Towards Adaptive Web Services QoS Prediction

Jing Li1,2, Yongwang Zhao2, Jiawen Ren2, Dianfu Ma1,2
State Key Lab of Software Development Environment1 Institute of Advanced Computing Technology2
Beihang University
Beijing, China
{lijing, zhaoyw}@act.buaa.edu.cn, rjw888@126.com, dfma@buaa.edu.cn

Abstract—Quality of Service (QoS) has been widely used to support dynamic Web Service (WS) selection and composition. Due to the volatile nature of QoS parameters, QoS prediction has been put forward to understand the trend of QoS data volatility and estimate QoS values in dynamic environments. In order to provide adaptive and effective QoS prediction, we propose a WS QoS prediction approach, named WS-QoSP, based on the technique of forecast combination. Different from the existing QoS prediction approaches that choose a most feasible forecasting model and predict relying on this “best” model, WS-QoSP selects multiple potential forecasting models, combines the results of the selected models to optimize the overall forecast accuracy. Results of real data experiments demonstrate the diversified forecast accuracy gains by using WS-QoSP.

Keywords—Web Services; Quality of Service; QoS prediction

I. INTRODUCTION

Web Services (WS) [23] give a standardized solution for Service-oriented architecture (SOA) [19] applications. Quality of service (QoS) [15] refers to various non-functional properties of WSs, such as response time, availability, price etc. QoS properties of WSs can be categorized into two main types [26]: technical quality and managerial quality. The technical quality refers to properties that can be measured and improved using specific techniques, e.g. security, response time, availability and reliability. The managerial quality is used for capturing service management information, regarding, e.g. ownership, cost, validity period and visit count.

QoS properties of WSs have been widely used to support WS ranking, WS selection and WS composition [10, 13, 23, 24]. In real applications, both the network condition and status of WSs may change during the life time of a business process. Due to the uncertainty and volatility of WS QoS parameters, applications relying on static QoS data or providing inaccurate dynamic values of QoS parameters tend to suffer from poor availability and reliability [13, 24]. In the domain of WS QoS management, QoS registries have been put forward to store and monitor the performance of WS [17, 23, 24]. Among the fundamental services that QoS registries provide, the ability to estimate the future performance of WS at development time, and iteratively refine these estimates at execution time, can significantly reduce the overall application cost and risk [17, 24]. On the other hand, due to the changeful runtime environment and the possibility of system failures, QoS information might not be updated in time. In this case, one potential way to provide a missing value is to perform prediction based on the variety of historical data. This is especially useful in time sensitive applications.

WS QoS prediction has been recently studied as an important procedure to understand the trend of QoS data volatility and estimate QoS values so as to support intelligent WS QoS management [17, 24], QoS-based WS selection and composition [13, 23] in dynamic environments. In the existing approaches for Web Services QoS prediction, researchers basically choose a most feasible forecasting model and predict relying on this “best” model [13, 16, 19, 21]. We argue that these one-model based QoS prediction solutions are not adaptive and flexible for supporting real QoS prediction practices for the following reasons:

- As mentioned above, values of QoS properties may be influenced by different factors, technical factors or managerial factors. The trends of QoS data caused by different factors might be various. A uniform forecasting model may not support all types of QoS properties.
- The value of a QoS property may be influenced by multiple contextual factors nondeterministically. (e.g. The response time of a public weather forecasting service might be influenced by the underlying factors including the server’s hardware condition, the visit traffic and the network conditions.) Hence, due to the nondeterministic QoS data variation, the risk of not choosing the best forecasting model might be very high.
- When the historical QoS data are not very informative, it might be not possible to identify a single dominant forecasting model. From the aspect of system design, forecasting with a fixed model may suffer from poor flexibility and extendibility in dynamic environments.

To address these challenges, we propose an adaptive WS QoS prediction approach based on the technique of forecast combination, named WS-QoSP. The main contributions of this paper include:

1. We propose a novel WS QoS prediction approach, named WS-QoSP, based on the idea of forecast combination. A model selection algorithm is proposed for selecting the potential forecasting models. Three combination methods are used for accuracy optimization under different accuracy measures.

