Medium-term Client-Perceived Performance Prediction

Niko Thio and Shanika Karunasekera
NICTA Victoria Laboratory
Department of Computer Science and Software Engineering
University of Melbourne, Australia
{nthio, shanika}@csse.unimelb.edu.au

***Abstract*—In recent years the networking infrastructure has improved and as a result there has been a tremendous growth in the number of providers and the services they offer. With the wide choice of available services, many clients are interested in differentiating providers based on Quality of Service (QoS) - performance being one of the most important QoS attributes. In this paper we focus on provider differentiation based on client-perceived performance. The client-perceived performance better represents client experience compared to server-side performance measurement used widely today. We analyze and characterize client-perceived performance, based on Internet measurements. Based on this characterization, we propose a technique - last-period prediction (LPP) - for medium-term performance prediction between a client-provider pair, which will be used for provider differentiation. The LPP technique is specially designed to capture the characteristics of client-perceived performance, such as periodic fluctuations during the concerned period. Through experiments on the Internet, we demonstrate that the proposed technique provides better prediction accuracy for medium-term prediction of client-perceived performance compared to some popular time series models such as ARIMA, seasonal ARIMA, exponential smoothing, and Holt-Winters.**

***Index Terms*—Client-side performance, time series prediction, application-oriented performance**

I. INTRODUCTION

The Internet has enabled providers to offer services globally. In the context of Internet operations, clients are normally unaware of the location of a service provider. In recent years the networking infrastructure has improved and is now more widely available. As a result, there has been a tremendous growth in the number of providers and the services they offer. Depending on the application domain, many clients are interested in differentiating providers based on Quality of Service (QoS). This scenario is commonly found in cases where multiple providers that can fulfill a client's requirements exist. A typical example of such a scenario is in the Web Services (WS) context, where QoS-based provider selection has been actively discussed. Another example is server selection in application-layer anycast [1]. It is shown that dynamic selection based on QoS, especially performance, has advantages compared to static policy or random selection. In this paper, we focus on performance as one of the most important QoS attributes.

To date various approaches related to performance-based provider selection have been proposed from both industry and academia [2-4]. Many of these approaches have commonality on the general definition of performance metrics used to differentiate providers. One of the widely used approaches is to use performance metrics related to the provider, referred to as *server-side performance metrics*, such as request processing time, and processing capacity. These metrics are typically advertised by the provider for the client. This approach is adopted in Service Level Agreements (SLA) from HP and IBM [5-7], proxy-based UDDI extension [8], QoS Certifier [4], and UDDIe [9].

Server-side performance metrics, however, do not always represent the performance that is perceived by the client. Many different factors such as connection characteristics between the client and provider, and delays introduced by competing traffic are likely to occur in a real life environment; therefore server-side performance metrics may poorly represent the performance from the client point of view. As a result, server-side performance-based provider selection may not produce favorable outcomes from the client's point of view. In other words, the current provider selection that is based on server-side performance metrics is very likely to produce the result that does not satisfy clients' performance requirement, especially for the services that are offered world-wide in which the clients are expected to reside in a heterogeneous network environment.

In order to address this limitation, the provider recommendation system needs to be based on the performance metrics that represent the client experience accurately. We refer to this metrics as *client-perceived performance metrics*, which are measured from the client point of view, based on every invocations made by the client. These metrics have been utilized within provider recommendation systems in SPAND [10] and UX [11], as well as for web server's resource allocation control in eQoS [12, 13], and performance evaluation in WQMES [14].

One of the open challenges on utilizing client-perceived performance within a provider recommendation system is to recommend the appropriate provider based on *future* performance. Many third party providers today require subscription prior to service usage (such as listed in www.strikeiron.com). In this situation, once subscribed, the client is expected to use the chosen provider for a period of time (such as one week) before it can switch to other providers. Therefore, in order to produce an accurate provider recommendation outcome, candidate providers need to be compared based on the medium-term predicted performance (specifically, the period of the expected service usage), as opposed to the short-term predicted performance (e.g. the next service invocation). Unfortunately, medium-term performance prediction has received very little attention in the past works. Most of the existing provider selection schemes, such as [1, 10, 15, 16] are based on the short-term prediction, which do not accommodate the medium-term prediction needs of the current service-oriented provider selection schemes. Developing effective techniques to predict client-perceived performance in the medium-term still remains an open research challenge.

Aim

Prediction can be classified as either short-term, medium-term, or long-term from the forecast horizon point of view. Consider performance is measured hourly, short-term prediction generally refers to one or few step-ahead prediction (which corresponds to one to few hour ahead), and usually is intended for immediate use. For example, UX uses exponential smoothing for performance prediction, and it is intended that the enquiring client uses this prediction for choosing the provider to be used immediately.

In some other scenarios, performance prediction is required to be longer than a few hours ahead, such as a day or few days ahead. This essentially requires more than 24 step-ahead prediction (considering the measurement is collected hourly). We refer to this prediction as medium-term prediction, and this type of prediction is basically useful to assess a longer term of usage. Example of some projects which will benefit from this type of prediction are [2, 17]. These projects allow specification of the guaranteed period of performance on the service offered, and the selection is performed by matchmaking with the demand description in which the usage period is also specified.

Our aim in this paper is to investigate the medium-term prediction of client-perceived performance; specifically, we evaluated the prediction accuracy of nonseasonal and seasonal time series models. Seasonal time series models is evaluated for their ability to capture the seasonality property, which is in this paper context, is the performance fluctuation across time of day.

Contributions

Our investigations focus on two aspects: characterization of client-perceived performance, and evaluation of time series prediction methods for medium-term performance prediction (forecast horizon of one to seven days). Based on measurement gathered on the Internet, we analyze and model client-perceived performance. Based on this analysis we propose a novel prediction method for predicting medium-term client-perceived performance – *last-period prediction* (LPP). The LPP method is specifically designed to capture the distinctive characteristic of client-perceived performance series. We evaluate the proposed LPP scheme by comparing it with some popular time series prediction methods (i.e. ARIMA, Exponential smoothing, Seasonal ARIMA, and Holt-Winters). The results show that our proposed method produces significantly higher accuracy for medium-term prediction. Our evaluation in this paper is based on the real world client-perceived performance measurements, which are collected from an application deployed in the PlanetLab (http://planet-lab.org).

The organization of this paper is as follows: Section II discusses related work. Section III discusses client-perceived performance characterization. Section IV discusses the existing time series models and their limitations. We present our proposed prediction method to address the limitations. Section V presents an evaluation on the practical datasets, followed by a discussion in Section VI. The conclusion of this paper is discussed in Section VII.

II. RELATED WORK

Time series analysis has been used for prediction and modeling for various purposes and fields. ARIMA describes time series in terms of autoregressive and moving average components, and differencing is used for handling nonstationary series. Applications of ARIMA for performance prediction are NWS predictors [18] and the wide-area data transfer predictor in [19].

The exponential smoothing model is widely used for smoothing time series, where recent observations are given more weight, and the weight decreases exponentially as the observation gets older. This model is simple to implement, as it can be expressed in recursive terms, and is widely used for smoothing in applications such as round trip time smoothing at TCP protocol (RFC 793) and summarizing QoS in UX architecture [11].

Time series that exhibit seasonality can be modeled by Seasonal ARIMA or Holt-Winters model. Seasonal ARIMA is an extension of ARIMA model which accounts for the seasonality property in the time series. Example applications of seasonal ARIMA in network traffic forecasting include [20, 21]. The Holt-Winters model can be viewed as an extension of the exponential smoothing model, which handles trend and seasonality in time series. Network traffic anomaly detection [22] uses Holt-Winters prediction to define the normal traffic boundary.

Client-perceived performance prediction is a critical aspect of provider recommendation, such as in UX [11] and SPAND [10]. Both projects take into consideration only short term prediction such as one or a few step-ahead prediction. Prediction in SPAND uses a sliding window median, while UX uses exponential smoothing with 0.8 smoothing factor. Another project, eQoS [12, 13], also focuses on short term prediction as it aims to control QoS classes of the web server during heavy load period. Exponential smoothing with 0.5 smoothing factor is employed in eQoS for predicting the immediate performance.

