Information Systems 76 (2018) 46–58

Contents lists available at ScienceDirect

Information Systems

journal homepage: www.elsevier.com/locate/is

# Performance prediction and adaptation for database management system workload using Case-Based Reasoning approach[☆]

Basit Raza[a,*], Yogan Jaya Kumar[b], Ahmad Kamran Malik[a], Adeel Anjum[a], Muhammad Faheem[c]

[a] Department of Computer Science, COMSATS Institute of Information Technology, Islamabad 45550, Pakistan
[b] Faculty of Information and Communication Technology, Universiti Teknikal Malaysia Melaka, Melaka 76100, Malaysia
[c] Department of Computer Engineering, Abdullah Gul University, Kayseri 38039, Turkey

ARTICLE INFO

*Article history:*
Received 9 August 2017
Revised 13 March 2018
Accepted 23 April 2018
Available online 25 April 2018

*Keywords:*
Workload management
Autonomic Computing
Case-based Reasoning
Prediction
Adaptation

ABSTRACT

Workload management in a Database Management System (DBMS) has become difficult and challenging because of workload complexity and heterogeneity. During and after execution of the workload, it is hard to control and handle the workload. Before executing the workload, predicting its performance can help us in workload management. By knowing the type of workload in advance, we can predict its performance in an adaptive way that will enable us to monitor and control the workload, which ultimately leads to performance tuning of the DBMS. This study proposes a predictive and adaptive framework named as the Autonomic Workload Performance Prediction (AWPP) framework. The proposed AWPP framework predicts and adapts the DBMS workload performance on the basis of information available in advance before executing the workload. The Case-Based Reasoning (CBR) approach is used to solve the workload management problem. The proposed CBR approach is compared with other machine learning techniques. To validate the AWPP framework, a number of benchmark workloads of the Decision Support System (DSS) and the Online Transaction Processing (OLTP) are executed on the MySQL DBMS. For preparation of training and testing data, we executed more than 1000 TPC-H and TPC-C like workloads on a standard data set. The results show that our proposed AWPP framework through CBR modeling performs better in predicting and adapting the DBMS workload. DBMSs algorithms can be optimized for this prediction and workload can be controlled and managed in a better way. In the end, the results are validated by performing post-hoc tests.

© 2018 Elsevier Ltd. All rights reserved.

## 1. Introduction

Workload management in the Database Management System (DBMS) plays an important role in its performance tuning. Database workload consists of a batch of queries or requests that are executed in a DBMS. The volume of data is increasing day by day which in turn increases the complexity of data management and decreases the performance of a database. For humans, it has become difficult or even impossible to manage the large-scale data due to its complexity and heterogeneity. Due to this, it gained the attention of DBMS researchers and vendors to build such types of DBMSs that have the capability of managing activities proactively. In DBMSs, as it is difficult to monitor and control the workload before execution, therefore, its performance could not be determined beforehand. If we know the performance of a workload in advance, we can control its management. Similarly, other DBMS issues such as system sizing and capacity planning cannot be handled without knowing the information about a workload before its execution. In workload management, many questions arise regarding workloads, such as, when to execute a workload or stop a problematic workload and what will be its performance? and other questions related to system sizing and capacity planning. The Autonomic Computing (AC) technology can be used

*Abbreviations:* DBMS, Database Management System; AWPP, Autonomic Workload Performance Prediction; OLTP, Online Transaction Processing; KCCA, Kernel Canonical Correlation Analysis; CBR, Case-Based Reasoning; DBA, Database Administrator; AC, Autonomic Computing; KNN, K-nearest neighbor; SVM, Support Vector Machine; TPC, Transaction Processing Council; QEP, Query Execution Plan; WFV, Workload Features Vector; PMV, Performance Metrics Vector; Dbt2, Database test 2; ET, Execution Time; WL Size, Workload Size; APV, Adjusted p-value.

☆ The TPC benchmark setups have not been audited as per TPC specifications, TPC-H and TPC-C mean TPC-H like and TPC-C like queries as our representative workloads.

* Corresponding author.
*E-mail address:* basit.raza@comsats.edu.pk (B. Raza).

https://doi.org/10.1016/j.is.2018.04.005
0306-4379/© 2018 Elsevier Ltd. All rights reserved.

in self-management of a DBMS workload that can enable a system to handle the workload with less or no human involvement. Many algorithms, frameworks, and tools have been developed for managing a database workload autonomically [1,2]. The autonomic workload management has incorporated a few autonomic characteristics which include self-inspection, self-optimization, self-adaptation, self-configuration, and self-prediction [3–5]. The prediction of a few database performance metrics has been performed in literature; however, many performance metrics are yet to be investigated which can be helpful in scheduling, adaptation, optimization, and resource allocation in DBMSs. As a workload may change anytime, due to its evolving behavior, so the workload adaptation is becoming a need for database systems. Many researchers have developed adaptation frameworks and models by using traditional approaches to solve workload adaptation issues. Currently, several research works are being carried out in designing applications and systems having self-* properties of AC [6,16–18,30,52].