2. We conduct experiments using real QoS datasets of two different types of QoS properties to testify WS-QoSP.
The rest of the paper is organized as follows. Section 2 describes the adaptive WS QoS prediction approach, WS-QoSP. Section 3 states the design and results of our experimental study. Section 4 gives an overview of the related work. Section 5 makes a conclusion and outlines our future work.

II. ADAPTIVE QoS PREDICTION

Before presenting WS-QoSP, we first introduce some background knowledge of forecasting technologies.

A. Forecast Background

(1) Data Pattern

There are four basic patterns of data variation in time series forecasting theory: trended, seasonal, cyclical and stochastic [13]. The variation of a data sequence may follow one of the patterns or a combination of them. Forecasting methods [16] have been developed for each data pattern, and some of the methods can fit more than one pattern. Through a comprehensive analysis on the data pattern, type of model, the minimal data requirements and expert experiences, forecasters choose a feasible method to model the target data. Each forecasting method can generate a group of forecasting models by configuring different model parameters. Through the process of parameters determination, the best forecasting model which optimally matches the input data can be obtained. Then the best forecasting model is used to estimate the future values. In this paper, we are not concerned with the process of parameters determination. For simplicity, we use the term “forecasting model” to refer to the best forecasting model of the related forecasting method.

(2) Forecast Combination

In the traditional way of forecasting, forecasters choose the best forecasting model based on data analysis (e.g. Autocorrelation Analysis [13]) and former experiences. Combination forecasting has proposed to handle the situations where the data is not very informative and it is not possible to identify a single dominant model [9]. Combination forecasting technology has been successfully applied in diverse areas such as forecasting currency market volatility, money supply, stock prices, meteorological data, city populations, outcomes of football games and political risks [5].

The idea of combination forecasting is to combine the results of multiple individual forecasting models. A crucial step of combination forecasting is to determine the combination weight of each individual model. The methods of weight determination can be classified into two catalogs: variance-covariance methods and regression-based methods [6]. In variance-covariance methods (e.g. the optimal weighting methods [4], [12]), weights are determined by both the underlying variances and covariances. Regression-based methods combine forecasts by simply regressing realizations on forecasts. The most commonly used regression-based methods include time-varying combining weights, dynamic combining regressions, Bayesian shrinkage of combining weights toward equality, and nonlinear combining regressions [6].

(3) Forecast Evaluation

In the following we introduce three most common accuracy measures that serve as popular benchmarks to evaluate the performance of forecasting models.

- Mean Squared Error (MSE) [13]: MSE is computed by squaring each forecast error, and then averaging the squared errors. Let \( x_t, t = 1, 2, ..., N \) denotes the observed value of target data sequence at time \( t \). \( N \) is number of target data. \( \hat{x}_t, t = 1, 2, ..., N \) denotes the forecasting value at time \( t \). MSE is computed as follows:

\[
MSE = \frac{1}{N} \sum_{t=1}^{N} (x_t - \hat{x}_t)^2
\]

- Mean Absolute Error (MAE) [13]: MAE measures forecast accuracy by averaging the absolute values of the forecast errors. MAE is computed as follows:

\[
MAE = \frac{1}{N} \sum_{t=1}^{N} |x_t - \hat{x}_t|
\]

- Mean Absolute Percentage Error (MAPE) [13]: MAPE is computed by finding the absolute error in each period, dividing this by the actual observed value for that period, and then averaging these absolute percentage errors. MAPE is computed as follows:

\[
MAPE = \frac{1}{N} \sum_{t=1}^{N} \left| \frac{x_t - \hat{x}_t}{x_t} \right|
\]

B. System Model

The system model of WS-QoSP is based on the following assumptions: (1) QoS data values are collected from reliable and accurate sources. In this paper we do not mention how to identify the noisy data points that are collected due to technical errors or hostile attacks. (2) QoS data values are measured at successive times spaced at uniform time intervals. For simplicity, we are not concerned with the situation that data are collected at varying time intervals.