III. CLIENT-PERCEIVED PERFORMANCE CHARACTERIZATION

We first focus on analyzing and characterizing client - perceived performance, which requires performance measurements that represent the performance experienced by the client. Although publicly available performance traces exist, we have not found any traces that directly relate to client-perceived performance we could use in our analysis. Although modifying existing live client applications to gather perceived performance was a possibility, we chose not to do this due to the complexity of the involved process. As an alternative we used a custom client-provider application, which was deployed on the Planetlab test bed. Using Planetlab allows us to observe the performance of our custom application under a realistic network environment. In this environment, communication between client and provider pairs can be considered very similar to a real world setting, although there could be minor differences. Our intention is to analyze the characteristics of the client-perceived performance which we believe will be similar to that exist in the real world applications.

We used a Web Services (WS) as our provider.. However, the performance measurement gathered is not specifically tied to the WS, context, thus is applicable to any TCP-based applications such as ordinary web applications.

Application and Environment Settings:

The WS provider application serves a fixed 100kB of textual data (100869 bytes including payload). A report in (www.optimizationweek.com/reviews/average-web-page) shows that the average webpage size is about 130kB. A modified Axis SOAP engine v1.1 (http://ws.apache.org/axis) was used on both of the server and the client to enable automated performance monitoring [23]. The client application (Java-based) was built to invoke the service for a given number of iterations per time interval. Jetty version 5.1.3 (http://jetty.mortbay.org/jetty) was used as the web container of the WS provider application. The WS provider application was deployed on four

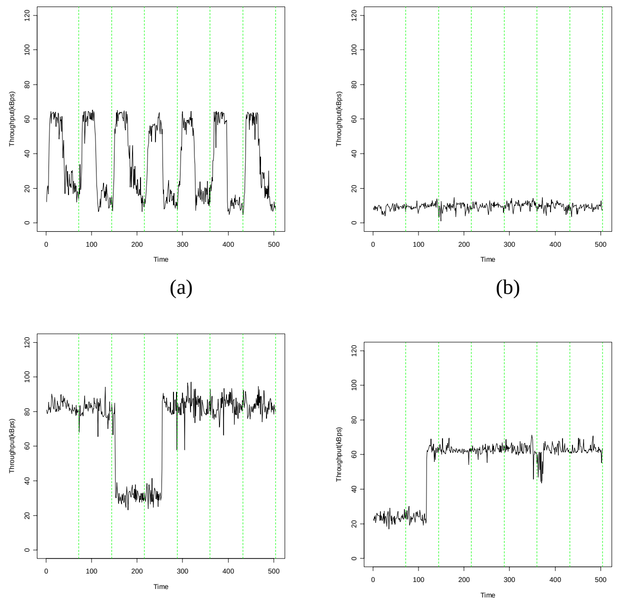

Figure 1: Plotted throughput (in kBps) against time. The vertical lines indicate start/end of the day.

different machines: a local server at University of Melbourne, and three PlanetLab machines. These providers are:
S_AU: Local server at University of Melbourne (Australia)
S_CN: PlanetLab node located at Beijing University (China)
S_US: PlanetLab node located at University of Pennsylvania (US)
S_EU: PlanetLab node located at Tübingen University (Germany)

Client applications were deployed on PlanetLab nodes in three different geographical regions, (US, Europe, and China), namely: New York University (U1-U3), Rutgers University (U4-U5), Princeton University (U6-U7), Polytechnic University (U8-U9), Kaiserslautern University (E1-E2), Swiss Federal Institute of Technology, Lausanne (E3), Swiss Federal Institute of Technology, Zurich (E4-E5), University of Karlsruhe (E6-E7), Tianjin University (C1-C2), and Tsinghua University (C3-C4).

In this setting, clients invoked the providers' service once every 20 minutes. Data was gathered from October to November 2005, and was filtered to avoid discontinuity as caused by downtime of PlanetLab nodes. The dataset therefore contains no missing data and therefore, time series models can be applied directly. In a real life situation, discontinuity is likely to be encountered and such discontinuities need to be dealt with using approaches such as listwise deletion, mean/median substitution, multiple regression, expectation-maximization (EM) algorithm, and multiple imputation [24]. Effectiveness of such techniques for handing discontinuities will be investigated in our future work.

Measurement Methodology:

In [23] we defined three application-oriented client-perceived measurement metrics: (1) *Latency*: is measured as the time between service request sent and earliest bytes response received. Latency measurement captures the round-trip time and the server processing time. Server code is possible to be modified to return the measurement of server processing time. (2) *Transfer rate*: is measured as the average of response transfer rate that excludes the latency. Transfer rate is essentially a measure of the connection's rate on transferring continuous data. Therefore, this measurement is independent of the latency measurement. (3) *Throughput*: is measured as the average of response rate (total response bytes divided by total response time). Throughput is measured in the same way as transfer rate and represented in the same dimension (i.e. kBps). For every invocation the measured throughput will never exceed the measured transfer rate since the latency cannot be zero. If the latency portion is much smaller than the total execution time, then the throughput will approach close to the measured transfer rate.

Throughput is used as the performance measurement in this paper since it captures both the latency and transfer rate components, which therefore reflects client experience from a more general sense.

Visual Observation of Client-Perceived Performance Trace Characteristics

The initial step in investigating client-perceived performance characteristics was the visual observation through plotted performance against time. Using the dataset collected from the PlanetLab experiment, Fig. 1 shows 1-week client-perceived performance measurement samples of four main patterns. Data traces are classified into two categories: seasonal and nonseasonal. Seasonal traces exhibit periodic disruption (mean or variance shifts) in a fixed time interval (e.g., daily). Fig. 1a provides an example of a daily pattern of mean shifts. Seasonal characteristics are generally caused by periodic activity, such as competing applications that run at regular intervals, or peak/off-peak activity. Nonseasonal traces do not exhibit periodic behavior, as shown in Fig. 1b. Fig. 1c and Fig. 1d show nonseasonal traces, in which the disruption occurs temporarily or permanently, rather than periodically. A temporal disruption (Fig. 1c) is indicated by a short period of disruption (in the context of observation time interval) followed by restoration of the baseline behavior. A permanent shift (Fig. 1d) is indicated by a shift which lasts for a relatively long period of time. In this example, it lasts for more than five days.

Testing Trend in Client-Perceived Performance Traces

Trend and seasonal components are contributing factors to nonstationary characteristics of time series. Our experimental datasets (observed visually) do not show persistent trends across a daily time interval. As a statistical measure, Phillips-Perron tests [25] can be used to test the existence of a linear trend in a time series, which assumes the given series has a unit root as null hypothesis.

The result of a Phillips-Perron test on our dataset gives a *p*-value of 0.01 for the whole dataset period (ranging from 8 to 33 days). The same result is produced for fixed periods of 1 and 3 days. The small *p*-value (1%) indicates that there is strong evidence against the null hypothesis for a 99% confidence level. In other words, the test suggests the absence of a linear trend in our datasets.

The absence of a trend is expected, since the client-perceived performance tends to be bounded in a certain range. In contrast to financial forecasting data, such as that used in the airline model [26], that exhibit persistent trend. Therefore, the prediction model should not capture the trend, although modeling with trend may give a better fit. Otherwise, the trend will be reflected in the prediction, which will deviate further from the bound as the forecast horizon increases. In turn, it will produce significantly larger prediction errors for medium-term prediction.

Testing Seasonality in Client-Perceived Performance Traces

Besides trend, seasonality is also a contributing factor of a nonstationary time series. A statistical test for the existence of seasonality is Bartlett's Kolmogorov-Smirnov test [27], which tests if the cumulative periodogram of the data follows a uniform distribution. The result of running this test on our experimental data shows that 97.4% of the 1-week series data exhibits significant seasonality with a 99% confidence level. It is observed that this test favors positive to seasonality even if the series has weak seasonality.

