Storm Prediction in a Cloud

Ian Davis, Hadi Hemmati, Ric Holt, Mike Godfrey
David R. Cheriton School of Computer Science
University of Waterloo
Waterloo, Ontario, Canada N2L 3G1
{ijdavis, hhemmati, holt, migod}@uwaterloo.ca

Douglas Neuse, Serge Mankovskii
CA Labs, CA Technologies
{Douglas.Neuse, Serge.Mankovskii}@ca.com

Abstract — **Predicting future behavior reliably and efficiently is key for systems that manage virtual services; such systems must be able to balance loads within a cloud environment to ensure that service level agreements (SLAs) are met at a reasonable expense, and they must be able to facilitate longer term reconfiguration of the system to support future anticipated demand. In principle accurate predictions can be achieved by mining a variety of data sources, which describe the historic behavior of the services, the requirements of the programs running on them, and the evolving demands placed on the cloud by end users. Of particular importance is accurate prediction of maximal loads likely to be observed in the short term, and the accurate estimation of long term peak behavioral trends. However, standard approaches to modeling system behavior, by analyzing the totality of the observed data, tend to predict average rather than exceptional computer behavior and ignore important patterns of change over time. Consequently, these predictions are of limited use in providing warnings of future "storms" within a cloud environment. In this paper, we present several mechanisms for improving such predictions, derived from standard prediction techniques, which we have found more accurately predict these storms.**

Index Terms— **Regression, time-series, prediction, cloud environments**

I. INTRODUCTION

Large collections of computers cooperatively providing distributed services so as to support demand for these services are becoming the norm. Single computers hosting multiple virtual machines are now also common. And parallel computing techniques offer the potential for very time consuming computation to be distributed across multiple machines, thus delivering results from a single computation to the consumer much more rapidly. In each case decoupling of computer software from the underlying hardware, permits greater utilization of the hardware, under more balanced workloads, with improvements in response times, and dramatic reduction in costs. In addition the ability to replicate services over multiple machines permits greater scalability, combined with more robustness than would otherwise be possible.

However, if the demands placed on the hardware by the software exceed the capabilities of the hardware, thrashing will occur, response time's rise, and customer satisfaction will plummet. Therefore it is essential [23] to ensure that software is appropriately provisioned across the available hardware in the short to medium term, and that appropriate levels of hardware and software are purchased and installed to support the collective future end user requirements in the longer term [11, 17, 22].

Without good forecasts, data center managers are forced to over-configure their pools of resources to achieve required availability, in order to honor service level agreements (SLAs). This is expensive, and can still fail to consistently satisfy SLAs. Absent good forecasts, system management software tends to operate in a reactive mode and can become ineffective and even disruptive [18]. In this paper we present modifications to standard prediction techniques, which are better able to predict when utilization of a service will be high. Knowing this it becomes possible to predict when storms will occur.

II. MOTIVATION

This research is motivated by CA Technologies [3] need to develop industrial solutions for effectively predicting workload of their clients' cloud services, both in the short term (measured in hours), and in the longer term (measured in months). Good short term predictions permit better adaptive job scheduling, while longer term predictions permit appropriate and financially effective proactive provisioning of resources within the cloud environment.

We were provided with a substantive body of performance data relating to a single large cloud computing environment running a large number of virtual services over a six month period. In total there were 2,133 independent entities whose performance was being captured every six minutes. These included 1,572 virtual machines and 495 physical machines. The physical machines provided support for 56 VMware hosts. On average 53% of the monitored services were active at any time, with a maximum of 85% (Fig. 1). The data captured described CPU workloads, memory usage, disk I/O and network traffic. This data was each consolidated into average and maximum hourly performance figures, and it was the hourly CPU workload data that our research was predicated on.

At least 423 services were dedicated to providing virtual desktop environments, while the cloud was also proving support for web based services, transaction processing,

database support, placement of virtual services on hosts, and other services such as performance monitoring and backup.

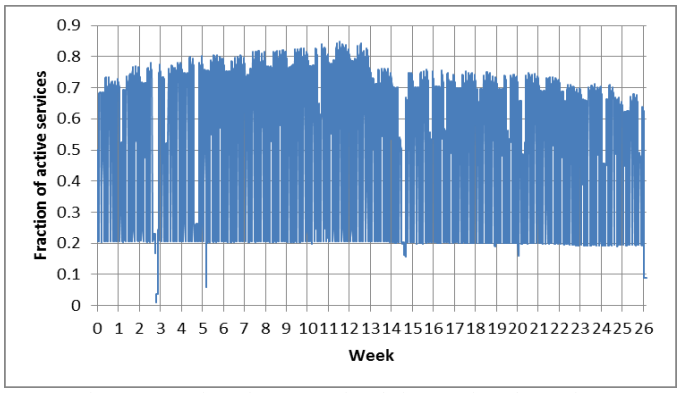

Figure 1. Fraction of 2,133 services being monitored over time

As is typically the case in desktop environments, individual computer utilization varies dramatically. Much of the time little if any intensive work is being done on a virtual desktop and the virtual service appears almost idle. However, for any virtual desktop there are periods of intense activity, when CPU, memory, disk I/O, and/or network traffic peaks. Similarly within transaction processing environments, there will be a wide variety of behaviors, depending on the overall demand placed on such systems.

The average utilization across all services was at most 20% (except for the weekly backup when average utilization rose to almost 50%), but the maximum utilization across all services, was almost invariably very close to 100% (Fig. 2).

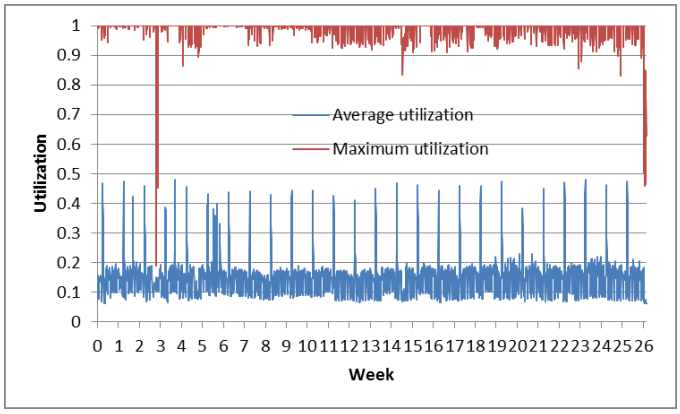

Figure 2. Cloud utilization over time

The frequency distribution of the utilizations actually observed, was highly skewed, with the vast majority of utilizations (83.5%) not exceeding 25%. The exponential utilization curve was approximated by $100*(2-u)^{13.5}$ (Fig. 3).

To explore what trends [24] exist within peak usage, we collapsed the time series by deleting all hours where utilization was less than 25%. Then, within the shortened time series, we computed separately for each virtual service the trend line (as the standard linear regression slope [6]) for the training period (first half of the data), and separately the testing period (second half of the data). Finally we looked for change in the trend line between these two periods for each virtual service (Fig. 4).