The study provides a solution for predicting the performance and adapting the changing behavior of the workload. An Autonomic Workload Performance Prediction (AWPP) framework is developed for predicting the performance metrics which is helpful in performance tuning of DBMS. For workload performance prediction, the Kernel Canonical Correlation Analysis (KCCA) algorithm computes the distance between new query projection and existing query projection using K Nearest Neighbor (KNN). However, it is not adaptive as it computes the distance for each new query [19]. In this study, for workload performance prediction, Case-Based Reasoning (CBR) approach is used that works through reasoning based on available solution-cases stored in the repository and provides the required solution. The CBR has four phases that are Retrieve, Reuse, Revise, and Retain [36]. It works autonomically without human involvement, however, initial training is required by the Database Administrator (DBA). For workload performance prediction and adaptation, the cases are retrieved and retained in the repository making it predictive and adaptive. The performance of the proposed CBR approach is compared with other machine learning techniques. The AC properties such as self-inspection, self-prediction, and self-adaptation are incorporated in AWPP framework. Upon execution firstly, the workload self-inspected to extract its metrics. Performance metrics are predicted through self-prediction and self-adaptation is performed to adapt according to behavioral changes in the workload. The training and testing data is created by executing several workloads which enabled in predicting the workload performance.

The objectives of the study are as follows. The first objective is to propose a workload performance prediction framework. The second objective is to predict the performance metrics of the workload before executing it based on workload metrics. The third objective is to adapt the changes to the changing behavior of workload. The contributions of the study are described here. It provides CBR-based AWPP framework that produced effective and accurate workload performance prediction and has the ability of workload adaptation. MySQL status variables are studied and existing features from different studies are analyzed. Additional metrics are identified that has an impact on workload performance predictions. The proposed AWPP is compared with well-known machine learning techniques and is evaluated through evaluation measures that include effectiveness, accuracy, adaptiveness, and significance. Post-hoc test is performed to validate the results. The rest of the paper is organized as follows. Related work is presented in Section 2. The proposed AWPP framework is described in Section 3. The evaluation of the proposed AWPP framework using CBR and machine learning techniques is provided in Section 4. Results and discussions are presented in Section 5. The conclusion of the paper and future work is presented in Section 6.

## 2. Related work

Autonomic Computing (AC), is an important technology has been used in many application areas, including web services [21,22] and databases [24–26,28,29]. AC performs the tasks in a system autonomically without or less human intervention, i.e. self-management [5]. The concept of self-management is taken from the nervous system of human [3]. AC helps in predicting the behavior of a system which is also being used for workload performance prediction in DBMSs. Prediction is performed for forecasting the time ranges [41], throughput and response time [42]. Machine learning has been used for predicting the performance metrics of a workload [19]. The performance metrics that have been predicted include message bytes, message count, records used, records accessed, disk I/O, and elapsed time. KCCA algorithm is used for finding a correlation between query and performance metrics. The projection for new query is performed using KNN for finding the nearest neighbor that predict the performance metrics. Research is also carried out for modeling and predicting the workload performance [20]. Adaptiveness is achieved by researchers using different techniques and approaches such as fuzzy logic [27] that uses Fuzzy Inference System (FIS) to predict buffer-hit-ratio, database size, and the number of users. The studies Rosas et al. [7] provides an adaptive methodology for performance improvement in large data using data partition, processing nodes and adapting size.

A learning-based framework WiSeDB [9] has been proposed for workload management. It works on workload characteristics and performance goals. For resource provisioning and scheduling, decision tree approach is used to adapt offline model for retraining the performance goals with low training overheads. The adaptation is fast for performance assessment and cost trade-offs. The study Singhal and Nambiar [15] provided a modular approach for estimation of the execution time of an SQL query for high information volume. The studies Wu et al. [13,14] predicted SQL query execution plan and proposed prediction of accurate estimation of disk visits and CPU time for large data size. For CPU performance prediction, different models have been proposed such as query execution prediction model [8], performance model [9] and framework COMPASS [10]. PEMOGEN [11] predicts query response time that works on the neural system. The MAG framework is presented in [23] to control, monitor, predict and analyze the configuration parameters that works for performance problems of the database. Performance and resource analysis has been performed using a framework that predicts resource consumption, bottleneck, and throughput [40]. The work Hasan [12] provides the performance prediction of SPARQL query through machine learning techniques. Our work is different from the existing studies as it provides a framework that takes workload as input and predicts the performance of that workload and also handles the evolutionary behavior of the workload through adaptation without human intervention.

## 3. Proposed AWPP framework

In this section, we present the proposed Autonomic Workload Performance Prediction (AWPP) framework and describe its all components, input/output and autonomic functionality using CBR approach. The AWPP consists of three components as shown in Fig. 1. These components are *Features Extraction*, *Workload Performance Prediction*, and *Workload Adaptation. Workload feature extraction* and *Performance metrics extraction* are two sub-components of *Features Extraction* component. *Workload Performance Prediction* forecasts the performance of the workload using machine learning techniques. *Workload Adaptation* retains the workload on the change in behavior.