Seasonality strength can be measured through $R^2_{\text{autoregression}}$ [28]. This measurement gives a range from 0 to 1, where [28] prescribes a general interpretation as: nonexistence of seasonality or weak (0 to 0.4), moderate to strong seasonality (0.4 to 0.7), and strong to perfect seasonality (0.7 to 1). Before the seasonality strength can be measured, the seasonality length has to be determined. We used periodogram to identify dominant frequencies (i.e. the candidate seasonality lengths), and took the highest seasonality strength of these candidates. Therefore, the result indicates the upper bound of the seasonality strength. We applied this measurement in our datasets, and calculated the upper bound seasonality strength for every client-provider pair. The result is shown in Table I, in which we used [28] classification of weak/moderate/strong as described previously. It can be seen that the seasonality property is present in a large portion (72.2%) of our datasets, and

TABLE I.
SEASONALITY STRENGTH OF EXPERIMENTAL DATASETS

Seasonality Strength	Count Percentage (%)
Nonexistent to weak	27.8
Moderate to strong	51.1
Strong to perfect	21.1

TABLE II.
LENGTH OF SEASONALITY IN THE EXPERIMENTAL DATASETS

Data Length (Days)	Seasonality Length (Days)	Count Percentage (%)
3	0.5	12.4
	0.75	28.6
	1	40.8
7	0.5	7.9
	1	12.0
	1.75	23.3
	2.3	22.6

about half of the datasets exhibit moderate seasonality property.

The follow-up investigation was to identify the common seasonality length in our datasets through periodogram and seasonality strength measurements. Similar to the previous measurement, the periodogram was used to identify the candidate seasonality lengths, and the seasonality length was chosen as the one that gave the strongest seasonality measure. The result of this evaluation is shown in Table II.

It is shown that for short data lengths, there is a considerable number of 1-day seasonality periods; however, for long data length, only 12% show 1-day seasonality. This effect can be caused by weak daily seasonality, which is only apparent for short periods of observation. For longer periods of observation, the strong daily seasonality is required to be consistent throughout the observation.

Temporal Disruption

In order to quantify the occurrence temporal disruption in our dataset, we applied change-point detection to detect mean shifts, and for each segments (i.e. the interval between change points) the mean was calculated. Disruption can be identified at this stage by observing the mean shifts; however, there is a practical difficulty that the segment means before and after the disruption is slightly different. Therefore, before identifying disruption, we defined states in which for each state is associated with a range of performance. For determining the range dynamically, we used QT Clustering algorithm [29] on the segment means.

Every segments therefore is associated with state, and we identify disruption based on this state changes. Each state is numbered and ordered, so that the upper bound performance of the state with lower number is lower than the lower bound performance of the state with the higher number. For each trace, the baseline state is considered to be the median state of the trace, and disruption is considered if the states following the baseline state are lower, and the length is calculated as the total length of the disruption until the baseline state or higher, is restored. An example of this procedure is shown in Fig. 2, in which from the state changes shown in (d), there are five occurrences of temporal disruptions.

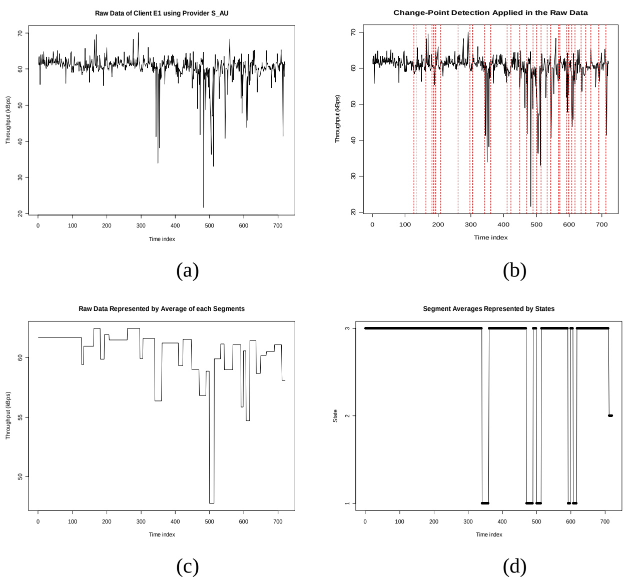

Figure 2: Procedures on identifying temporal disruptions: (a) plotted raw data, (b) applying change point detection, (c) taking the average of each segments, and (d) classify the averages into states produced by QT clustering.

The results of running this procedure on our dataset shows about half of the occurrences of temporal disruption last for less than four hours, and about 80% of the occurrences the disruption lasts for less than thirteen hours.

Permanent Shift

Permanent shifts can be identified with a similar procedure as that used to identify temporal disruption. Quantifying the occurrence of permanent shift practically can be difficult without defining the minimum length of the shift.
Consider taking three days as the minimum length of the shift, we define that the shift is considered permanent if for a point within the dataset, the baseline behavior of the previous three days and the three days ahead from this point is different. Baseline behavior is determined by taking the median of the states of the specified interval of time.
We observed that for the minimum shift length of three days, there are 145 permanent shifts within the whole dataset. For the minimum shift length of seven days, we observe 37 permanent shifts. The number of occurrence of permanent shifts is relatively low compared to the number of the occurrence of temporal disruption which is 798 disruptions.

IV. CLIENT-PERCEIVED PERFORMANCE PREDICTION

In this section, we evaluate existing time series approaches for medium-term prediction of client-perceived performance. The prediction evaluation is first based on synthetic datasets generated to simulate the various traffic characteristics observed in the real-world dataset presented in Section III. Prediction models are tested on these synthetic datasets in order to examine their behavior on performing medium-term prediction. Evaluation based on the real dataset will then be performed in Section V to compare the prediction accuracy quantitatively.

Existing Time Series Approaches for Performance Prediction

We investigated four existing time series prediction models: ARIMA, exponential smoothing, seasonal ARIMA, and Holt-Winters.

1) ARIMA Model: The ARIMA (Autoregressive Integrated Moving Average) model can be viewed as consisting of 3 components: autoregressive (AR), moving-average (MA), and differencing (I). The AR model describes the time series as an expression of the previous observation, while the MA model describes it as an expression of previous errors. Differencing is aimed at dealing with nonstationary series, such as those which exhibit linear trend.

An ARIMA(p,d,q) model of time series is expressed as:

$$\left(1-\sum_{i=1}^{p}\phi_i B^i\right)(1-B)^d y_t = \left(1-\sum_{i=1}^{q}\theta_i B^i\right)\varepsilon_t$$

or

$$\phi_p(B)(1-B)^d y_t = \theta_q(B)\varepsilon_t$$

Where:

p: order of autoregressive part

d: order of integrated part

q: order of moving average part

B: lag operator, where

ϕ_i : autoregressive parameters

θ_i : moving average parameters

ε_t : error terms

2) Exponential Smoothing Model: Exponential smoothing is a popular model for smoothing time series, as well as for prediction. Exponential smoothing assigns exponentially decreasing weights as the observation gets older.

Exponential smoothing is expressed as:

$$s_t = \alpha \cdot y_t + (1-\alpha)\cdot s_{t-1}$$

Where:

α : smoothing factor in which the value $0<\alpha<1$

s_t: smoothed observation at time t

3) Seasonal ARIMA: Seasonal ARIMA (SARIMA) extends the ARIMA model to handle a seasonality component.

An SARIMA(p,d,q)(P,D,Q)L model is expressed as:

$$\Phi_P(B^L)\phi_p(B)(1-B^L)^D(1-B)^d y_t = \Theta_Q(B^L)\theta_q(B)\varepsilon_t$$

Where:

L: seasonality period

P: order of seasonal autoregressive part

D: order of seasonal integrated part

Q: order of seasonal moving average part

4) Holt-Winters Model: The Holt-Winters model is an extension of exponential smoothing which considers trend and seasonality.

Holt-Winters model is expressed as:

(Multiplicative model)

$$s_t = \alpha\frac{y_t}{I_{t-L}} + (1-\alpha)(s_{t-1} - b_{t-1})$$

$$b_t = \beta(s_t - s_{t-1}) + (1-\beta)b_{t-1}$$

$$I_t = \gamma\frac{y_t}{s_t} + (1-\gamma)I_{t-L}$$

$$F_{t+m} = (s_t + m\cdot b_t)\cdot I_{t-L}$$

Where:

s_t: smoothed observation at time t

b_t : trend smoothing at time t

I_t : seasonal smoothing at time t

L : seasonal period

F_{t+m} : forecast at m periods ahead

α,β,γ : is overall, trend, and seasonal smoothing parameters respectively.