WS-QoSP consists of three stages. Figure 2 gives an overview of WS-QoSP. In the first stage, potential individual forecasting models are selected as the component models using the component model selection algorithm. In the second stage, each component model is evaluated based on the fitting results. Then a combining model is constructed through weight computation. After model construction, the combining model is improved by trimming the weak-performance component models. In the final stage, forecasting values are generated based on the final combining model.
C. Definitions

The following definitions are used in the rest part of this paper:

Definition 1. QoS data sequence (Q): A QoS data sequence \( Q = \{q_t | t = 1,2,...,n\} \) is the set of observed values of a target QoS property, where \( q_t \) is the observed value at time \( t \), \( n \) is the total number of the observed QoS data.

\( t' \) denotes the target future time point at which the value of QoS data is expected to be predicted.

Definition 2. Forecasting model (\( m \)): A forecasting model \( m \) is an individual forecasting model in the model library. \( M = \{m_i | i = 1,2,...,r\} \) denotes the set of universal forecasting models in the model library. Each forecasting model \( m_i \) is associated with a data pattern set \( P_i, P_i \subseteq \{T,S,C,ST\} \), that means \( m_i \) is applicable to the data patterns in \( P_i \), where \( T,S,C,ST \) represent the data patterns of trended, seasonal, cyclical and stochastic respectively. Each forecasting model has its minimum data requirement. \( m_i, nd \) denotes the minimum number of data that a non-seasonal model \( m_i \) requires; \( m_i, nsd \) denotes the minimum seasons of data that a seasonal model \( m_i \) requires.

Definition 3. Component model (\( c_m \)): A component model \( c_m \in M \) is a forecasting model which is selected for constructing the combining model.

\( CM = \{c_m | i = 1,2,...,v\}, CM \subseteq M \) is the set of component models, \( v \) is the number of component models. \( Q_s = \{q_s | t = 1,2,...,n\} \) is the set of fitting values generated from component model \( c_m \), where \( q_s \) is the fitting value of \( c_m \) at time \( t \). \( e_o = q_s - q_o \) is the fitting error of \( c_m \) at time \( t \).

Definition 4. Combining model (\( \hat{m} \)): A combining model \( \hat{m} =< CM, L > \) is a 2-tuple, where \( CM \) is the set of component models, \( L = \{l_i | i = 1,2,...,v\} \) is the set of weight coefficients, \( l_i, \sum_{i=1}^{v} l_i \geq 0 \) is the weighting coefficient of component model \( c_m \). \( \hat{m}(t') \) denotes the forecasting value at time \( t' \) generated from \( \hat{m} \).

D. Selecting Component Models

The first stage of WS-QoSP is to select appropriate forecasting models as component models. In order to model the different variation patterns of QoS data, component models should cover all types of data patterns: trended, seasonal, cyclical and stochastic. Moreover, since the gains are only possible if the models provide forecasts that differ, it is better to select a set of models that differ substantially from one another [1]. Thus, we select one component model for each data pattern. A component model selection algorithm (as shown in Table 2) which consists of the following three steps is proposed for model selection.