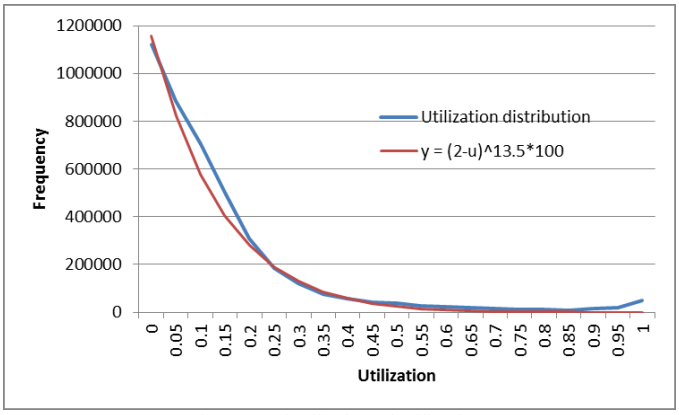

Figure 3. Distribution of utilizations

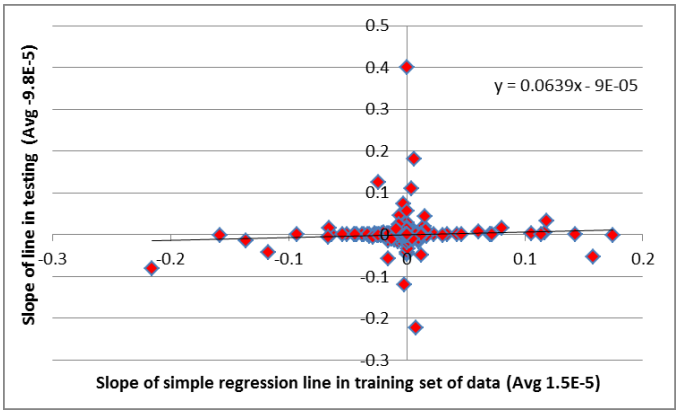

Figure 4. Linear trend within data sources

The majority of the data sources had no conspicuous trend either during the testing or the training period, and the overall trend line (y = 0.0639x - 0.00009) within the scatter plot was itself almost flat, suggesting that trend was more conspicuous in the training period than in the testing period. The lack of trend is perhaps not as surprising as it might at first appear, since while one might hope for increased utilization of business software over time, it is to be expected that individual usage of virtual desktops will not change radically over time.

Given this overall picture, our challenge was to resolve how one might reasonably predict in the short term (next hour) when a given service was likely to be heavily utilized, so as to permit it to be provided with necessary resources, while also avoiding conflicts between multiple heavily utilized activities, or the resources they depended on, in an environment where typically services would not be expected to be heavily utilized.

In this context, accurate predictions that an idle service will remain idle most of the time, while very occasionally as consequence being wrong, are useless. What are required are predictions which (with a reasonable degree of confidence) indicate when future loads will be high, even if such

predictions do not mathematically fit the totality of observed and future data as closely as other statistical approaches.

Over a longer time frame (measured in weeks or months) we wished to be able to accurately predict expected workloads, so that a cloud data center could be appropriately managed. Without such long term predictions it becomes difficult to ensure in a timely fashion that a computing center is provided with adequate power and hardware, and to predict what the future cost of operating the facilities will be.

III. RELATED WORK

Andreolini et. al. proposed using moving averages to smooth the input time series, and then using linear extrapolation on two of the smoothed values to predict the next [1]. Dinda et. al. compared the ability of a variety of ARIMA like models to predict futures [5]. Nathuji et. al. proposed evaluating virtual machines in isolation, and then predicted their behavior when run together using multiple input signals to produce multiple predictive outputs using difference equations (exponential smoothing) [17].

Istin et. al. divided the time series into multiple series using different sampling steps, obtained predictions from each subseries, and then used neural networks to derive a final prediction [14]. Khan et. al. also used multiple time series (but derived from distinct workloads) and hidden markov models, to discover correlations between workloads which can then be used to predict variations in workload patterns [15].

Huhtal et. al. proposed that similarity between related time series might be discovered by using wavelets [12]. Ganapathi discovered linear mappings from related information (such as workload and performance data), which produced maximum correlations between this distinct data, and then exploited the discovered similarities between the input data, before recovering actual predictions through inverting these transformations [10].

Povinelli proposed mapping data points at fixed temporal lags into n-dimensional phase space, clustering the resulting points on their Euclidian distance, and then employing a user specified goal function for each point within a cluster to discover (using genetic programming techniques) unknown patterns/clusters strongly predictive of future events deemed interesting [20]. Srinivasan et. al. also used genetic algorithms, but translated the time series into substrings, with symbols within the string representing the various types of behavior within the time series [21].

More generally, Nikolov proposed that performance might be understood as consequence of given underlying patterns that determined behavior, and clustered observed behavior according to the pattern it was most similar to, using a simple distance metric. Behavior might then be predicted, because of the discovered similarity with an understood behavioral pattern [19]. Magnusson proposed that simple underlying short term patterns be detected and that larger longer term composite patterns then be discovered through a hierarchical composition of these simpler patterns recursively [16].

IV. LINEAR REGRESSION

We focused on predicting physical and virtual CPU utilizations. For each data series, the observed utilizations were partitioned into small intervals in increments of 0.05, and for each such partition the average absolute difference between observed and predicted values were obtained. This provides a clear picture of how closely prediction matches observed utilization across the spectrum. Then these averages are themselves averaged across the set of data series.

Unchanged: Since there was at least a 30% probability that utilizations would remain essentially unchanged from one hour to the next (Fig. 5), as a trivial baseline measurement we predicted that there would never be change in the utilization observed during the previous hour. As expected this approach did not perform as well as any of the other approaches (Fig. 8).

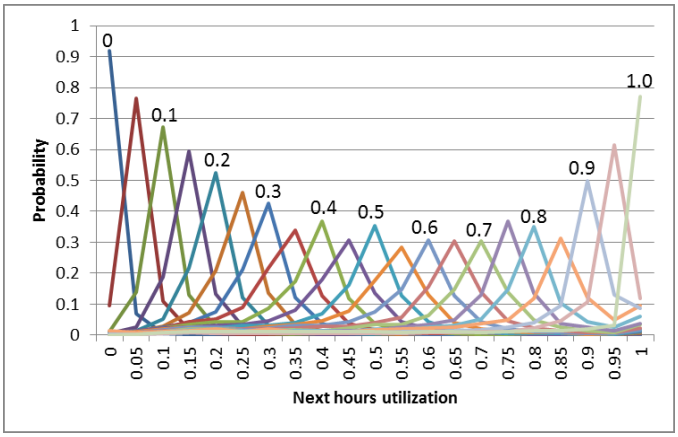

Figure 5. Next hour's utilization distribution given this hour's value

We then obtained correlograms from the provided data, by computing the auto-correlation of each time series with each lagged version of the same time series (Fig. 6). This indicated the strongest auto-correlation was at the hourly (1,676 sources), weekly (247), daily (106) and bi-weekly (41) levels, with these correlations degraded only slowly over longer intervals.

MVLR: Using the discovered significant lags, multivariate linear regression [6] was then applied using 10 lags of 1 and 2 hour, 1 and 2 day, 1, 2, 3 and 4 week, and 1 and 2 months, to identify coefficients which when applied to this strongly correlated lagged data, linearly fit observed data, with minimal least squared residue error. This provided good general predictability across the data sources. The resulting linear equation was then used to predict the next hour's utilization.

Two minor problems required special consideration. The first was that the provided time series data contained missing

data. Short gaps were approximated by prior value, while 769 time series having more missing data than actual data were discarded as they might otherwise have unreasonably skewed our results. The second was that due to the difficulties associated with monitoring the utilization of virtual processes 1% of the data points exceeded their expected upper bound of 1. This was attributed to Intel Turbo Boost/up clocking being enabled and was resolved by reducing excessive values to 1.