Limitation of Existing Time Series Approaches for Medium-Term Client-Perceived Performance Prediction

In the earlier section, we discovered that the seasonality characteristic is expected to be found in the client-perceived performance measurement series. While this may suggest the seasonal time series models (i.e. Seasonal ARIMA and Holt-Winters) would produce higher prediction accuracy for medium-term prediction as it can capture the seasonality characteristic in their model; our preliminary evaluation using synthetic dataset reveal an interesting outcome. We found that seasonal models produce relatively accurate medium-term (e.g. 1-day) prediction, only if the seasonality of the series is strong. However, in the case where the seasonality is moderate, the prediction outcome can be drifted (especially for Holt-Winters model) for medium term, thus degrading the prediction accuracy severely. Nonseasonal models (i.e. ARIMA and exponential smoothing) in such case may produce lower prediction error than the seasonal counterpart.

Nonseasonal models, however, have the limitation that they cannot capture seasonal characteristic. Despite they may produce lower prediction error in some scenarios, the prediction outcome will exclude the fluctuation pattern (e.g. performance during peak time or off-peak time) that is valuable to distinguish providers at a particular timeframe, such as in temporal-aware provider recommendation systems [17].

Last-Period Prediction (LPP) Method

In order to overcome the limitations of the current models, we propose the *last-period prediction* (LPP) method, which we developed to capture seasonality properties as well as to deal with temporal and permanent disruption scenarios. While the problem of disruption scenarios in seasonal models cannot be eliminated completely without knowledge of the type of disruption in the series being examined, it can be suppressed so the effect diminishes quickly after a few seasons, provided there are no further occurrences. On the other hand, if the disruption occurrences are consistent over seasons, then the effect will be maintained, allowing seasonality behavior to be captured.

In our proposed scheme, a prediction is based on the last period and is independent of the most recent observation. Consider for a seasonal period of L, the forecast for t+m is:

$$\hat{y}_{t+m} = y_{t+m-L} \tag{1}$$

This strategy is essentially analogous to last-value prediction as in CPU load prediction in [30], and one of the NWS predictors [18], with L=1.

There are three main steps involved in the above scheme: variance suppression, determining seasonality length, and multiple-period consideration.

1) Variance Suppression: The basic model as described in (1) has the potential to result in high forecasting error in cases where variation of y_t is high. We hypothesize that suppressing the variance through smoothing will reduce the forecasting error significantly compared to using the raw data. The model is then expressed as:

$$\hat{y}_{t+m} = S(y_{t+m-L}) \tag{2}$$

where S() is the smoothing function.

We evaluated two smoothing techniques: exponential smoothing and segmented-average. Exponential smoothing is a well known scheme to smooth noisy series [31]. Segmented-average employs a change-point detection heuristic to segment the time series, and takes the average of each segment as the final result. In order to identify the change points within the time series, we employ a heuristic based on the Hidden Markov Model (HMM) [32]. The basic idea of this heuristic is to infer the state changes (either mean or variance shifts) of the given observation, which in this case is the time series of the client-perceived performance measurement. The model is trained using a sample of client-perceived time series measurements to estimate the model parameters. These parameters are then used to predict the state changes of the given observation. The state changes in the resulting state sequence can be used to infer the change points. While the number of states (i.e., segments) is unknown, by iteratively segmenting the two states we can infer all change-points of the time series. Iteration stops when the resulting adjacent segments have equivalent statistical properties, which is determined by using a statistical

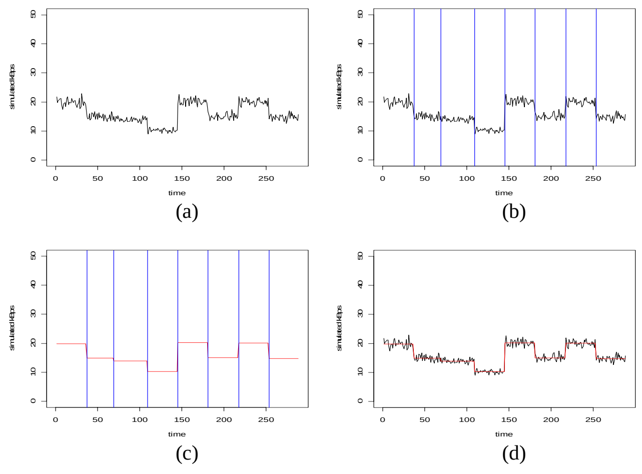

Figure 3: Illustration of segmented-average application of a synthetic dataset: original time series (a), identifying change points (b), averaging each segment (c), plotted original and smoothed series (d).

t-test (mean test), and F-test (variance test).

After segmentation is performed, the average of each segment can be calculated and used to represent the smoothed value of the segment time interval. The segmented-average smoothing process is illustrated in Fig. 3. Fig. 3a shows the original series. Fig. 3b shows the segmented series using the change-point detection heuristic, and Fig. 3c shows the average calculated for each segment, which is the final result of the segmented-average. Fig. 3d shows the superimposed original series and segmented-average.

2) Determining Seasonality Length: Seasonality length can be determined from spectral analysis and the seasonality strength measurement as described in Section III. For initial selection of seasonality length, daily seasonality was assumed, since it is prevalent as shown in Section III. The effect of seasonality length assumption will be investigated in terms of its accuracy and significance in Section V.

3) Multiple Period Consideration: The last-period scheme implies that for 1-period-ahead prediction, only the last period is considered. In the case in which a temporal disturbance occurs in the previous period, it will be fully reflected in the 1-period-ahead prediction. This effect has the potential to affect the prediction accuracy, especially for multiple-period-ahead predictions, which accumulate the reflection of temporal disturbance.

This effect can be suppressed by considering previous periods, and exponential smoothing can be used to derive the prediction result, which weights the recent period more heavily than the older periods. This scheme is expressed as:

$$\hat{y}_{t+m} = ES(\{S(y_{t+m-L}), S(y_{t+m-2L}), S(y_{t+m-3L}), \ldots, S(y_{t+m-nL})\}) \tag{3}$$

In (3), ES() is the exponential smoothing function, and n goes as far as the historical data still available, or is predetermined by a cut-off setting, depending on the number of periods to be considered.

V. EVALUATION

In this section we present the results of the experiments we carried out to evaluate our proposed concepts by using the datasets we collected as described in Section III, which is based on the measurements of a WS application on deployed on PlanetLab nodes. We use this dataset as the basis of evaluation, as it represents the measurement gathered from a real world application.

Model Fitting and Prediction Error Measurement

Throughout the evaluation, we use the metric referred to as *throughput*, which is the ratio between message being exchanged and the client access time (in kBps) to represent effective performance, as it accounts the delay introduced by both of the network and the server processing time [23]. Furthermore, the evaluation was based on prediction cases with 3 days of training data. Each prediction case was created such that there was no overlap in training data for different cases. For example, consider a dataset with a length of 8 days, in a 1-day forecast horizon scenario. There are two prediction cases that can be created from this dataset. This method of extraction is intended to minimize the correlation among prediction cases, therefore avoiding bias in the overall analysis.

Model fitting and prediction was performed using the R statistical package (http://www.r-project.org). For ARIMA model fitting, we used the R library supplied by Rob Hyndman (http://www-personal.buseco.monash.edu .au/~hyndman). Model fitting employed in this implementation is basically trying different sets of p and q which in ARIMA(p,d,q). Parameter d was determined by another procedure which utilizes the Phillips-Perron unit-root test. The best model was chosen such that it minimized AIC.

Prediction error in a particular time was calculated as the relative difference to the actual performance of the particular time. For example, at time t, the predicted value is $\hat{y}_t$ and the actual value is y_t, therefore, the prediction error for time t is:

$$\varepsilon_t = \frac{\hat{y}_t - y_t}{y_t}$$

The prediction error of one prediction case was calculated as the root mean squared relative error (RMSRE) for the whole forecast horizon n, which is expressed as:

$$RMSRE = \sqrt{\frac{1}{n}\sum_{t=1}^{n}\left(\frac{\hat{y}_t - y_t}{y_t}\right)^2}$$

Evaluation of LPP Settings

There are three settings in the LPP method which were evaluated: variance suppression methods, seasonality length assumption, and multiple period considerations.

