Operational Data Analysis: Improved Predictions Using Multi-Computer Pattern Detection

Ricardo Vilalta, Chid Apte, and Sholom Weiss

Operational Data Analysis: Improved Predictions Using Multi-Computer Pattern Detection

Ricardo Vilalta, Chid Apte, and Sholom Weiss

IBM T.J. Watson Research Center
30 Saw Mill River Rd., Hawthorne N.Y., 10592 USA
Ph. (914) 784-7784
http://www.research.ibm.com/people/v/vilalta
vilalta@us.ibm.com

Abstract. Operational Data Analysis (ODA) automatically 1) monitors the performance of a computer through time, 2) stores such information in a data repository, 3) applies data-mining techniques, and 4) generates results. We describe a system implementing the four steps in ODA, focusing our attention on the data-mining step where our goal is to predict the value of a performance parameter (e.g., response time, cpu utilization, memory utilization) in the future. Our approach to the prediction problem extracts patterns from a database containing information from thousands of historical records and across computers. We show empirically how a multivariate linear regression model applied on all available records outperforms 1) a linear univariate model per machine, 2) a linear multivariate model per machine, and 3) a decision tree for regression across all machines. We conclude that global patterns relating characteristics across different computer models exist and can be extracted to improve the accuracy in predicting future performance behavior.

1 Introduction

Knowledge about the performance of a computer through time requires continuous monitoring. If the computer reports problems, information captured during this monitoring period can be of valuable help to determine the nature of the problem. Analyzing performance data through time may also help to decide how to fine-tune or refine the computer under analysis, which may ultimately lead to improved performance.

The automated process of monitoring the performance of a computer, storing all recorded data, analyzing the data, and generating results, is what in this paper we refer to as Operational Data Analysis (ODA). Such process is comprised by an area relatively unexplored: although mechanisms exist for monitoring a device or mechanism through time, little has been done to automate the process of extracting patterns from such data to predict potential failures, or to suggest an action plan that can guarantee optimal performance.

This paper begins with a brief overview of Operational Data Analysis, a multi-step process highly relevant to systems management. The theme of the paper, however, focuses on a novel data mining approach for the time-series forecasting problem that extract patterns from historical data. The data-mining technique builds a multivariate linear regression model to the data using information extracted from a central database repository containing information about thousands of computers. We show how this model outperforms 1) a linear univariate model per machine, 2) a linear multivariate model per machine, and 3) a decision tree for regression across all machines. Our results provide evidence to believe that global patterns relating characteristics across different computer models exist. Taking advantage of such global patterns can improve the accuracy in predicting future performance behavior.

The organization of the paper follows. Section 2 explains each of the four steps comprised by the term Operational Data Analysis. Section 3 focuses on the data-mining step, describing a set of experiments that compare several models for the accurate prediction of machine performance. Section 4 shows our empirical results. Section 5 reviews related work in the literature. Finally, Section 6 presents our conclusions and future work.

2 Operational Data Analysis: Overview

Relevant to systems management is the automated process of predicting computer performance. One way to automate such process can be described in four basic steps (Figure 1). We describe the characteristics of each step next.

1. **Data Collection.** A mechanism in charge of taking snapshots reflecting the status of the computer must exhibit the following characteristics: a) snapshots should be taken every fixed interval of time, to ensure data consistency, b) the length of each interval should allow us to capture all important events. The length of each interval should provide enough granularity in the data, such that no important event is missed by the probes. Data collected by the probes may be averaged or aggregated before sending it to a data repository for analysis.

2. **Data Storage.** Information reflecting the status of a device through time is expected to yield large number of records. Furthermore, we expect to have applications where many devices are being monitored simultaneously. Consequently, data collected regularly over time should be stored in a common data warehouse. By consolidating all information in one place, data-analysis techniques can not only find trends on a single customer machine, but also search for correlations among different machines. This may lead to detect patterns that span across different models or even platforms (Section 3).