<table>
<thead>
<tr>
<th>TABLE I. COMPONENT MODEL SELECTION ALGORITHM.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Algorithm Selecting Component Models(M,Q)</strong></td>
</tr>
<tr>
<td><strong>Input:</strong> the set of universal models ( M ), the target QoS data sequence ( Q ), the accuracy measure ( c )</td>
</tr>
<tr>
<td><strong>Output:</strong> a set of component models ( CM )</td>
</tr>
<tr>
<td>1: for ( i = 1 ) to length(( M )) do</td>
</tr>
<tr>
<td>2: if ( (M[i].nd &gt; length(Q)) ) and ( (M[i].nsd &gt; slength(Q)) ) then</td>
</tr>
<tr>
<td>3: ( M ).delete(i)</td>
</tr>
<tr>
<td>4: else</td>
</tr>
<tr>
<td>5: ( M[i].c ) ( \leftarrow ) ( M[i].fit(Q) )</td>
</tr>
<tr>
<td>6: end if</td>
</tr>
<tr>
<td>7: end for</td>
</tr>
<tr>
<td>8: for ( i = 1 ) to length(( M )) do</td>
</tr>
<tr>
<td>9: for each ( X \in {T, S, C, ST} ) do</td>
</tr>
<tr>
<td>10: if ( (M[i].P_i.contains(X)) ) then</td>
</tr>
<tr>
<td>11: ( Mx.add(M[i]) ) /*( Mx \in {Mt, Ms, Mc, Mst} */</td>
</tr>
<tr>
<td>12: end if</td>
</tr>
<tr>
<td>13: end for</td>
</tr>
<tr>
<td>14: end for</td>
</tr>
<tr>
<td>15: for each ( Mx \in {Mt, Ms, Mc, Mst} ) do</td>
</tr>
<tr>
<td>16: for ( i = 1 ) to length(( Mx )) do</td>
</tr>
<tr>
<td>17: if ( (C_{x,c} &gt; Mx[i].c) ) then /* ( C_{x,c} \in {Ct, Cs, Cc, Cst} ) is the component model for ( Mx^* ) */</td>
</tr>
<tr>
<td>18: ( Cx \leftarrow Mx[i].c )</td>
</tr>
<tr>
<td>19: end if</td>
</tr>
<tr>
<td>20: end for</td>
</tr>
<tr>
<td>21: end for</td>
</tr>
<tr>
<td>22: for each ( Cx \in {Ct, Cs, Cc, Cst} ) do</td>
</tr>
<tr>
<td>23: if ( (CM.contains(Cx)) ) then</td>
</tr>
<tr>
<td>24: ( CM.add(Cx) )</td>
</tr>
<tr>
<td>25: end if</td>
</tr>
<tr>
<td>26: end for</td>
</tr>
<tr>
<td>27: return ( CM )</td>
</tr>
</tbody>
</table>
Model filtering: The first step is to filter out the inapplicable forecasting models based on the minimal data requirements (line 2 to 3).

Model fitting: The next step is to fit each forecasting model into the QoS data, and evaluate the fitting errors of each forecasting model (line 4 to 5). The key step of model fitting is to compute the best parameters. The best parameters of each model can be obtained by applying the existing model fitting techniques [16]. The accuracy measure can be set to MSE, MAE or MAPE (the selection of accuracy measure is discussed in section 3.3). The value of accuracy measure is computed according to the formulas in section 2.3.

Model selection: The next step is to classify the forecasting models into the four data patterns (line 8 to 14). Let \( M_t, M_s, M_c, M_st \subseteq M \) denote the model classes of data patterns: trended, seasonal, cyclical and stochastic respectively. Then select the most effective model with the lowest fitting error as the component model for each data pattern class (line 15 to 21). Since a forecasting model may fit more than one data pattern (e.g. Box-Jenkins models), there is a possibility to select a same model more than one time. Hence, the final step is to discard the redundant component models (line 22 to 26).

E. Constructing Combining Models

(1) Weight Computation

After model selection, the set of component models \( CM \) has been obtained for model construction. As defined in section 3.1, a combining model \( \bar{m} = (CM, L) \) is a weighted combination of the component models. The next step is to determine the weighting coefficient \( l_i \) for each component model.

The easiest weight computation method is the equal-weighted combination [12]. The strategy of equal-weighted combination is to simply assign equal weight \( l_i = 1/v \) (where \( v \) is the number of component models) to each component model. However, as component models may have unequal accuracies, equal-weighted combination cannot reflect the unequal contributions of the component models. In this paper, we adopt three optimal weighting methods that assign weights based on the accuracy contributions and optimize the overall accuracy.

a) Optimal sum squared error combination

Optimal weighting methods have been proposed to optimize overall forecasting accuracies under different accuracy measures. Optimal sum squared error method (SSE) [12] is one of the most commonly used optimal method. While choosing combination method SSE, the accuracy measure used to select component models (in section 3.2) is set to MSE.

\[
SSE = \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} \sum_{j=1}^{n} l_i e_i e_j
\]  

(4)

where \( e_i \) is the fitting error of combining model \( \bar{m} \) at time \( t \), \( e_i = \sum_{i=1}^{n} l_i e_i, e_i = q_t - q_n \).