1) Evaluation of Variance Suppression Methods:

Variance suppression methods evaluated were exponential smoothing and segmented-average. Raw data, without variance suppression, was also included in the comparison. These three methods are referred to as:

1. Last-period (LP) - Raw: no smoothing performed.
2. LP-ES: smoothing using exponential smoothing of the previous day.
3. LP-SegAvg: smoothing taking averages of the segments of the previous day.

The last-period scheme in this evaluation assumed a period length of one day. These smoothing methods were applied to perform a 1-day prediction (72 prediction points) by using 3 days of training data. For each method, 679 prediction cases were derived from the datasets. Visual observation through empirical cumulative distribution function showed that in general any smoothing technique tested reduces prediction error, compared to using the raw data directly.

Side-by-side comparison was performed by examining paired differences, for example, examining prediction error differences between LP-Raw and LP-ES for each case. A statistical test was employed to determine whether the mean (parametric tests) or median (nonparametric tests) were significantly different, according to the predetermined confidence level.

It was found that the distributions of prediction error differences were not normal, as tested using Anderson-Darling normality test with the confidence level of 99.9%. One possible reason is that some prediction cases exhibit different types of features, such as permanent or temporal disruptions, and different levels of performance variations. In this evaluation, we focused on the forecasting performance based on the overall cases, rather than based on the specific nature of the prediction cases. Therefore, we employed a nonparametric test (Wilcoxon test) to evaluate whether one method has a lower median of prediction error than another. The result shows that the prediction error median of LP-SegAvg and LP-ES is significantly lower than LP-Raw, with a confidence level of 99.9%. The results for a forecast horizon of 2 to 7 days, also shows the same result.

The results show that smoothing improves the prediction accuracy significantly. Furthermore, smoothing using the segmented-average produces significantly lower median error compared with using exponential smoothing throughout all forecast horizons. This result was confirmed through the paired Wilcoxon test with 99.9% confidence level.

For the subsequent evaluations in this paper, segmented-average was chosen as the smoothing technique as it produces less prediction error.

TABLE III.
P-VALUE OF MOOD'S TEST OF SIX SEASONALITY LENGTH SETTINGS

Forecast Horizon (days)	P-Value
1	0.590
2	0.741
3	0.965
4	0.875
5	0.916
6	0.765
7	0.947

2) Evaluation of the Significance of Length of Seasonality:

In order to evaluate the significance of the length of seasonality in the last-period prediction scheme, we first chose SegAvg as described in the previous section as the smoothing technique. There were six schemes to be compared in this evaluation (seasonality assumptions were taken from Table II):

1. Assume daily seasonal period
2. Assume 0.5 day seasonal period
3. Assume 0.75 day seasonal period
4. Assume 1.75 day seasonal period
5. Assume 2.3 day seasonal period
6. Determine seasonal period from training data

One-day predictions (72 prediction points) were performed for each seasonality assumption scheme using 5 days of training data. Five days of training data was chosen, since 2.3-day seasonality requires a minimum length of 5 days (at least twice the assumed seasonality).

The Mood's test (nonparametric alternative of a one-way ANOVA) was performed to test whether one or more of the means of these six settings is significantly different. The result is shown in Table III, which indicates that there is no significant difference between the median values among these settings. This suggests that any popular seasonality can be chosen without significantly penalizing prediction accuracy.

3) Evaluation of Multiple-Period Consideration:

This evaluation addressed the prediction accuracy of the scheme which considers multiple periods compared to a scheme which only considers a single seasonal period. The LPP method in this evaluation assumed a period length of one day, and used SegAvg for smoothing. The two schemes to be compared were:

1. LP-SegAvg: using only the single seasonal period.
2. LP-SegAvgES: using all available seasonal periods which were smoothed through exponential smoothing (as in equation 3).

These two schemes were applied to perform 1-day predictions (72 prediction points) by using 3 days of training data. For each method, there were 679 predictions made using our datasets.

TABLE IV.
P-VALUE OF PAIRED WILCOXON TEST FOR LP-SEGAVGES AND LP-SEGAVG METHODS WITH AN ALTERNATE HYPOTHESIS OF LP-SEGAVGES<LP-SEGAVG

Forecast Horizon (days)	P-Value
1	0.400
2	0.022
3	>0.001
4	>0.001
5	>0.001
6	>0.001
7	>0.001

The result of the paired Wilcoxon test is shown in Table IV, which shown that LP-SegAvgES produces significantly lower median error compared to LP-SegAvg with a 99.9% confidence level, starting from the forecast horizon of 3 days. This suggests that multiple-periods have an advantage over single-period for longer forecast horizon.

The evaluation results for LPP settings are summarized as follows:

1. Smoothing reduces prediction error significantly. Smoothing using SegAvg as evaluated, produces significantly lower prediction error compared to using exponential smoothing.
2. On choosing the seasonality period, evaluation shows there is no significant difference (in terms of median prediction error) between using different seasonality periods. Furthermore, these seasonality period candidates need to be selected through periodogram and the $R^2_{\text{autoregression}}$ measure from the training data.
3. Multiple-period consideration reduces the prediction error significantly for longer forecast horizons (3 or more days), since the multiple-period consideration is able to suppress temporal disruption, and this effect is more apparent for longer forecast horizons. For the following evaluations, we used daily-based LP-SegAvgES for comparison against other models, as these settings produced optimal prediction. Daily-based was chosen as it can reduce computation complexity without increasing prediction error significantly, and there is evidence of daily cycle of internet activity reported in other literature [33].

Predictions of Various Time Series Models

This section examines the prediction errors of time series models discussed in Section IV: nonseasonal models which are ARIMA and exponential smoothing (ES); and seasonal models which are seasonal ARIMA (SARIMA) and Holt-Winters (HW). The LP-SegAvgES method with daily period is also evaluated. Comparison of the models was based on the prediction cases from real data traces as described in Section III. The prediction cases of previous evaluations were based on a default starting time (i.e., the prediction always started at 00:00 +10GMT). In this evaluation, the prediction cases needed to be extended so they covered different starting times, rather than choosing an arbitrary starting time. This need arose for nonseasonal models where their prediction was potentially affected by the most recent observation. The prediction cases were therefore created such that they covered various starting times in 2-hour increments. The comparison was then based on all the prediction cases with different starting times.

The cumulative distribution function of the RMSRE for each model is shown in Fig. 4. The cumulative distribution function provides a visual comparison of each prediction model—which shows the Holt-Winters prediction is clearly out performed by the others.

The frequency distribution of the prediction error of these models as observed does not follow a normal distribution, as there are high-valued outliers. Anderson-Darling normality test also confirms that the distribution is not normal with a 99.9% confidence level. In this paper, our focus is to compare the prediction accuracy in general, rather than on a case-by-case basis; therefore, the comparison is based on median and interquartile range (IQR) rather than average and standard deviation, providing a more robust measurement in the presence of outliers. The summary of prediction error for 1-day forecast horizons is shown in Table V. Side-by-side comparisons using paired Wilcoxon tests were performed to determine whether the median difference is statistically significant. Using 99.9% confidence level, the tests show the ordering is as follows:

LP-SegAvgES < ARIMA < ES < SARIMA < HW.

Another method of evaluation is to plot the median prediction error at various starting times, which are shown in Fig. 5. The median prediction error of ES is the most affected by the choice of starting time, while LP-SegAvgES produces considerably more consistent results for different starting times.

N-Step Ahead Performance Estimation

In order to investigate the prediction error of each model for various forecast horizons, seven sets of prediction horizons were created. Every set represents a different forecast horizon starting from 1-day up to 7-day prediction. All of these cases use 3 days of training data.

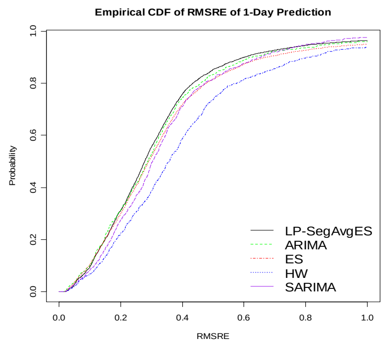

Figure 4: Empirical CDF based on 1-day prediction of LP-SegAvgES, ARIMA, ES, SARIMA, and HW.