3. **Data Analysis.** Next is the application of data analysis techniques (e.g. machine learning, data mining, non-parametric statistics) to operational data. The idea is to search for patterns hidden in the data that may be helpful to understand how to improve machine performance. Data about the
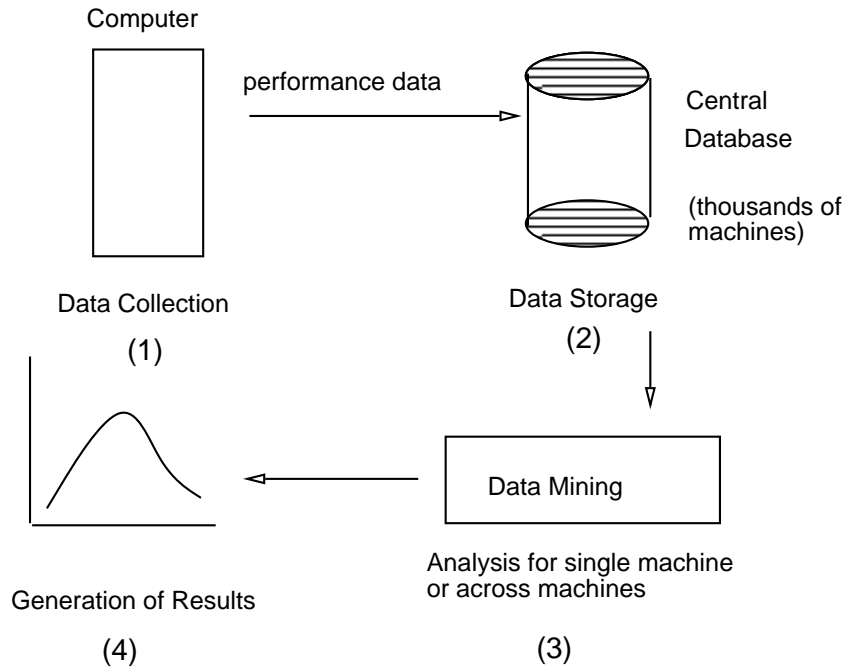
Computer

performance data

Central
Database

(thousands of
machines)

Data Collection

(1)

Data Storage

(2)

Data Mining

Generation of Results

(4)

Analysis for single machine
or across machines

(3)

Fig. 1. The four steps in Operational Data Analysis.

performance of a machine through time may reveal that, at some point in
the future, certain components may experience failure. This information can
be important to avoid reaching conditions where it is known that certain
component(s) will not work properly. The person in charge of the device or
process may then be in good position to plan for possible upgrade scenarios.

4. **Generation of Results**. We may go beyond diagrams displaying statistics
   about the data, to include: a) the construction of a database enabling us
   to see results from different perspectives and under different search criteria,
   b) promotions and announcements; if the result of the analysis suggests the
   need for a machine upgrade, an opportunity exists to promote products that
   can satisfy the upgrade requirements.

2.1   System Description

The 4-step process described above has been successfully implemented in the
automatic notification of computer performance to users of the IBM AS/400
platform. The prospective user starts by registering trough a web-based applica-
tion. Separately the PM/400 monitoring software begins capturing performance
data on the user's (AS/400) machine.

A central database contains information on the performance of all AS/400
computers that have been registered for this service. Each record in the database
reports the values of tens of performance parameters for a particular machine,
month of the year, and shift, e.g., the average response time for machine with

serial number X, was on average 1.2 seconds in the month of July, 1999, during the morning shift. We limit ourselves to only twenty relevant performance parameters as input for the analysis; the parameters are listed as follows: response time, maximum response time, cpu utilization, memory utilization, disk utilization, disk arm utilization, system utilization, shift, avg. trans./hr, max. trans./hr, avg. jobs /day, max. jobs /day, avg. IO Mb/sec, max. IO Mb/sec, avg. interactive jobs, max. interactive jobs, avg. batch jobs, max. batch jobs, avg. total transactions, max. total transactions. From the parameters above we try to form predictions for the first six parameters: response time, maximum response time, cpu utilization, memory utilization, disk utilization, and disk arm utilization.