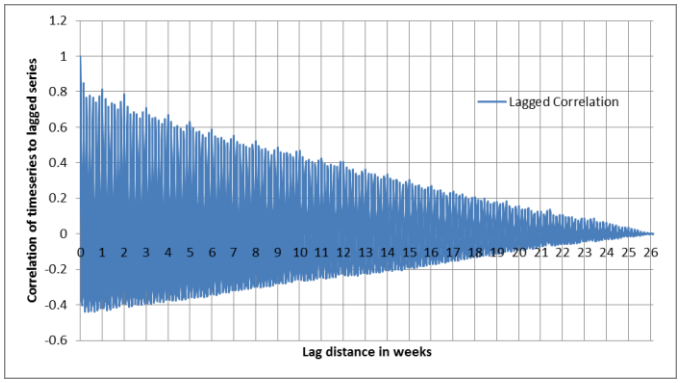

Figure 6. Example correlogram for one timeseries

While visually predictions appeared to fit observed data reasonably well (Fig. 7), close examination revealed that often sudden rises in observed utilizations were not anticipated by the regression.

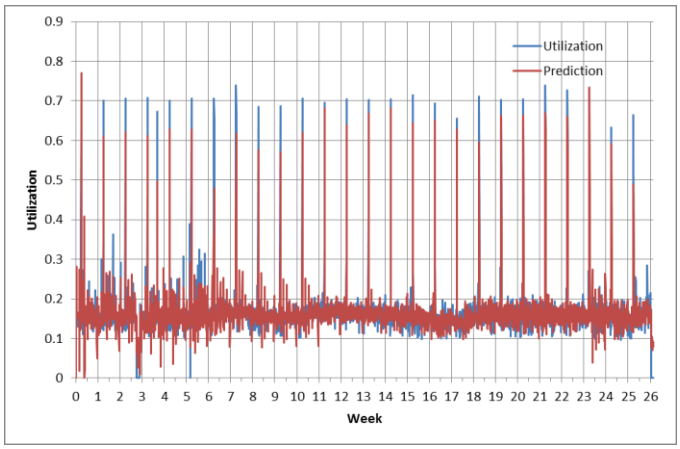

Figure 7. Example Multivariate Linear Regression

Instead of predicting such events, the regression exploited observed peaks in the data to produce higher predicted utilizations one hour later [13]. The regression was thus not predicting as yet unobserved trends. In addition, because the regression was linearly fitting to least squared residues of all the data points, and the vast majority of data points were associated with idle time, it could not hope to fit well to maximal values occurring within the time series storms.

Windowed MVLR: We next restricted the multivariate linear regression to employ a 5 week window, with some resulting improvement (Fig. 8).

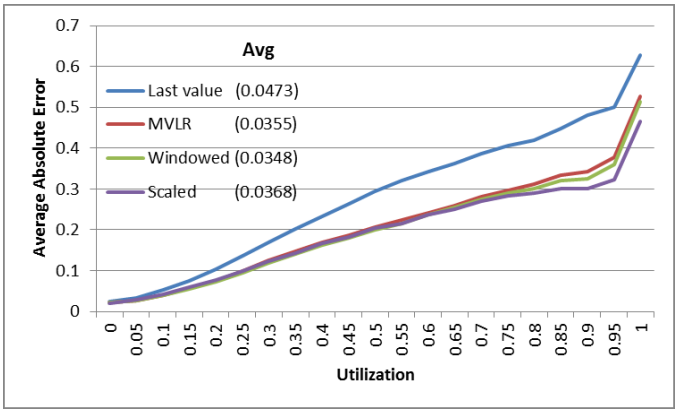

Figure 8. Multivariate Linear Regression

Scaled MVLR: We next computed the running average and variance in the data seen, and scaling the predicted values to have the same average and variance as the till then observed data. This improved the fit to high utilizations, at very small expense of fit to small utilization values (Fig. 8).

Weighted MVLR: Within the regression summations we then weighted [8] each data point. Because the overall distribution of utilizations was observed to be exponential (Fig. 3), we employed exponential weighting in which a data point having utilization u as well as all lags associated with this data point were multiplied by $(1+u)^c$. This naturally assigned higher utilizations a significantly greater weight, thus skewing the predictions towards higher values, while simultaneously bounding them by the highest values (Fig. 9). As can be seen a value of c=12 provided the most consistent average absolute error across all of the data sources.

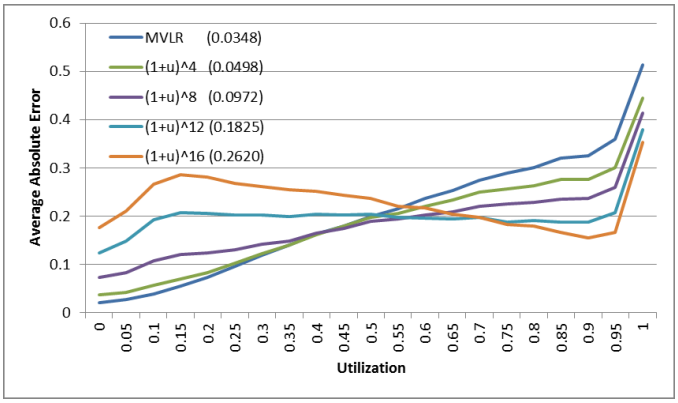

Figure 9. Weighting Regression by $(1+u)^c$

Power MVLR: We then presented as input values to the multivariate linear regression u^c, rather than u. The regression subsequently used these modified input values in all internal computations. The final prediction obtained from this modified regression, was then recovered by taking the c^{th} root of the value obtained by the regression.

As an informal justification for this approach, consider a set of non-negative points a_i with $1 \le i \le n$. The average of this set of

points is $\Sigma a_i/n$. Applying the above transformation for c=2, gives a revised average value of $\sqrt{(\Sigma a_i^2/n)}$. The difference of the square of the revised value to the square of the actual average is simply the variance in the sample.

Since the variance cannot be negative, and neither can our summations, this implies that the revised average can be no smaller than the original average, and can agree only when the variance in the original sample is 0. Further, the revised value must remain less than or equal to the maximal original point, with equality reached again, only if the variance in the original sample is 0.

This argument can be used repeatedly for $c=2^k$ (Table 1), suggesting that the shift towards maximal values in the set examined, is a monotonically increasing function, for increasing values of k. The proposed transformation has minimal impact on collections of small values, but for large variations, the shift upwards is quite dramatic (Fig.10), and does provide a much better fit to high observed utilizations, by shifting predictions upwards, while producing a very poor fit to the majority of observed low observed utilizations.