TABLE V.
BASIC STATISTICS OF PREDICTION ERROR OF 1-DAY PREDICTION.

	LP-SegAvgES	**ARIMA**	**ES**	**SARIMA**	**HW**
Median	0.278	0.288	0.291	0.303	0.358
IQR	0.224	0.242	0.263	0.238	0.294

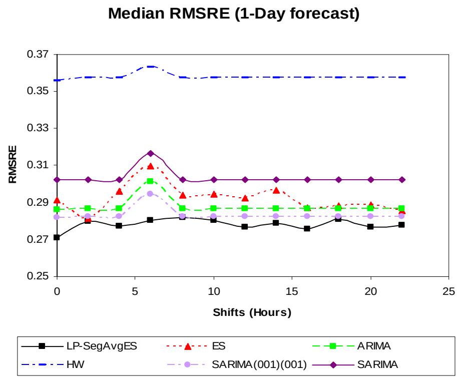

Figure 5: The effect of starting time (relative to 00:00 +10 GMT) to the median forecast error, based on 1-day forecast horizons.

The result is summarized in the median RMSRE prediction, which is shown in Fig. 6. While it is expected that the center measure and the spread of prediction error grows as the forecast horizon gets longer, it is shown that LP-SegAvgES produces consistently low errors throughout different forecast horizons when compared to other models. To confirm this observation, the paired Wilcoxon tests, with a 99.9% confidence level is run for each forecast horizon. The result is shown in Table VI.

SARIMA prediction accuracy degrades more quickly when compared to other models as the forecast horizon increases.

This effect is most likely caused by choosing the SARIMA model that accounts for linear trend. The best-fit SARIMA in this experiment shows about 81% of the prediction cases selected SARIMA (0,1,1)(0,1,1), which accounts for linear trend. The prediction error becomes more apparent in a longer forecast horizon, as the prediction drifts away from the effective performance range. Forcing the forecast to use SARIMA (0,0,1)(0,0,1) which does not consider linear trend on both seasonal and nonseasonal components, improves the medium-term forecasting, as shown in Fig. 6. This validates our initial observation of the dataset in which the linear trend does not exist.

In general, the Holt-Winters prediction gives the poorest prediction accuracy among all of the models, caused by a problem shared with the best-fit SARIMA model: the prediction falls outside the effective performance range. Referring back to Section IV, the Holt-Winters prediction suffers when the seasonality is not strong, and is potentially prone to large bias when the recent training data is shifted.

VI. DISCUSSION

In Section III it is shown that the real datasets exhibit seasonality characteristics. Although the existing seasonal models (SARIMA and Holt-Winters) theoretically are suited to model such seasonal characteristics, our evaluation results show high prediction errors as compared to the nonseasonal models. One possible explanation is that the seasonality in the real datasets is not consistently strong. In such cases, seasonal models assume the recurring pattern is consistently strong, therefore produces bias, which will be more apparent for longer forecast horizons.
SARIMA in the evaluation is chosen such that the parameters produce the lowest AIC criteria. As described by [34], there is no guarantee that a good-fit model will always produce good forecasts; and this case is shown in the SARIMA prediction in the previous section. The best-fit SARIMA in the evaluation tends to select the SARIMA (0,1,1)(0,1,1) model over other models. While Section shows there is no persistent trend in the data traces we use, this model may identify a false trend due to the temporal disruption, leading to out-of-bound performance prediction. On the other hand, SARIMA (0,0,1)(0,0,1) produces high AIC and is less likely to be selected as the best-fit model. This model, however, produces lower prediction error than the best-fit SARIMA, as well as the Holt-Winters. For one day forecast horizon, SARIMA (0,0,1)(0,0,1) also produces significantly lower prediction error compared to exponential smoothing.

Nonseasonal models in general produce lower prediction error compared to seasonal models, despite their inability to capture seasonal properties. This may well indicate that application of ARIMA or exponential smoothing is adequate to perform medium-term forecasting in general. Among all models, ARIMA produce lowest prediction error by number. Exponential smoothing may produce slightly higher prediction error (for forecast horizon of one to five days) than ARIMA model, however the difference is not significant for forecast beyond five days. In practice,

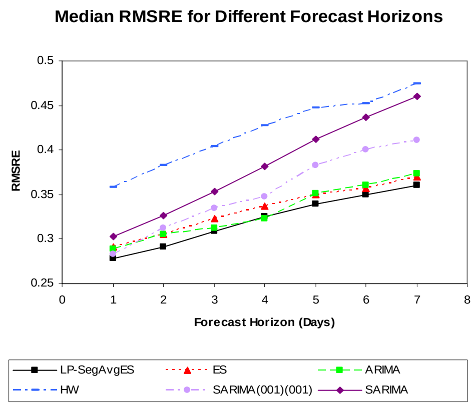

Figure 6: Median RMSRE of 1- to 7-day forecast horizons of various prediction models.

TABLE VI.
MODEL ORDERING BASED ON RMSRE BY PAIRED WILCOXON TESTS.

Forecast Horizon	Ordering
1 day	LP-SegAvgES < ARIMA,SARIMA(001)(001) < ES < SARIMA < HW
2 days	LP-SegAvgES < ARIMA < SARIMA(001)(001),ES < SARIMA < HW
3 days	LP-SegAvgES < ARIMA < SARIMA(001)(001),ES < SARIMA < HW
4 days	LP-SegAvgES < ARIMA < ES < SARIMA(001)(001) < SARIMA < HW
5 days	LP-SegAvgES < ARIMA,ES < SARIMA(001)(001) < SARIMA < HW
6 days	LP-SegAvgES < ARIMA < ES < SARIMA(001)(001) < SARIMA < HW
7 days	LP-SegAvgES < ARIMA,ES < SARIMA(001)(001) < SARIMA < HW

exponential smoothing may be more preferred for its simplicity.

For medium-term forecasting, the LPP method produces the lowest prediction error in this evaluation. While this model is able to capture seasonality characteristics, unlike the existing seasonal models, it does not suffer from the forecast bias for long forecast horizons. In general, LPP method is suited for the datasets which exhibit weak to medium seasonality, and which were dominant in the client-perceived performance. LPP method is also capable of reflecting the seasonality in the prediction, which can be used to identify the peak/off-peak period in the forecast. This allows a more tailored planning or recommendation, based on the time and period of usage.

VII. CONCLUSION

Popular time series approaches were evaluated for client-perceived performance prediction: ARIMA and exponential smoothing, as well as their seasonal counterparts, seasonal ARIMA and Holt-Winters. Seasonal models were found to produce higher prediction error compared to the nonseasonal models, despite the evidence of seasonality. The most probable causes are that the seasonality of the client-perceived performance is not strong, and the presence of temporal disruptions. The evaluation also found that nonseasonal models produce lower prediction error than the seasonal models, which suggests their suitability for client-perceived performance prediction in general. These nonseasonal models however, do not have the ability to reflect the seasonal behavior in the forecast.

From real life experiments, the LPP method has shown its ability to reflect seasonal pattern, as well as produce significantly lower prediction errors in medium-term forecasting as compared to the existing models evaluated. Compared with other existing time series models, this scheme produces more consistent results in cases involving different starting time selection and longer forecast horizons (up to 7 days). The success of this method for medium-term prediction is best explained through how it captures the main characteristics of the client-perceived performance: firstly, the seasonality does not need to be strong, and secondly, the mean and variance may vary over time, but the overall fluctuation is still bounded within a finite range.

VIII. FUTURE WORK

In this section we briefly describe the provider recommendation framework we developed, named **P**rovider **R**ecommendation based **O**n client-**P**erceived **PER**formance (PROPPER). The main interaction is shown in Fig. 7. In this framework, client-perceived performance is measured automatically at the client's machine, by using middleware library modification approach [23]. The performance measurement log is then submitted to a *recommender* periodically by the background process. In this framework, the recommender will receive performance measurement logs from the registered clients.