When a user logs into the system, the four steps of ODA are performed. Assuming enough information has been collected by the PM/400 monitoring software, a remote connection brings data to a server where the data-mining step results in a graph displaying (for each parameter) past performance, together with predictions on future performance. Figure 2 shows an example of a result graph.
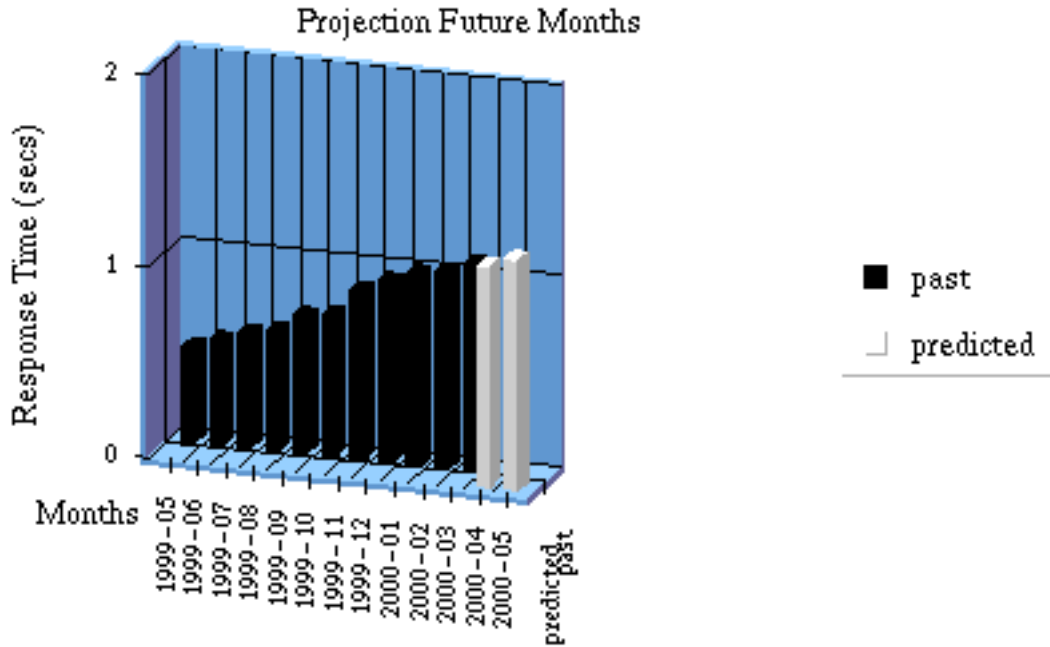

Fig. 2. The resulting graph after the four steps in Operational Data Analysis have been performed.

3   Time-Series Analysis: A Data Mining Approach

The most challenging step in ODA is choosing the right data-mining technique for the application at hand. This section describes a series of experiments that compare several models for the time-series prediction problem. We wish to find a good model for the accurate prediction of future performance behavior. Our central claim is that an advantage is gained by extracting information related to multiple machines. Such global analysis enables us to extract patterns that help improve the accuracy of our predictions. In contrast, a model that is limited to information about a single machine is shown to perform less effectively.

3.1   Experimental Methodology

Our experiments use different models for prediction, e.g., univariate linear regression, multivariate linear regression, decision tree. These models attempt to capture the general trend of a dependent variable through time. We ignore periodic trends (seasonal analysis) or cyclic behavior, which we plan to address in the next project phase.