Weight computation for SSE can be formulated as the following optimization problem. The optimization problem is quadratic.

\[
\text{min } SSE = \sum_{i=1}^{n} \sum_{j=1}^{n} l_i e_i e_j
\]  

\[
\text{s.t. } \sum_{i=1}^{n} l_i = 1, \quad l_i \geq 0
\]  

(5)

By solving this optimization problem using any nonlinear programming technique, the SSE weight coefficient for each component model can be obtained.

b) Optimal sum absolute deviation combination

Due to squaring forecasting errors, the SSE method has an effect of magnify the forecasting errors of component models. Optimal sum absolute deviation combination (SAD) [12] is an optimal method without the possible basis cause by error squaring. While choosing combination method SAD, the accuracy measure used to select component models (in section 3.2) is set to MAE.

The sum absolute deviation of combining model \( \bar{m} \) is denoted as SAD:

\[
SAD = \left| \sum_{j=1}^{n} \sum_{i=1}^{n} l_i e_i \right|
\]  

(6)

Weight computation for SAD can be formulated as the following optimization problem. Since the optimization problem is linear, we can use any linear programming technique to solve it. The complexity of method SAD is lower than the complexity method SSE where only nonlinear programming techniques can be used to solve the optimization problem [4].

\[
\text{min } SAD = \left| \sum_{i=1}^{n} l_i e_i \right|
\]  

\[
\text{s.t. } \sum_{i=1}^{n} l_i = 1, \quad l_i \geq 0
\]  

(7)

By solving this optimization problem using any linear programming technique, the SAD weight coefficient for each component model can be obtained.

c) Effective measure based combination

The optimal methods SSE and SAD achieve optimal accuracy while ignoring the time influence that later data may have more useful information than the former data. To take both the accuracy and time influence into consideration, we use another optimal combination method, effective
measure based combination (EFF) [4]. While choosing combination method EFF, the accuracy measure used to select component models (in section 3.2) is set to MAPE. Let \( \hat{e}_m, i=1,2,...,v, t=1,2,...,n \) denote the relative fitting error of component model \( cm_i \) at time \( t \). \( E= (\hat{e}_m)_{v \times n} \) is the relative fitting error matrix of \( cm_i \).

\[
\hat{e}_m = \begin{cases} -1 & \text{when } (q_i - q_{i_d}) / q_i < -1 \\ (q_i - q_{i_d}) / q_i & \text{when } -1 \leq (q_i - q_{i_d}) / q_i \leq 1 \\ 1 & \text{when } (q_i - q_{i_d}) / q_i > 1 
\end{cases}
\] (8)

\( A_m = 1 - |\hat{e}_m| \) is the accuracy of component model \( cm_i \) at time \( t \), \( i=1,2,...,v \), \( t=1,2,...,n \). Apparently, \( 0 \leq A_m \leq 1 \), when \( |(q_i - q_{i_d}) / q_i| > 1 \), \( A_m = 0 \). Zero value of \( A_m \) indicates component model \( cm_i \) has no effort at time \( t \). \( \hat{A}_m \) denotes the accuracy of combining model \( \hat{m} \).

\[
\hat{A}_m = 1 - |\hat{e}_m| = 1 - \sum_{t=1}^{n} \sum_{i=1}^{v} \hat{e}_{mi}, t=1,2,...,n
\] (9)

\( em = \sum_{t=1}^{n} Q_i A_m \) is the effective measure of \( cm_i \), where \( i=1,2,...,v \). \( Q_i \) is the set of discrete probability distributions, \( Q_i = \{Q_i | t=1,2,...,n\} \) is the set of discrete probability distribution at time \( t \). The value of \( Q_i \) represents the time influence that is how much the QoS value at time \( t \) is supposed to impact the forecasting value. For example, when later observations have greater influence than the prior ones, the value of \( Q_i \) should be set increasingly with respect to time \( t \). When we do not want to consider the time difference, the value of \( Q_i \) can be set equally \( Q_i = 1/n \).