Table 1. Examples of adjustment using powers

Sample Values	Power of the input				
	c=1	c=2	c=10	c=20	c=30
{0,0.05,0.1}	0.05	0.0645	0.0896	0.0947	0.0964
{0,0.1,0.2}	0.10	0.1291	0.1792	0.1893	0.1928
{0,0.15,0.3}	0.15	0.1936	0.2688	0.2840	0.2892
{0,0.2,0.4}	0.20	0.2582	0.3584	0.3786	0.3856
{0,0.25,0.5}	0.25	0.3227	0.4480	0.4733	0.4820
{0,0.3,0.6}	0.30	0.3873	0.5376	0.5679	0.5784
{0,0.35,0.7}	0.35	0.4518	0.6272	0.6626	0.6748
{0,0.4,0.8}	0.40	0.5164	0.7168	0.7572	0.7712
{0,0.45,0.9}	0.45	0.5809	0.8064	0.8519	0.8676
{0,0.5,1.0}	0.50	0.6455	0.8960	0.9466	0.9640

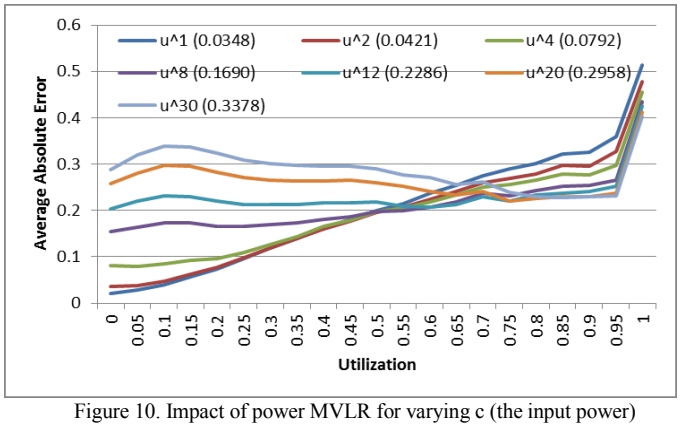

Figure 10. Impact of power MVLR for varying c (the input power)

Similarities clearly exist between the results observed by weighting (changing the regression algorithm itself), and by raising inputs to powers (changing the inputs to the algorithm). Both, provide the most consistent average error when c≈12.

Across a wide range of parameterization of *c* the weighted regression produced a consistently lower average absolute error (Fig. 11), and appears to be the better strategy for predicting large utilizations (Fig. 12 and 13). It also has the advantage of not requiring that input values be in the range 0-1.

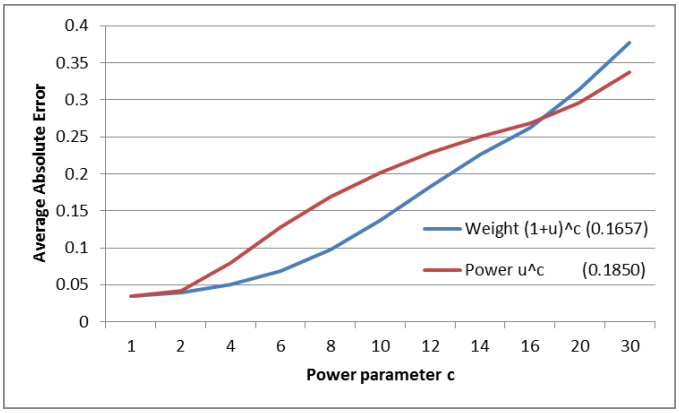

Figure 11. Comparison of power .v. weighting

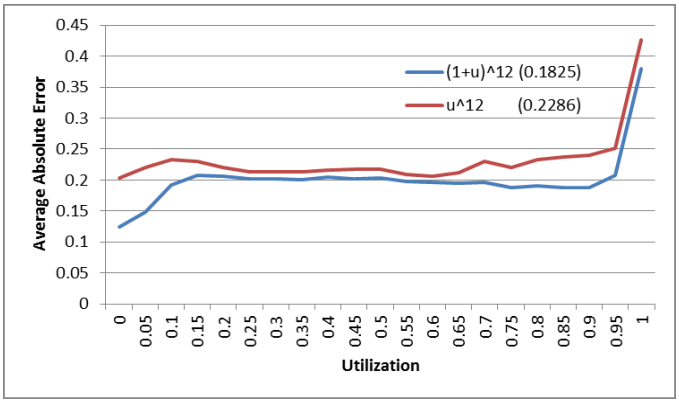

Figure 12. Comparison of power .v. weighting for c=12

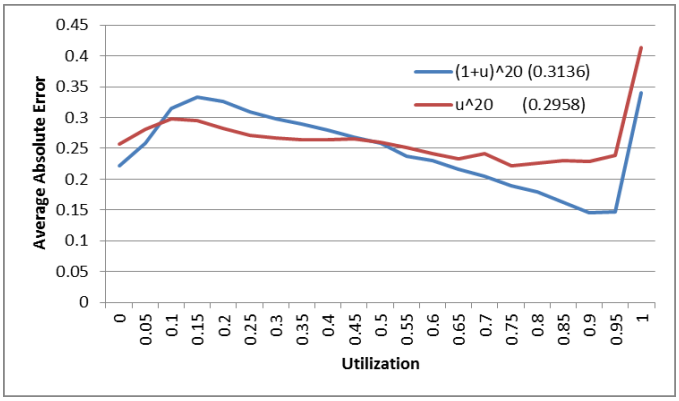

Figure 13. Comparison of power .v. weighting for c=20

V. SEASONALITY

To achieve good long term predictions, one must consider not only changing trends, modelled reasonably well by the approaches suggested above, but also longer term seasonal contributions.

Fourier: Fourier transforms [9] were exploited in an effort to discover obvious cyclic patterns within the data. Strong patterns were observed at the daily and weekly level. Graphing the summation of the ten sine waves with the largest amplitudes, which are the terms that describe the most dominant variability within the input data, fit the overall seasonality within the provided data well, but failed to fit the peaks in the data at all well (Fig. 14).

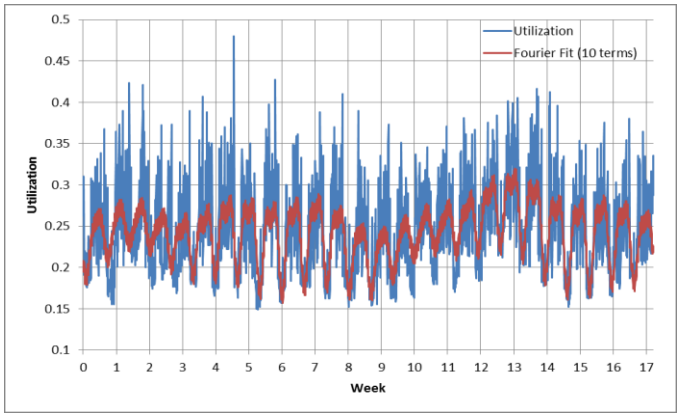

Figure 14. Sample Fourier Fit

We extended this computed transform into the future and used its value at each given hour to predict, which worked well for small utilizations, but not for large utilizations irrespective of the number of sine waves summed (Fig. 15).

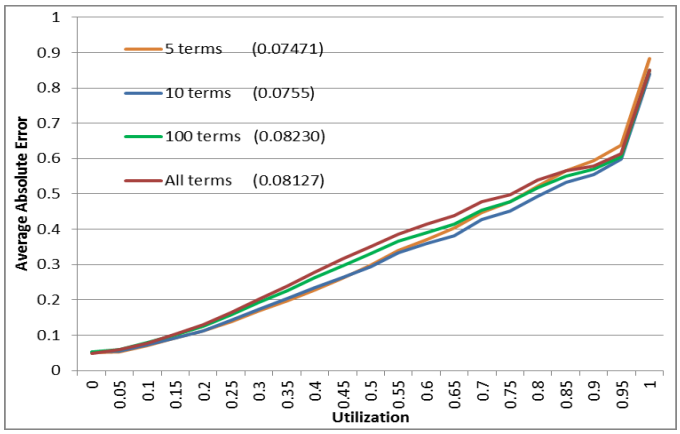

Figure 15. Fourier fit to future data

Scaled Seasonality: To account for the terms not included in the contribution to our prediction, it is reasonable to attempt to scale the Fourier transform to better fit the utilization.