Some models were applied to data corresponding to a particular machine, while other models were applied to all the database at once. In the first case, we report average performance over all machines (Figure 3, Algorithm 1). If a particular machine has n records in the database, we take $n - 1$ records for training and use the last record for testing. In other words, we test the ability of the model to predict one month in the future. In the second case we transform the database by appending to each record the value of the target performance parameter for the next month (Figure 3, Algorithm 2). Assume a particular machine has n records in the database and that response-time acts as the target function. Record 1 for this machine is appended an extra target field corresponding to response-time in Record 2 (in the next month). This field becomes the target-function value to learn. The last record is evidently eliminated. We repeat this transformation process for all records in the database. We then train a model with $\frac{2}{3}$ of the transformed database, and test with the remaining $\frac{1}{3}$. We do this for each of the six parameters of interest (listed above), resulting in six transformed databases.

For each target parameter we report on the relative error. The relative error compares the error of the model with the error obtained by using the mean of all past values as our prediction. Relative error is the fraction resulting by dividing model-error by mean-error. We wish to see values much less than 1. Values above 1 implies we are better off simply using the mean value as our prediction.

In all our experiments we used the Data-Miner Software Kit [11]. Tests were done on an IBM RISC/6000 model 7043.

4   Results

Looking One Month In The Past Our first experiment uses a very simple model as baseline for comparison. We take the value of the previous month to

Algorithm 1: Single-Machine Analysis
Input: Dataset D, size of dataset m,
Model M
Output: Average Accuracy A
SINGLE_MACHINE_ANALYSIS()
(1)   Let $A = 0$
(2)   **while** (exist records in D)
(3)   Train M in $n-1$ records
(4)   of same machine
(5)   Test M in last record n
(6)   Let $A = A+$ accuracy of M
(7)   **end while**
(8)   Let $A = A/$ no. of machines
(9)   **return** A

Algorithm 2: Multiple-Machine Analysis
Input: Dataset D, Model M
Output: Accuracy A
MULTIPLE_MACHINE_ANALYSIS()
(1)   Transform database D by
(2)   appending to each record
(3)   the target variable one
(4)   month ahead.
(5)   Let the new database be D'
(6)   Let the size of D' be p
(7)   **while** (exist records in D')
(8)   Train M in $\frac{2}{3} \times p$ records
(9)   Test M in the other
(10) $\frac{1}{3} \times p$ records
(11)   **end while**
(12)   **return** Accuracy A of M

Fig. 3. Algorithm 1 describes a single-machine analysis. Algorithm 2 desctribes a multiple-machine analysis.

predict the value for the next month. We do this for each of the six relevant performance parameters. Table 1, Row 2, shows the results. Surprisingly this model is reasonably accurate, which we take to mean there is on average little variation in performance activity between consecutive months. A parameter with very low relative error is disk-arm utilization.

Single-Machine Linear Regression Model The next experiment tests a single univariate linear regression model. As explained before, all records corresponding to the same machine, except the last, are used to train the model, the last record is used for testing. We then averaged over all machines. Table 1, Row 3, shows the results. Relative error is in all cases above one, and in three out of six cases above 2. Model performance is poor.

Table 1. Results for Relative Error.

Model	Relative Error					
	Response Time	Max. Response Time	CPU Util.	Memory Util.	Disk Space	Disk Arm
Previous Month	0.25	0.60	0.43	0.104	0.109	0.06
Linear Regression Univariate Per Machine	2.04	2.10	1.99	1.96	1.74	2.09
Linear Regression Mulivariate Per Machine	1.66	2.07	1.16	1.22	0.96	1.13
Linear Regression Multivariate All Records	0.27	0.49	0.28	0.00	0.00	0.00
Decision Tree All Records	0.54	0.54	0.26	0.01	0.00	0.07

Single-Machine Multivariate Linear Regression Next we test a multivariate linear regression model per customer. To proceed we use the transformed database in which the value of the target function is appended at the end of each record. After averaging over all computers, we obtain the results shown in Table 1, Row 4. Most relative errors are above one. Although this model outperforms the univariate linear regression model per computer, accuracy remains below expectations.

