting by seasonal SVR with chaotic genetic algorithm. Int J Electr Power Energy Syst 44(1):604–614
27. Hu Z, Bao Y, Xiong T (2014) Comprehensive learning particle swarm optimization based memetic algorithm for model selection in short-term load predicting using support vector regression. Appl Soft Comput 25:15–25. https://doi.org/10.1016/j.asoc.2014.09.007
28. Moazenzadeh R et al (2018) Coupling a firefly algorithm with support vector regression to predict evaporation in northern Iran. Eng Appl Comput Fluid Mech 12(1):584–597

29. Sanaei Z, Abolfazli S, Gani A, Buyya R (2013) Heterogeneity in MCC: taxonomy and open challenges. IEEE Commun Surv Tutor 16(1):369–392
30. Li C, Liu S, Zhang H, Hu Y (2013) Machinery condition prediction based on wavelet and support vector machine. In: 2013 international conference on quality, reliability, risk, maintenance, and safety engineering (QR2MSE)
31. De Giorgi MG, Campilongo S, Congedo PM (2014) Comparison between wind power prediction models based on wavelet decomposition with least-squares support vector machine (LS-SVM) and artificial neural network (ANN). Energies 7(8):5251–5272. https://doi.org/10.3390/en7085251
32. Sun Y, Leng B, Guan W (2015) A novel wavelet-SVM short-time passenger flow prediction in Beijing subway system. Neurocomputing 166:109–121
33. Roy N, Dubey A, Gokhale A (2011) Efficient autoscaling in the cloud using predictive models for workload predicting. In: 2011 IEEE international conference on cloud computing (CLOUD)
34. Chen K-Y (2007) Predicting systems reliability based on support vector regression with genetic algorithms. Reliab Eng Syst Saf 92(4):423–432
35. Zhang WY, Hong W-C, Dong Y, Tsai G, Sung J-T, Fan G-F (2012) Application of SVR with chaotic GASA algorithm in cyclic electric load predicting. Energy 45(1):850–858
36. Hong W-C (2009) Chaotic particle swarm optimization algorithm in a support vector regression electric load predicting model. Energy Convers Manag 50(1):105–117
37. Hong W-C (2011) Traffic flow predicting by seasonal SVR with chaotic simulated annealing algorithm. Neurocomputing 74(12–13):2096–2107
38. Barati M, Sharifian S (2015) A hybrid heuristic-based tuned support vector regression model for cloud load prediction. J Supercomput 71(11):4235–4259
39. Liang Y, Qiu L (2016) Network traffic prediction based on SVR improved by chaos theory and ant colony optimization. Int J Future Gener Commun Netw 8(1):69–78. https://doi.org/10.14257/ijfgcn.2015.8.1.08
40. Chen Y, Ganapathi A, Griffith R, Katz RH (2010) Analysis and lessons from a publicly available Google cluster trace. In: EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2010-95 94. https://arxiv.org/abs/1501.01426
41. Yang Q, Peng C, Zhao H, Yu Y, Zhou Y, Wang Z, Du S (2014) A new method based on PSR and EA-GMDH for host load prediction in cloud computing system. J Supercomput 68(3):1402–1417

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.