One way of better fitting peaks is to apply a linear transformation to the computed Fourier transform, ignoring all values below some suitable cutoff (ie. maximum – 0.05). We arrange for the minimum to remain unchanged by subtracting it, and then simply scale by mean(observed)/mean(predicted) before adding the minimum back in.

The difference between the scaled Fourier function, and observed data within one representative data source, scaling using the formula $y = 1.51284 * (prediction - 0.145927) + 0.145927$ is shown in Figure 16.

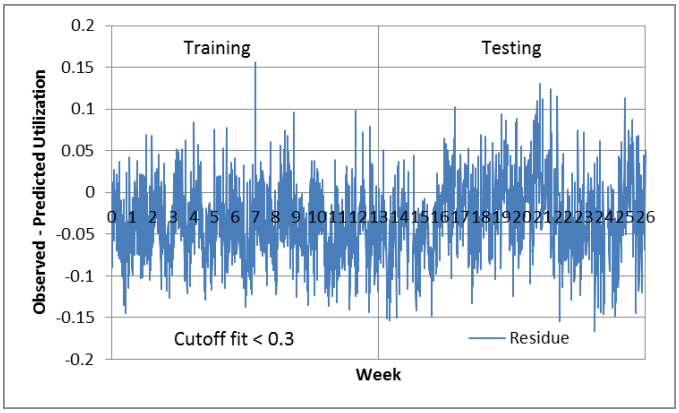

Figure 16. Residue for one data source after scaling seasonality

Using this approach across all data sources was suprisingly effective. Across all of our data sources the average absolute error associated with using a fourier transform to predict future behaviour was visually halved for large utilizations when the fourier transform was suitably scaled (Fig 17).

MVLR did provide better predictive accuracy for low utilizations, and weighted MVLR for high utilizations, but using scaled seasonality to predict future seasonality was competitive with linear regression at high utilizations, and remained so over much longer time frames measured not in hours but in months. This is exciting since long term prediction is inherently difficult.

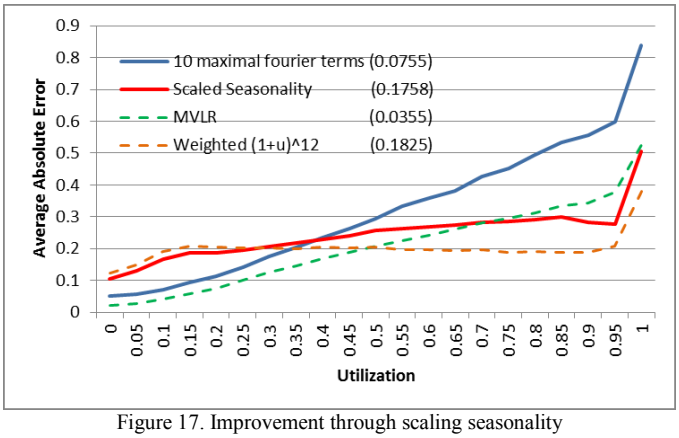

Figure 17. Improvement through scaling seasonality

In general, one cannot assume that seasonal behaviour of machines within a cloud will remain sufficiently static to provide such long term predictability. And clearly, further improvement in predictions may be achieved by developing a hybrid algorithm that exploits both fourier transforms and linear regression simultaneously.

Subtracted Seasonality: We then subtracts the scaled seasonality from the observed utilizations, performing MVLR (as before) on the resulting residue, and then adding the seasonality back in to the resulting prediction (Fig.18).

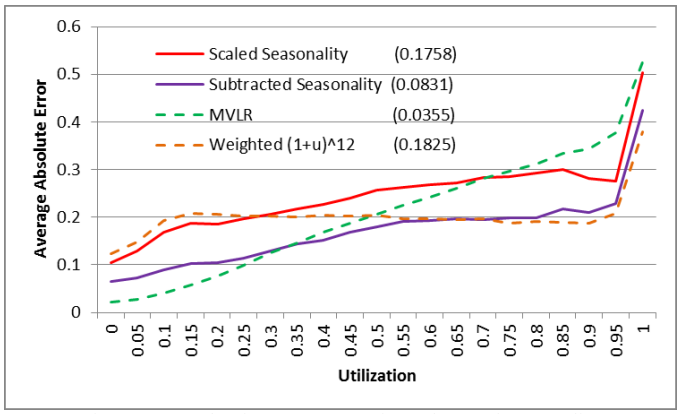

Figure 18. Further improvement using subtracted seasonality

By subtracting the seasonality from the original data we removed much of the variability in the original data, making it more linear, and thus making it fit better with linear prediction models. The results obtained were significantly better than using either fourier transforms, or multivariate linear regression alone. This approach reducted the average absolute error across all inputs for large utilizations by a third, and halved the overall average absolute error.

Table 2: Accuracy of subtracted seasonality prediction

	Observed Utilization										
	0	0.05	0.1	0.15	0.2	0.25	0.3	0.35	0.4	0.45	0.5
0	0.590	0.046	0.004	0.001	0.001	0.001	0.001	0.001	0.001	0.000	0.001
0.05	0.254	0.518	0.070	0.012	0.006	0.005	0.004	0.003	0.003	0.002	0.002
0.1	0.073	0.244	0.379	0.083	0.025	0.015	0.011	0.009	0.007	0.006	0.005
0.15	0.032	0.084	0.236	0.237	0.081	0.032	0.021	0.018	0.014	0.012	0.010
0.2	0.015	0.042	0.127	0.210	0.210	0.086	0.041	0.032	0.022	0.018	0.012
0.25	0.009	0.022	0.074	0.165	0.188	0.222	0.109	0.058	0.037	0.029	0.019
0.3	0.006	0.012	0.041	0.109	0.154	0.185	0.224	0.130	0.066	0.041	0.028
0.35	0.004	0.008	0.023	0.067	0.121	0.140	0.162	0.208	0.130	0.069	0.042
0.4	0.004	0.005	0.013	0.040	0.080	0.101	0.124	0.170	0.248	0.132	0.062
0.45	0.003	0.005	0.008	0.023	0.050	0.068	0.094	0.116	0.153	0.230	0.134
0.5	0.002	0.004	0.006	0.013	0.027	0.046	0.069	0.081	0.106	0.182	0.270
0.55	0.002	0.003	0.004	0.009	0.015	0.031	0.046	0.060	0.077	0.095	0.166
0.6	0.001	0.002	0.004	0.006	0.010	0.020	0.031	0.042	0.054	0.067	0.087
0.65	0.001	0.002	0.003	0.004	0.007	0.014	0.019	0.025	0.034	0.037	0.052
0.7	0.001	0.002	0.002	0.004	0.005	0.009	0.012	0.014	0.015	0.024	0.035
0.75	0.001	0.001	0.002	0.003	0.004	0.007	0.010	0.009	0.009	0.016	0.027
0.8	0.001	0.001	0.002	0.003	0.004	0.005	0.006	0.006	0.007	0.015	0.018
0.85	0.001	0.000	0.002	0.003	0.003	0.004	0.005	0.006	0.005	0.009	0.012
0.9	0.000	0.000	0.001	0.002	0.003	0.003	0.004	0.005	0.005	0.006	0.008
0.95	0.000	0.000	0.001	0.002	0.002	0.003	0.003	0.003	0.003	0.004	0.006
1	0.000	0.000	0.000	0.002	0.002	0.003	0.005	0.003	0.004	0.005	0.005
Good	0.844	0.808	0.685	0.531	0.479	0.493	0.496	0.509	0.531	0.544	0.570
Fair	0.106	0.126	0.204	0.287	0.306	0.288	0.280	0.287	0.286	0.273	0.243
Bad	0.051	0.067	0.111	0.182	0.215	0.219	0.224	0.204	0.183	0.184	0.188