All-Machines Multivariate Linear Regression We now describe a series of experiments that employ all records in the transformed database. The general approach is depicted in Figure 4 (explained in Figure 3, Algorithm 2). The database is transformed by adding to each record the value of the target performance parameter one month ahead in time. The training set contains data for multiple parameters and thousands of computers. Applying a learning algorithm to this data is expected to capture patterns that span across multiple AS/400 computers.

The first experiment applies a multivariate linear regression model to $\frac{2}{3}$ of all available records. We used the remaining $\frac{1}{3}$ for testing. It is important to emphasize the main goal behind this analysis. The final model is intended to capture patterns not only relevant to each machine, but across machines. Table 1, Row 5, shows the results. Relative error is in all cases below one and for memory utilization, disk-space utilization, and disk-arm utilization, the error is zero with two significant figures. We notice also that most parameters have low variance which lead us to conclude that the multivariate linear regression model across all machines is a good model to predict performance behavior.

A similar experiment like the one above was conducted to predict 2-6 months in the future (tables not shown). In this case for most parameters relative error increases when compared to a one-month prediction, but remains stable after four months, and the relative error rarely goes above one. We conclude that even
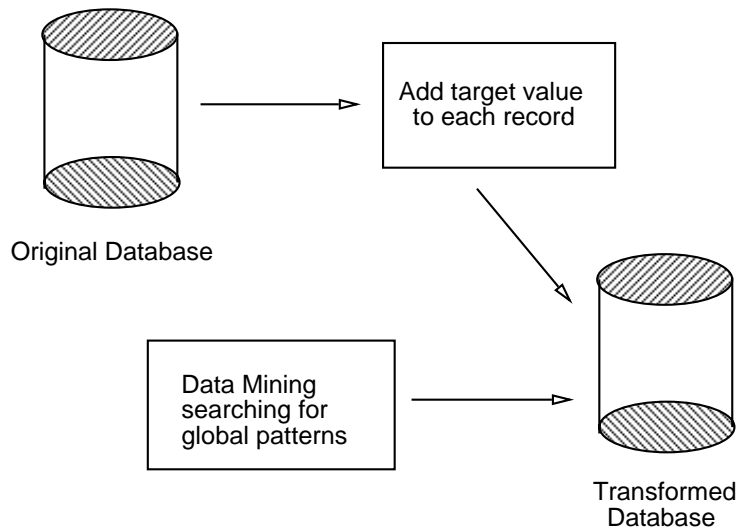
Fig. 4. The process to extract gobal patterns from the performance database.

for 2-6 month predictions, the multivariate linear regression method remains a good model of choice.

All-Machines Decision-Tree For Regression Lastly, we tried a decision tree for regression over all records in the transformed databases. We used again the $\frac{2}{3}$ vs. $\frac{1}{3}$ split for training and testing. Results are shown in Table 1, Row 6. Overall the relative error for the decision tree is higher than the multivariate linear regression model. The decision tree performs well on memory, disk space, and disk arm utilization. We conclude a decision-tree model for regression does not outperform the multi-variate linear regression model.

5 Related Work

Computer performance forecasting is a topic of intensive research. Work is reported in the prediction of network performance to support dynamic scheduling [2], and in the prediction of traffic network [3].

An example of a study that uses past historical data to predict computer performance is reported by Yeh and Patt [4], where the goal is to produce a branch predictor to improve the performance of a deep pipelined micro-architecture. Similar studies are reported in the literature [5,9,10]. Compared to our approach these studies focus on predicting at the instruction level, whereas we contemplate the idea of predicting system behavior (e.g., response time, cpu utilization, etc.).

Zhichen et al. [6] predict the execution time of programs on parallel architectures. The approach is also data based, but is limited to predicting execution time alone.

A common approach to performance prediction proceeds analytically, by relying on specific performance models. For example, Hsu and Kremer [8] study prediction models at the source code level which play an important role for compiler optimization, programming environments, and debugging tools. Our approach is not model-based but data-based, which enables us to apply data mining techniques. Other studies have tried to bridge the gap between a model-based approach versus a data-based approach [7].