Let \( em = \sum_{t=1}^{n} Q_i \hat{A}_m \) denotes the effective measure of \( \hat{m} \), where \( \hat{Q}_i = \sum_{t=1}^{n} Q_i \hat{A}_m > 0 \) is the discrete probability distribution of \( \hat{A}_m \). Apparently, when the value of \( em \) is larger, the model is more effective.

Weight computation for EFF can be formulated as the following optimization problem. The optimization problem is linear.

\[
\begin{align*}
\text{max } em & = \sum_{t=1}^{n} Q_i \hat{A}_m = \sum_{t=1}^{n} Q_i \left( 1 - \sum_{i=1}^{v} l_i \hat{e}_{mi} \right) \\
\text{s.t. } & \sum_{i=1}^{v} l_i = 1 \\
& l_i \geq 0
\end{align*}
\] (10)

By solving this optimization problem using any linear programming technique, the EFF weight coefficient for each component model can be obtained.

(2) Model Trimming

Rather than combining the full set of component models, it is often advantageous to discard the component models with the worst performances [1]. In an extreme case, the optimal forecast combination might put zero weight on the pure noise forecasts. Furthermore, forecasts that only add marginal information should also be dropped since they might increase more forecasting error rather than bringing benefits [1]. Thus, the final step of constructing the combining model is to trim the component models which have zeroed or relatively small weight coefficients.

Let \( l_i, 0 \leq l_i \leq 1 \) denote the weight. The value of \( l_i \) is set to 0 by default. It can also be set to a value in the range 0 to 1. For each component model \( cm_i \), if \( l_i \leq l_{cr} \), discard \( cm_i \) from the component model set \( CM \). \( CM = \{ cm_i | i=1,2,...,v \} \), \( CM \subseteq CM \) is the final set of component models, where \( v \) is the number of final component models. \( L' = \{ l_i | i=1,2,...,v, l_i \leq l_{cr} \} \) is the final set of corellative weight coefficients. So far, the final combining model \( \hat{m} = \{ CM', L' \} \) is obtained.

F. Generating Forecasting Values

The final stage is to generate forecasting values using each component model, and then combine the forecasting values using the combining model.

For each component model \( cm_i \), generate the forecasting value \( cm_i(t') \) at time \( t' \). Then, the forecasting value \( \hat{m}(t') \) of the combining model \( \hat{m} \) at time \( t' \) is computed as follows:

\[
\hat{m}(t') = \sum_{i=1}^{v} l_i \times cm_i(t')
\] (11)

III. EXPERIMENTAL STUDY

We have conducted two experiments to evaluate the performance of WS-QoSP. In the first experiment, real data are used to validate that WS-QoSP combines the advantages of multiple forecasting models and improves the forecast accuracy. In the second experiment, simulation experiments are conducted to evaluate the computation cost of each combination method.

A. Experiment settings

Experiments are conducted on a DELL Q9500 machine with 2 Intel Quad 2.83GHz processors and 4GB RAM. The machine is running under Windows XP and Java 1.6. The statistic software SPSS version 17 [28] is used for model fitting and forecast generation. The open source Mixed Integer Linear Programming (MILP) solver lp_solve version 5.5 [29] is used for solving the linear programming problems for combination methods SAD and EFF. MATLAB Optimization Toolbox 4.3 [30] is used for solving the non-linear programming problem for combination method SSE. There are 4 trended models, 3 seasonal models, 1 cyclical model and 2 stochastic models equipped in our model library. We have used two real QoS datasets which represent two different types of QoS properties: technical quality and managerial quality. One is response time dataset which is collected by consecutively querying a public WS resource,
the National Weather Forecasting Service \cite{26} and measuring the response time every hour during 10 days. The other is visit count dataset from statistics of hourly visit counts of the Collaborative Geospatial Visualization Service \cite{7} during 20 days. We conducted one-step-ahead prediction starting from the half time point of the total duration for each dataset.