	0.55	0.6	0.65	0.7	0.75	0.8	0.85	0.9	0.95	1
0	0.001	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.001	0.000
0.05	0.002	0.001	0.001	0.001	0.001	0.001	0.001	0.000	0.001	0.000
0.1	0.005	0.005	0.004	0.004	0.004	0.004	0.003	0.002	0.004	0.001
0.15	0.009	0.006	0.006	0.004	0.004	0.004	0.008	0.003	0.005	0.003
0.2	0.017	0.009	0.010	0.008	0.006	0.004	0.004	0.003	0.007	0.004
0.25	0.020	0.015	0.013	0.011	0.007	0.006	0.007	0.004	0.004	0.004
0.3	0.027	0.021	0.016	0.012	0.008	0.008	0.008	0.005	0.004	0.004
0.35	0.035	0.027	0.025	0.018	0.013	0.012	0.011	0.008	0.005	0.005
0.4	0.051	0.037	0.023	0.020	0.012	0.015	0.012	0.009	0.007	0.005
0.45	0.082	0.050	0.035	0.025	0.019	0.016	0.013	0.008	0.003	0.008
0.5	0.159	0.084	0.048	0.037	0.025	0.024	0.018	0.011	0.007	0.008
0.55	0.231	0.153	0.080	0.051	0.027	0.030	0.026	0.014	0.010	0.011
0.6	0.149	0.266	0.148	0.079	0.044	0.034	0.037	0.017	0.015	0.014
0.65	0.068	0.133	0.212	0.146	0.066	0.051	0.042	0.021	0.019	0.018
0.7	0.040	0.062	0.128	0.257	0.137	0.078	0.056	0.032	0.026	0.023
0.75	0.032	0.045	0.103	0.163	0.365	0.212	0.097	0.039	0.033	0.028
0.8	0.024	0.026	0.060	0.073	0.137	0.265	0.175	0.072	0.050	0.041
0.85	0.021	0.024	0.037	0.038	0.058	0.125	0.246	0.145	0.124	0.076
0.9	0.013	0.019	0.028	0.027	0.035	0.061	0.146	0.411	0.266	0.146
0.95	0.008	0.009	0.012	0.014	0.018	0.028	0.053	0.146	0.303	0.278
1	0.007	0.008	0.012	0.011	0.013	0.021	0.036	0.050	0.107	0.322
	0.55	0.6	0.65	0.7	0.75	0.8	0.85	0.9	0.95	1
Good	0.539	0.553	0.488	0.565	0.639	0.567	0.567	0.702	0.676	0.601
Fair	0.241	0.239	0.291	0.242	0.203	0.218	0.241	0.160	0.174	0.222
Bad	0.220	0.208	0.221	0.193	0.158	0.215	0.192	0.138	0.150	0.177

The most significant drawback of using fourier transforms was that unlike regression which could quickly start providing predictions from initially observed results, a substantial amount of prior data must be available, in order to discover seasonality within an input time series. In practice it is proposed that early predictions are predicated on regression alone, while periodically as sufficient data becomes available a fast fourier transform is employed to repeatedly discover seasonality with the input data.

The results certainly appear promising. As a validation of our approach to attempting to predict future values (across the spectrum of possible utilizations, we tabulate for all input sources and each utilization partition, the percentage of the time predictions obtained using scaled seasonality were associated with each utilization partition in Table 2.

Across all the data provided us our subtracted seasonality algorithm was able to correctly predict future utilizations 30% of the time, 59% of the time within ±0.05, (good in Table 2) and 81% of the time within ±0.15 (good+fair in Table 2). Prediction was out by 0.5 or more 2.1% of the time, and by 0.75 or more 0.3% of the time.

VI. LONG TERM PREDICTIONS

For long term scheduling of resources, utilization is not a very useful predictive quantity. Instead of asking how busy a machine is likely to be (which presumes that the number of machines in the system is to remain static) it is far more useful to ask how much work must be handled by the system. This permits some exploration as to how best to accommodate this total work load, through purchase of additional resources, or reductions in the online availability of these resources.

Fortunately, since utilization is simply observed workload divided by maximal workload, predicted workload can be trivially recovered from predicted utilizations. Simply permit utilization predictions to potentially exceed 1.0, and multiply the resulting prediction by the known maximal workload available during the predicted period.

Being interested in the accuracy of scaling the predicted seasonality, over a prolonged period, so as to facilitate long term forecasting, the average absolute error across all data sources is plotted using scaled seasonality (Fig. 19).

Scaled seasonality appears from the graph below to be a good long term predictor of future system performance. There is no systemic weakening of the prediction algorithm over time, even though predictions in week 19 are derived from data obtained between three and six months earlier.

The results indicated that the average absolute error, across all input sources was typically around 0.1, dropping to approximately 0.01 in some cases where there were very few inputs.

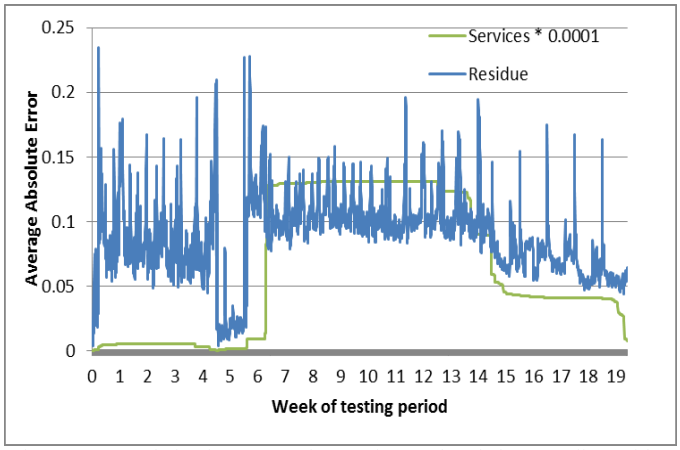

Figure 19. Correlation between active services and scaled seasonality residue

There appeared to be a correlation (0.469) between the average absolute error residue, and the number of services being monitored per hour. This is interesting because the prediction algorithm is applied independently to each service, and is not itself aware of any service but its own. So it cannot know how many services are active.

One plausible explanation for this is that when few services are being monitored, as result of reduced demand, there are as consequence fewer heavily loaded services within the system, with resulting improvement in the observed residues, since we are better at predicting the behavior of under-utilized services. There does appear to be an underlying correlation (0.479) between number of services being monitored and the average utilization of these services (Fig. 20).