Apte and Hong [1] use rule-based induction in the prediction of financial market behavior. Their approach is akin to ours in that the model (rule-based) is applied to an entire database containing information about many entities (large capital firms) to predict future behavior.

6 Conclusions and Future Work

This paper compares the predictive accuracy of several models within the time-series analysis framework. Each record in the database under analysis describes the monthly-average performance of an AS/400 computer; our goal is to produce accurate performance predictions. Our experiments compare a univariate linear regression model per machine, a multivariate linear regression model per machine, a multivariate linear regression model across all machines, and a decision tree for regression across all machines. The multivariate linear regression model across machines shows the best performance overall.

An interpretation of the results reported in Section 4 can be advanced by comparing the difference between the two kind of models applied to the performance database: one applied to each machine separately vs. one applied simultaneously to all machines. Experimental results give a preference to the latter. We then address the question: why is there an advantage by looking for patterns that span across multiple computers? To answer this question it is reasonable to suppose that for this particular domain, machines that belong to the same model will experience similar performance if the overall machine utilization is the same. Thus, looking for patterns across computers increases the evidential support for correlations between the input variables and the target variable. For example, assume cpu utilization grows linearly with the number of batch jobs. If we produce a hypothesis based on data corresponding to a single machine, we may be able to support this linear correlation, but only with a limited number of records. In contrast, applying the same model using data from all machines, enables us to increase our confidence to decide if the linear correlation holds or if it was merely apparent due to the low number of records in the single-machine analysis.

Overall, we believe this kind of global pattern analysis deserves special attention, particularly today in which the amount of information stored in many central databases provides stringent data with which to produce solid inferences.

Future work will apply a larger variety of models to the entire performance database, and will attempt to derive a deeper theoretical explanation of the advantages that come by analyzing data corresponding to multiple computers.

References

1. Apte Chid and Hong Se June: **Predicting Equity Returns from Securities Data**. In Advances in Knowledge Discovery and Data Mining. Ed. Fayyad, U.M. and Pratetsky-Shapiro and Smyth, P. and Uthurusamy, R. AAAI Press, (1996) 541–560.
2. R. Wolski: **Forecasting network performance to support dynamic scheduling using the network weather service**. In Proceedings of the High Performance Distributed Computing Conference (1997).
3. K. Claffy and G. Polyzos: **Tracking Long-term Growth of the NSFNET**. In Communications of the ACM (1994).
4. Yeh T-Y. and Patt, Y.: **Alternative implementation of Two-Level Adaptive Branch Prediction**. In Proceedings of the 19th International Symposium on Computer Architecture, Gold Coast. Australia (1992) 124–134.
5. Brad, Calder and Dirk, Grunwald and Joel, Emer: **A system level perspective on branch architecture performance**. In Proceedings of the 28th Annual IEEE/ACM International Symposium on Microarchitecture. Ann Arbor, MI (1995) 199-206.
6. Zhichen, Xu and Xiaodong, Zhang and Lin, Sun: **Semi-Empirical Multiprocessor Performance Predictions**. TR-96-05-01, University of Texas, San Antonio, High Performance Comp. and Software Lab (1996).
7. Mark E. Crovella and Thomas J. LeBlanc: **Parallel performance prediction using lost cycles analysis**. In Supercomputing '94 (1994).
8. C-H. Hsu and U. Kremer: **A framework for qualitative performance prediction**. In Department of Computer Science, Rutgers University. Technical Report LCSR-TR98-363 (1998).
9. Brad Calder and Dirk Grunwald: **Next cache line and set prediction**. In ACM (1995) 287–296.
10. N. P. Jouppi and P. Ranganathan: **The relative importance of memory latency, bandwidth, and branch limits to performance**. In The Workshop on Mixing Logic and DRAM: Chips that Compute and Remember (1997).
11. Weiss Sholom and Indurkhya Nitin: **Predictive Data Mining**. Morgan Kaufmann Publishers (1998).