In order to make a comparison to the “best” model, we selected the most feasible forecasting models each time according to the fitting results. We use labels “Best_MSE”, “Best_MAE” and “Best_MAPE” to refer to the best models under the three forecast error measures: MSE, MAE and MAPE respectively; labels “Com_SSE”, “Com_SAD” and “Com_EFF” to refer to the three combining models: SSE, SAD and EFF respectively; label “Observed” to refer to the observations.

\section*{B. Experiment results and discussions}

To evaluate the forecast accuracies for each method, we compute the average values of MSE, MAE and MAPE based on the forecasting results. Lower value of accuracy measure means higher accuracy. Figure 3 shows the results of the two datasets under the three forecast accuracy measures. It can be observed that on the both datasets, SSE model achieves better performance than the best model under MSE; SAD model outperforms the best model under the accuracy measure MAE; EFF model performs better that the best model while MAPE is used as the accuracy measure. The three combination methods all significantly improve the forecast accuracies on the two datasets of different QoS properties.

Figure 2. Forecasting accuracies under different measures.

In the second experiment, simulations have been conducted to evaluate the computation costs of the three combination methods. The target QoS data are generated randomly. The total number of QoS data varies from 50 to 8000. We execute the process of model combination 100 times and get the average execution time. The results displayed in Figure 4 show that by increasing the number of QoS data, SSE model which applies non-linear programming technique is more time-consuming and has more rapidly growing execution time compared to the other two methods that use linear programming technique to compute the weight coefficients.

Figure 3. Execution time with respect to the number of QoS data.
IV. RELATED WORK

WS QoS prediction has been put forward to support intelligent WS QoS management [21], [25]. QoS-based WS selection and composition [11], [14], [17] in dynamic environments.

In the domain of WS QoS management, Shao et al. [21] propose a collaborative filtering based approach to making similarity mining and prediction from consumers’ experiences. In [25], authors argue that performance metrics can be predicted based on historical data. They present the design and implementation of an event-driven QoS prediction system which can predict or refine the prediction of metrics in a real-time fashion.

From the aspect of supporting QoS-based service selection and composition in dynamic environments, researchers apply various forecasting technologies for WS QoS prediction. In [11], authors propose an Artificial Neural Network (ANN) based service performance prediction mechanism to support dynamic, personalized QoS-based service selection. Li et al. [14] use structural equation to model the QoS measurement of web services, and predict the change of quality of service dynamically so as to support adaptive QoS-based service selection. Malak et al. [17] propose a web services QoS prediction architecture to predict web service quality level during web services selection and monitoring phase. A double quantization time series forecasting model based on SOM Neural Networks is applied to predict web services QoS level.

In the existing approaches for Web Services QoS prediction, researchers basically choose a most feasible forecasting model and predict relying on this “best” model. Different from their solutions, we propose WS-QoSP, an adaptive WS QoS Prediction approach based on forecast combination. By taking the advantages of multiple potential forecasting models, WS-QoSP enables adaptability and flexibility in dynamic environments.

V. CONCLUSION

In this paper, we study the problem of WS QoS prediction. We argue that it is not possible to find a uniform forecasting model for all types of QoS properties. Moreover, prediction rely on a single forecasting model might suffer from poor adaptability and flexibility. We propose an adaptive QoS prediction approach, WS-QoSP, based on the technique of forecast combination. WS-QoSP aims at combining the advantages of potential individual forecasting models and optimizing the overall forecast accuracy. Experimental results demonstrate the diversified forecast accuracy gains by using WS-QoSP. Our experimental study has tried two QoS properties of technical quality type and managerial quality type respectively and achieved satisfactory results. An ongoing work is to add more QoS properties, such as reliability and availability and collect more real data to testify the approach.

ACKNOWLEDGMENT

This work was partially supported by the 863 program of China under Grants 2007AA010301 and 2009AA010310, the Chinese Special Project of Science and Technology: Core electronic devices, high-end general chips and infrastructural software (2010ZX01042-002-001) and the NFSC program China National Natural Science Funds under Grant 61003017.

REFERENCES