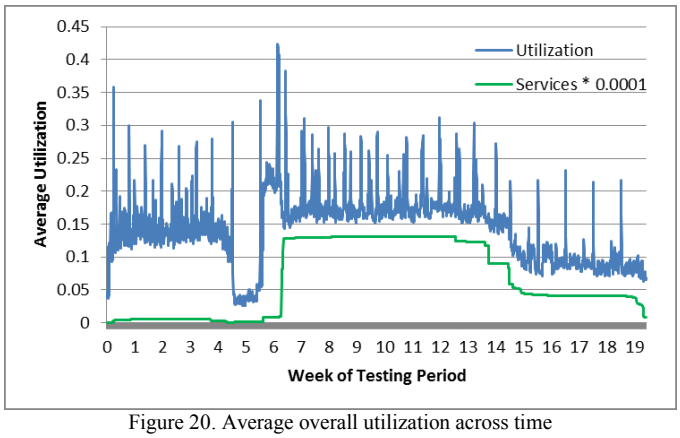

Figure 20. Average overall utilization across time

Subtracted seasonality provides some further improvement over scaled seasonality (Fig. 21). However, sometimes wildly inaccurate and obviously false predictions are generated by the regression algorithm. These are believed to be consequence of near singularities (linear dependence between input lags) within the input being processed by the regression algorithm. This should be addressed by better detecting such near singularities in the lags, and while detected reducing the number of lags used.

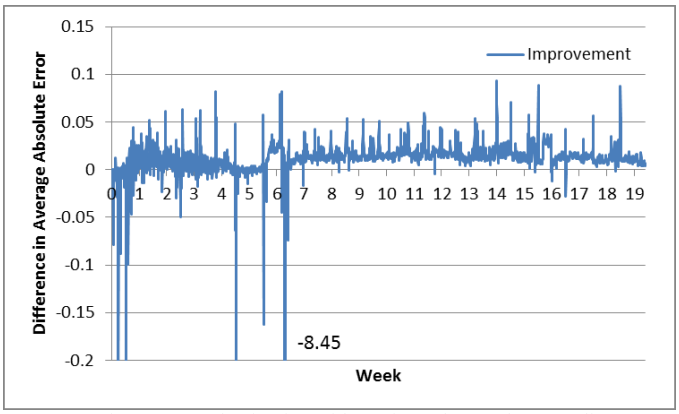

Figure 21. Reduction in residue using subtracted seasonality

VII. ANOMALIES

We are interested in cases where a wide disparity exists between observed and predicted behaviour [4]. Using the scaled seasonality, we plot the frequency of predictions which differ from observed utilization by at least 0.75. During the training period (Fig. 22) when seasonality is being derived, such anomalies highlight potential issues not accounted for, while during the testing period (Fig. 23) they highlight significant departures from expected seasonality.

The maximum number of services displaying an anomaly during the training period in any one hour was 161 (ie. 11.8%), while the average was 3.1 (ie. 0.23%). The maximum number of services displaying an anomaly in the testing period during any one hour was 163 (ie. 12%), while the average was 11.5 (ie. 0.8%). Thus anomalies are comparatively rare.

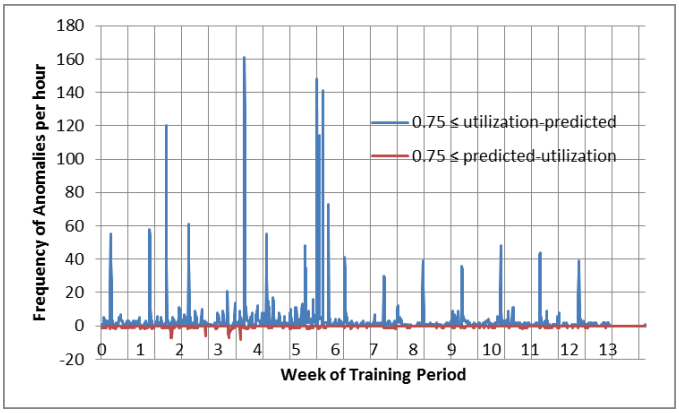

Figure 22. Frequency of anomalies during the training period

To our surprise, there appear to be strong patterns within the anomalies observed during the testing period. Firstly there appeared a strong correlation between over predicting some services, while within the same hour under predicting others. This may be consequence of heavier than expected utilised services bottlenecking the system, denying needed resources to other services. In this case we would under predict the heavily loaded services, and over predict the other services.

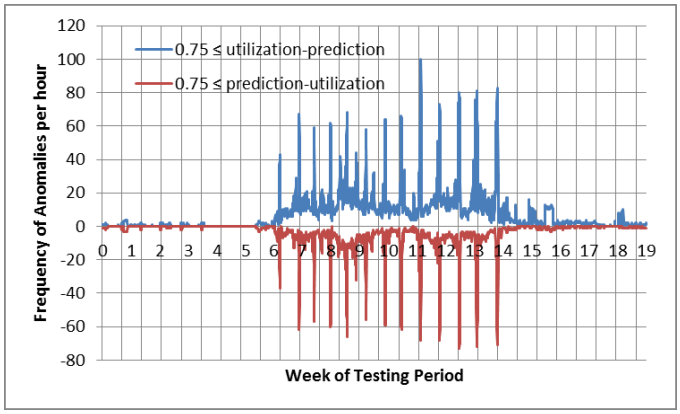

Figure 23. Frequency of anomaly during testing period

Secondly, as shown using the grid lines, anomalies across a large numbers of services during the testing period seem generally to arise approximately every 116 hours. It is not known what causes such a consistent pattern, but it may perhaps be that some very demanding activity performed approximately every 5 days only during the testing period, or some other change to scheduled software execution (such as a backup) was not factored into the earlier constructed seasonality model.

This anomaly may also be an emergent characteristic of the initially computed seasonality across all services, since the discrepancy between predicted and observed values can only exceed 0.75, when predictions or utilizations are either less than 0.25 or greater than 0.75 (Fig. 24). Strangely, within the Fourier series and scaled seasonality there are clear patterns every week (168 hours) as would perhaps be expected, but not every 116 hours.

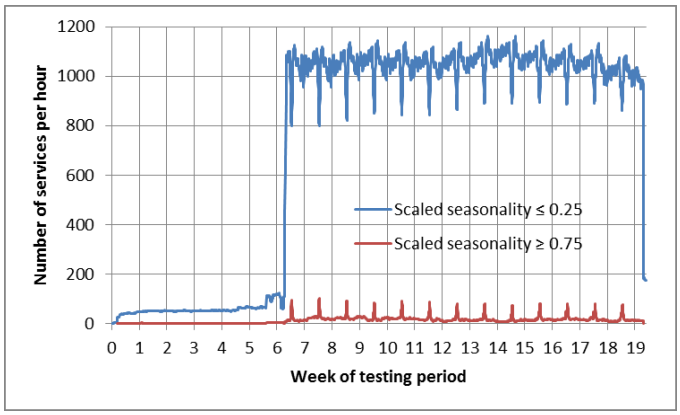

Figure 24. Maximum number of services that might display an anomaly

VIII. THREATS TO VALIDITY

This research was predicated on a single source of data that described performance of a very large number of physical and virtual services, running in a cloud environment, during a comparatively short six month interval. While there was considerable variability in the behaviour of these services, as a collective they appeared for the most part to be idle. This may not be typical within all cloud computing environments.