The recommender performs two main tasks as shown in Fig. 8: *analysis* and *recommendation*. Analysis is an ongoing process, which collects and analyses performance measurements submitted by clients. This process produces a summary of client performance over time. In contrast, the recommendation process is an on-demand process, which is only triggered when a client queries the recommender system. The main functionality of this process is to respond to a client inquiry with a list of suitable providers, based on predicted performance as determined using historical data from the analysis process.

The client that seeking for provider recommendation can specify the expected service usage duration (e.g. one, or two days), and the expected usage time interval (e.g. the local peak or off-peak time). The recommendation process compares the suitable providers based on the predicted

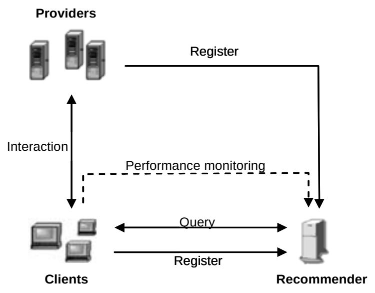

Figure 7: The main interaction within PROPPER framework.

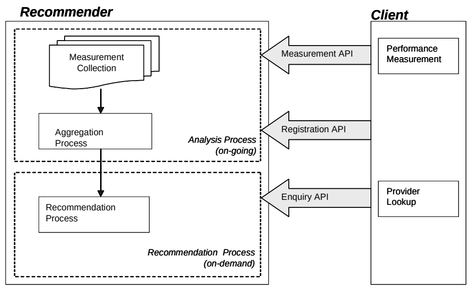

Figure 8: The main tasks of recommender in PROPPER framework.

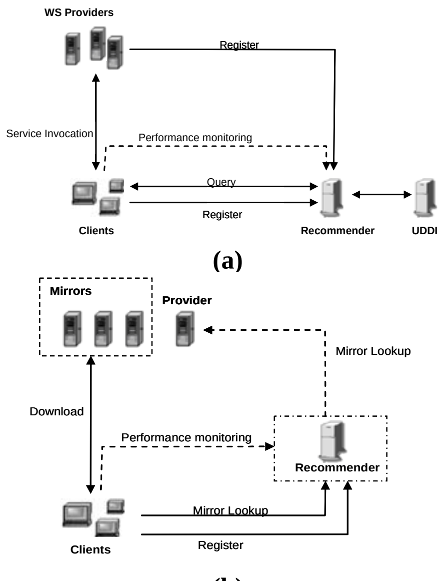

Figure 9: Application of PROPPER framework within WS context (a), and mirror selection context (b).

performance (calculated by using the medium-term prediction method discussed in this paper) which also accounts the expected service usage duration and the expected usage time interval specified by the client. From this point it is evident that the prediction aspect is crucial within the recommendation process, as it determines the effectiveness of the recommendation outcome.

PROPPER framework can be applied within the WS or mirror selection context, as illustrated in Fig. 9. Our future publication will discuss the approach to address sparseness in the measurement series through performance estimation and imputation, and to conduct comparative study against the existing provider recommendation frameworks, such as geographical distance-based or country based recommendation systems.

IX. REFERENCES

[1] E. W. Zegura, M. H. Ammar, Z. Fei, S. Bhattacharjee, and "Application-layer anycasting: a server selection architecture and use in a replicated Web service," *IEEE/ACM Transactions on Networking,* vol. 8, pp. 455-466, August 2000 1997.

[2] C. Facciorusso, S. Field, R. Hauser, Y. Hoffner, R. Humbel, R. Pawlitzek, W. Rjaibi, and C. Siminitz, "A Web Services Matchmaking Engine for Web Services," *E-Commerce and Web Technologies, Proceedings Lecture Notes in Computer Science,* 2003.

[3] J. Gonzalez-Castillo, D. Trastour, and C. Bartolini, "Description Logics for Matchmaking of Services," Trusted E-Services Laboratory, HP Laboratories Bristol (Technical Report HPL-2001-265), 2001.

[4] S. Ran, "A Model for Web Services Discovery with QoS," *ACM SIGecom Exchanges,* vol. 4, pp. 1-10, 2003.

[5] A. Dan, D. Davis, R. Kearney, A. Keller, R. King, D. Kuebler, H. Ludwig, M. Polan, M. Spreitzer, and A. Youssef, "Web Services on demand: WSLA-driven Automated Management," *IBM Systems Journal,* vol. 43, pp. 136-157, 2004.

[6] A. Keller and H. Ludwig, "The WSLA Framework: Specifying and Monitoring Service Level Agreements for Web Services," IBM Research Division 22 May 2002.

[7] A. Sahai, V. Machiraju, M. Sayal, L. J. Jin, and F. Casati, "Automated SLA Monitoring for Web Services," HP Laboratories.

[8] N. Gibelin and M. Makpangou, "Efficient and Transparent Web-Services Selection " in *Service-Oriented Computing - ICSOC 2005*. vol. Volume 3826/2005: Springer Berlin / Heidelberg, 2005, pp. 527-532.

[9] A. Shaikhali, O. F. Rana, R. Al-Ali, and D. W. Walker, "UDDIe: An Extended Registry for Web Services," *Proceedings of the Service Oriented Computing: Models, Architectures and Applications,* 2003.

[10] S. Seshan, M. Stemm, and R. H. Katz, "SPAND: shared passive network performance discovery," *Proceedings of the USENIX Symposium on Internet Technologies and Systems,* pp. 135-146, 1997 1997.

[11] L.-T. C. Chen Zhou, Bilhanan Silverajan, Bu-Sung Lee, "UX- An Architecture Providing QoS-Aware and Federated Support for UDDI.," in *Proceedings of the International Conference on Web Services, ICWS '03,* Las Vegas, Nevada, USA, 2003.

[12] J. Wei and C.-Z. Xu, "eQoS: Provisioning of Client-Perceived End-to-End QoS Guarantees in Web Servers," *IEEE Transactions on Computers,* vol. 55, pp. 1543-1556, 2006.

[13] C.-Z. Xu, B. Liu, and J. Wei, "Model Predictive Feedback Control for QoS Assurance in Webservers," in *Computer*. vol. 41, 2008, pp. 66-72.

[14] H. Sun, B. Fang, and H. Zhang, "User-Perceived Web QoS Measurement and Evaluation System," in *Frontiers of WWW Research and Development - APWeb 2006*. vol. 3841/2006: Springer Berlin / Heidelberg, 2006, pp. 157-165.

[15] C. Zhou, L.-T. Chia, B. Silverajan, and B.-S. Lee, "UX- An Architecture Providing QoS-Aware and Federated Support for UDDI," in *Proceedings of the International Conference on Web Services, ICWS '03*, Las Vegas, Nevada, USA, 2003, pp. 171-176.

[16] E. M. Maximilien and P. S. Munindar, "Toward autonomic web services trust and selection," in *Proceedings of the 2nd international conference on Service oriented computing* New York, NY, USA: ACM, 2004.

[17] O. Martín-Díaz, A. Ruiz-Cortés, A. Durán, and C. Müller, "An Approach to Temporal-Aware Procurement of Web Services," in *Service-Oriented Computing - ICSOC 2005*. vol. 3826/2005: Springer Berlin / Heidelberg, 2005, pp. 170-184.

[18] R. Wolski, N. Spring, and C. Peterson, "Implementing a performance forecasting system for metacomputing: the Network Weather Service," in *Proceedings of the 1997 ACM/IEEE conference on Supercomputing (CDROM)*, San Jose, CA 1997, pp. 1-19.

[19] S. Vazhkudai, J. M. Schopf, and I. Foster, "Predicting the performance of wide area data transfers," in *16th International Parallel and Distributed Processing Symposium (IPDPS 2002)*, Ft. Lauderdale, FL, USA, 2002, pp. 34-43.

[20] N. K. Groschwitz and G. C. Polyzos, "A time series model of long-term NSFNET backbone traffic," in *IEEE International Conference on Communications 1994*, New Orleans, LA 1994, pp. 1400 - 1404

[21] S. Basu, A. Mukherjee, and S. Klivansky, "Time Series Models for Internet Traffic," Georgia Institute of Technology GIT-CC-95-27, 1996.

[22] J. D. Brutlag, "Aberrant behaviour detection in time series for network monitoring," in *LISA '00: Proceedings of the 14th USENIX conference on System administration*, New Orleans, Louisiana, 2000, pp. 139-146.

[23] N. Thio and S. Karunasekera, "Automatic Measurement of a QoS Metric for Web Service Recommendation," in *Australian Software Engineering Conference*, Brisbane, Australia, 2005, p. 202.

[24] R. J. A. Little and D. B. Rubin, *Statistical analysis with missing data* 2ed. Hoboken, N.J.: John Wiley & Sons, 2002.

[25] P. C. B. Phillips and P. Perron, "Testing For A Unit-Root In Time-Series Regression," *Biometrika,* vol. 75, pp. 335-346, Jun 1988.

[26] G. E. P. Box, G. C. Reinsel, and G. N. Jenkins, *Time series analysis : forecasting and control* 3ed. Englewood Cliffs, N.J.: Prentice Hall, 1994.

[27] M. S. Bartlett, *An Introduction to Stochastic Processes with Special Reference to Methods and Applications*, 2 ed. Cambridge: Cambridge University Press, 1966.

[28] R. E. G. Upshur, R. Moineddin, E. J. Crighton, and M. Mamdani, "Is there a clinically significant seasonal component to hospital admissions for atrial fibrillation?," *Bmc Health Services Research,* vol. 4, Mar 19 2004.

[29] L. J. Heyer, S. Kruglyak, and S. Yooseph, "Exploring Expression Data: Identification and Analysis of Coexpressed

Genes," *Genome Research*, vol. 9, pp. 1106-1115, November 1999 1999.

[30] M. Harchol-Balter and A. B. Downey, "Exploiting process lifetime distributions for dynamic load balancing," *ACM Transactions on Computer Systems (TOCS)*, vol. 15, pp. 253-285, 1997.

[31] G. Valery and S. Jaideep, "Event detection from time series data," in *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining* San Diego, California, United States: ACM Press, 1999.

[32] L. R. Rabiner, "A Tutorial On Hidden Markov-Models And Selected Applications In Speech Recognition," *Proceedings Of The Ieee*, vol. 77, pp. 257-286, Feb 1989.

[33] S. D. Gribble and E. A. Brewer, "System design issues for internet middleware services: deductions from a large client trace," in *USITS'97: Proceedings of the USENIX Symposium on Internet Technologies and Systems on USENIX Symposium on Internet Technologies and Systems*, Monterey, California, 1997, p. 19.

[34] S. G. Makridakis, S. C. Wheelwright, and R. J. Hyndman, *Forecasting : methods and applications*, 3 ed. New York: John Wiley & Sons, 1998.

Niko Thio received the B.Sc. degree in electrical engineering from Gadjah Mada University, Indonesia, in 1998, and the M.S. degree in information technology from Swinburne University of Technology, Australia, in 2002.

He is currently working towards the Ph.D. degree at University of Melbourne, addressing quality of service of client-server internet applications.

Shanika Karunasekera received the B.Sc degree in electronics and telecommunications engineering from the University of Moratuwa, Sri Lanka, in 1990, and the Ph.D. degree in electrical engineering from the University of Cambridge, UK, in 1995.

From 1995 to 2002, she was a Software Engineer and a Distinguished Member of Technical Staff at Lucent Technologies, Bell Labs Innovations, USA. In January 2003, she joined the Department of Computer Science and Software Engineering, University of Melbourne as a Senior Lecturer. Her current research interests distributed computing, software engineering and peer-to-peer computing.

Call for Papers and Special Issues

Aims and Scope

Journal of Emerging Technologies in Web Intelligence (JETWI, ISSN 1798-0461) is a peer reviewed and indexed international journal, aims at gathering the latest advances of various topics in web intelligence and reporting how organizations can gain competitive advantages by applying the different emergent techniques in the real-world scenarios. Papers and studies which couple the intelligence techniques and theories with specific web technology problems are mainly targeted. Survey and tutorial articles that emphasize the research and application of web intelligence in a particular domain are also welcomed. These areas include, but are not limited to, the following:

- Web 3.0
- Enterprise Mashup
- Ambient Intelligence (AmI)
- Situational Applications
- Emerging Web-based Systems
- Ambient Awareness
- Ambient and Ubiquitous Learning
- Ambient Assisted Living
- Telepresence
- Lifelong Integrated Learning
- Smart Environments
- Web 2.0 and Social intelligence
- Context Aware Ubiquitous Computing
- Intelligent Brokers and Mediators
- Web Mining and Farming
- Wisdom Web
- Web Security
- Web Information Filtering and Access Control Models
- Web Services and Semantic Web
- Human-Web Interaction
- Web Technologies and Protocols
- Web Agents and Agent-based Systems
- Agent Self-organization, Learning, and Adaptation
- Agent-based Knowledge Discovery
- Agent-mediated Markets
- Knowledge Grid and Grid intelligence
- Knowledge Management, Networks, and Communities
- Agent Infrastructure and Architecture
- Agent-mediated Markets
- Cooperative Problem Solving
- Distributed Intelligence and Emergent Behavior
- Information Ecology
- Mediators and Middlewares
- Granular Computing for the Web
- Ontology Engineering
- Personalization Techniques
- Semantic Web
- Web based Support Systems
- Web based Information Retrieval Support Systems
- Web Services, Services Discovery & Composition
- Ubiquitous Imaging and Multimedia
- Wearable, Wireless and Mobile e-interfacing
- E-Applications
- Cloud Computing
- Web-Oriented Architectrues

Special Issue Guidelines

Special issues feature specifically aimed and targeted topics of interest contributed by authors responding to a particular Call for Papers or by invitation, edited by guest editor(s). We encourage you to submit proposals for creating special issues in areas that are of interest to the Journal. Preference will be given to proposals that cover some unique aspect of the technology and ones that include subjects that are timely and useful to the readers of the Journal. A Special Issue is typically made of 10 to 15 papers, with each paper 8 to 12 pages of length.

The following information should be included as part of the proposal:

- Proposed title for the Special Issue
- Description of the topic area to be focused upon and justification
- Review process for the selection and rejection of papers.
- Name, contact, position, affiliation, and biography of the Guest Editor(s)
- List of potential reviewers
- Potential authors to the issue
- Tentative time-table for the call for papers and reviews

If a proposal is accepted, the guest editor will be responsible for:

- Preparing the “Call for Papers” to be included on the Journal’s Web site.
- Distribution of the Call for Papers broadly to various mailing lists and sites.
- Getting submissions, arranging review process, making decisions, and carrying out all correspondence with the authors. Authors should be informed the Instructions for Authors.
- Providing us the completed and approved final versions of the papers formatted in the Journal’s style, together with all authors’ contact information.
- Writing a one- or two-page introductory editorial to be published in the Special Issue.

Special Issue for a Conference/Workshop

A special issue for a Conference/Workshop is usually released in association with the committee members of the Conference/Workshop like general chairs and/or program chairs who are appointed as the Guest Editors of the Special Issue. Special Issue for a Conference/Workshop is typically made of 10 to 15 papers, with each paper 8 to 12 pages of length.

Guest Editors are involved in the following steps in guest-editing a Special Issue based on a Conference/Workshop:

- Selecting a Title for the Special Issue, e.g. “Special Issue: Selected Best Papers of XYZ Conference”.
- Sending us a formal “Letter of Intent” for the Special Issue.
- Creating a “Call for Papers” for the Special Issue, posting it on the conference web site, and publicizing it to the conference attendees. Information about the Journal and Academy Publisher can be included in the Call for Papers.
- Establishing criteria for paper selection/rejections. The papers can be nominated based on multiple criteria, e.g. rank in review process plus the evaluation from the Session Chairs and the feedback from the Conference attendees.
- Selecting and inviting submissions, arranging review process, making decisions, and carrying out all correspondence with the authors. Authors should be informed the Author Instructions. Usually, the Proceedings manuscripts should be expanded and enhanced.
- Providing us the completed and approved final versions of the papers formatted in the Journal’s style, together with all authors’ contact information.
- Writing a one- or two-page introductory editorial to be published in the Special Issue.

More information is available on the web site at http://www.academypublisher.com/jetwi/.