The appearance that these machines were not heavily utilized might be flawed. Average utilization computed hourly, can potentially be low, even if there are bursts of intense activity within that hour. Performance information was sometimes unavailable, as consequence of either the machines being deactivated, or the monitoring software being disabled. It is not known how such disruptions impacted upon the averages provided us.

We made a best effort to accommodate missing data, but assumptions as to what missing values might have been, necessarily compromise predictive algorithms. Our decision to truncate utilizations greater than 1.0 to 1.0, and to set an upper bound on predicted utilizations at 1.0, helped ensure that prediction appeared closer to high actual utilizations than might otherwise have been the case.

While multivariate linear regression can be expected to respond appropriately to changing trends, our presumption (predicated on studying the data) was that no trend would be present within long term seasonality. If trends were present within the observed seasonality, it would be necessary to attempt to scale the seasonality using something more complex than a simple linear equation.

A final caveat is perhaps in order. No matter how good a predictive algorithm is, it is inevitable that it will sometimes give misleading results. And it is possible that a few misleading results will more than undermine the benefits of relying on such algorithms. That question lies beyond the scope of this paper.

IX. CONCLUSIONS AND FURTHER WORK

System utilization can peak both as a consequence of regular seasonality considerations, and as a consequence of a variety of anomalies, that are inherently hard to anticipate. It is not clear that the optimal way of predicting such peak system activity is through approaches such as multivariate linear regression, since such prediction is predicated on the totality of the data observed, and tends to produced smoothed results rather than results that emphasize the likelihood of system usage exceeding capacity.

We have presented a number of modifications to standard multivariate linear regression, and to Fourier transforms, which individually and potentially collectively do improve the ability of multivariate linear regression to predict peak utilizations with reasonably small average absolute error [2].

Using windowing; scaling the results; weighting the inputs; and exploiting knowledge about seasonality, each produce improvement in the ability to accurately fit predictions to peak utilizations [7]. Applying multivariate linear regression to

powers of the input data, and employing weighted linear regression, both proved to be good techniques for fitting predictions to large subsequently observed utilizations, but resulted in worse fits against low utilizations.

We have also explored prediction using Fourier analysis and have suggested mechanisms for improving the predictive capability of this analysis. We finally presented algorithms that leveraged both Fourier analysis, and multivariate linear regression, to improve our predictive capabilities further, which provided near optimal results, when compared to all other approaches, across the spectrum of observed utilizations.

With the data provided us, we were able to obtain good short term (future hour) predictions of system utilization and to achieve good prediction of longer term trends (measured in months). Both are important. We have also exploited our results to better understand the nature of anomalies that occurred within them.

Significantly, the approaches presented permitted construction of a relatively flat distribution of average absolute errors, across all observed utilizations, even though great asymmetry existed between the frequencies of high and low utilizations.

Within the data provided us, there was often comparatively narrow crossover regions where particular algorithms could be seen to transition from providing better performance than the competition to worse performance. Armed with the predictive ability to determine which side of the crossover system performance was likely to be, it would be very tempting to develop hybrid algorithms, which adaptively decided to employ differing predictive strategies, predicated on results seen to date. This remains an opportunity for further research.

ACKNOWLEDGMENT

This research is supported by grants from CA Canada Inc., and NSERC. It would not have been possible without the interest, assistance and encouragement of CA. Allan Clarke and Laurence Clay provided valuable input to this paper.

REFERENCES

[1] M. Andreolini and S. Casolari, "Load prediction models in web based systems." Proceedings of the 1st international conference on Performance evaluation methodolgies and tools. 2006, ACM: Pisa, Italy. p. 27.

[2] J.S. Armstrong. "Evaluating forecasting methods." Principles of forecasting. A Handbook for Researchers and Practitioners. Kluwer Academic Publishers. 2001

[3] CA Technologies. http://www.ca.com

[4] P.K. Chan, M.V. Mahoney. "Modeling multiple time series for anomaly detection." IEEE 5th Int. Conf. on Data Mining. 2005

[5] P. A. Dinda and D. R. O'Hallaron. "Host load prediction using linear models." Journal Cluster Computing. Vol. 3. Issue 4. 2000

[6] N.R. Draper, H. Smith. "Applied regression analysis," Third Edition. Wiley Series in Probability and Statistics. 1998 [2]

[7] M. Duan. "Time series predictability." PhD Thesis. Marquette. 2002

[8] W. Fair. Jr. "An algorithm for weighted linear regression." Code Project. 2008

[9] M. Frigo and S. Johnson. "The Fastest Fourier Transform in the West". MIT-LCS-TR-728. Massachusetts Institute of Technology. 1997

[10] A. S. Ganapathi. "Predicting and optimising system utilization and performance via statistical machine learning." PhD Thesis. University of California. Berkeley. 2009

[11] D. Gmach, J. Rolia, L. Cherkasova, A Kemper. "Workload analysis and demand prediction of enterprise data centre applications." Proc. IEEE 10th Int. Symposium on Workload Characterization 2007

[12] Y. Huhtala. J. Karkkainen. and H. Toivonen. "Mining for similarities in aligned time series using wavelets." Proc. SPIE Vol 3695. Data Mining and Knowledge Discovery: Theory, Tools and Technology.

[13] E. Hurwitz, T. Marwala. "Common mistakes when applying computational intelligence and machine learning to stock market modelling." 22 August 2012. http://arxiv.org/abs/1208.4429v1

[14] M. Istin., A. Visan, F. Pop, and V. Cristea. "Decomposition based algorithm for state prediction in large scale distributed systems." Ninth International Symposium on Parallel and Distributed Computing. 2010. Istanbul, Turkey

[15] A. Khan, X. Yan, S. Tao, N. Anerousis. "Workload characterization and prediction in the cloud. A multiple time series approach." Network Operations and Management Symposium (NOMS) April 2012

[16] M.S. Magnusson. "Discovering hidden time patterns in behavior: T-patterns and their detection." Behaviour Research Methods, Instruments and Computers. Feb 2000, 32(1).

[17] R. Nathuji. A. Kansal, and A. Ghaffarkhah. "Q-Clouds: Managing performance interference effects for QoS-aware clouds." Proc. 5th European Conf. on Computer Systems. EuroSys 2010

[18] D. Neuse, S. Oberlin, P. Peterson, S. Williamson. "Workload forecasting research – business problem and solution value." (Internal CA power point presentation). May 12, 2012

[19] S. Nikolov. "Trend or no trend: A novel nonparametric method of classifying time series." PhD Thesis. MIT. 2011

[20] R.J. Povinelli. "Time series data mining: Identifying temporal patterns for characterization and prediction of time series events." PhD Thesis. Milwaukee, Wisconsin. December 1999

[21] J. Srinivasan and N. Swaminathan. "Predicting behavior patterns using adaptive workload models." Proc. 7th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems. 1999

[22] M. Stokely, A. Mehrabian, C. Albrecht, F. Labelle, A. Merchant. "Projecting disk usage based on historical trends in a cloud environment." Proc. 3rd Int. Workshop on Scientific Cloud Computing. 2012

[23] D.W. Yoas. and G. Simco. "Resource utilization prediction: A proposal for information technology research." ACM Joint Conference on IT Education and Research in IT. 2012

[24] T. Zheng, M. Litoiu, M. Woodside. "Integrated Estimation and Tracking of Performance Model Parameters with Autoregressive Trends". ICPE'11, March 14–16, 2011