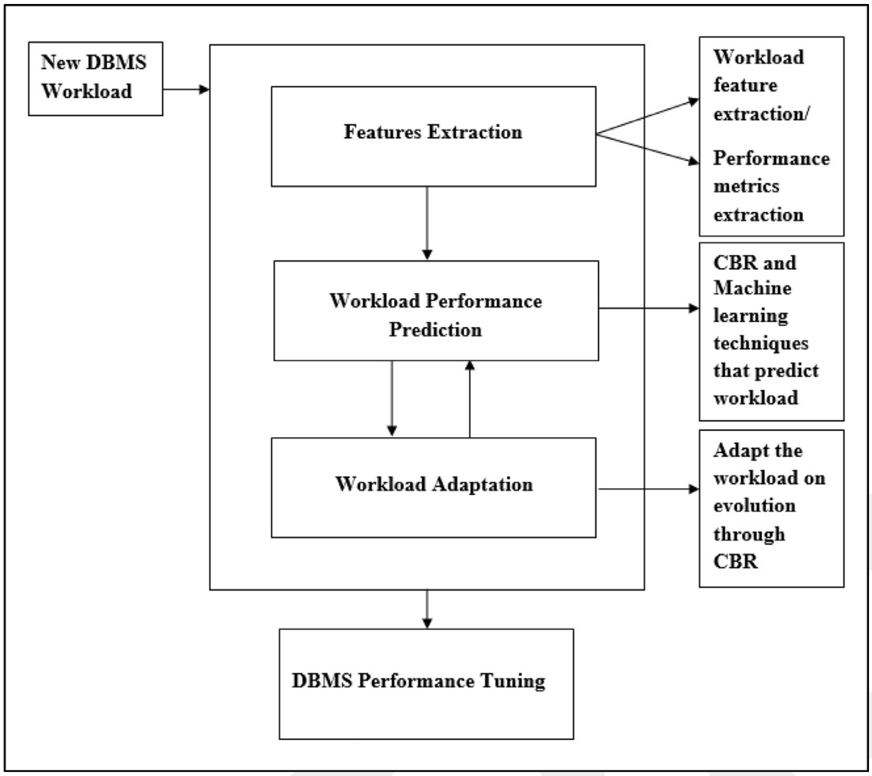


**Fig. 1.** Autonomic Workload Performance Prediction (AWPP) framework.

**Table 1**
WFV and PMV.

| Input WFV | Output PMV |
|---|---|
| 1. *Number of selection predicates,* | 1. *Key_read_requests,* |
| 2. *Number of equality selection predicates,* | 2. *Key_reads,* |
| 3. *Number of aggregation columns,* | 3. *Bytes_sent,* |
| 4. *Number of nested sub-queries,* | 4. *Bytes_received,* |
| 5. *Total number of join predicates,* | 5. *Query_cost,* |
| 6. *Number of sort columns,* | 6. *Execution time,* |
| 7. *Number of non-equality selection predicates.* | 7. *Workload size,* |
| | 8. *Key_writes,* |
| | 9. *Key_write_requests,* |
| | 10. *Innodb_dblwr_writes,* |
| | 11. *Innodb_dblwr_pages_written* |

MySQL database status variables have been studied and some of them are found to be useful for workload performance prediction which can help in optimizing internal algorithms of DBMSs. The Performance Metrics Vector (PMV) is selected on the basis of F-measure with threshold > 80% to observe its effectiveness. The input for *Workload feature extraction* is a workload which is represented as Workload Features Vector (WFV) and has seven features. The performance metrics are extracted through *Performance metrics extraction* which is represented as PMV and has eleven metrics as shown in Table 1.

The autonomic workload prediction and adaptation architecture are shown in Fig. 2 that is based on generic AC architecture [4]. We mapped the workload prediction and adaptation components of the AWPP framework with standard AC architecture. The workload entering the system acts as *Managed Element* and self-inspection is performed through *Sensor* and *Effector. Workload Performance Prediction* component uses the workload features extracted by *Features Extraction* component to find the correlation between WFV and PMV and self-prediction is performed. The *Workload Adaptation* component performs self-adaptation which adapts the behavior of the workload according to the changes occurred. Machine learning techniques support in prediction and adaptation, therefore, machine learning and reasoning techniques are investigated to obtain best prediction and adaptation results. This study uses CBR approach for predicting the performance metrics and adapting to the changing behavior. The CBR solves the new problems through reasoning with existing solutions that are stored in the case-base repository and consists of four phases which are *Retrieve, Reuse, Revise*, and *Retain* [35,36]. *Retrieve* phase fetches the similar cases. *Reuse* phase reuses the most similar retrieved case for the solution of the new case. The *Revise* phase updates the case according to the changing requirement and the *Retain* phase stores the revised case in the case-base for future use.

## 4. Evaluating AWPP using Case-Based Reasoning approach and machine learning techniques

This section describes the data modeling and evaluation of the proposed AWPP framework. For the evaluation of the AWPP framework, we have developed an algorithm based on the CBR approach, defined evaluation metrics and presented different similarity measures used in this work. The AWPP framework is validated using a lazy learning CBR approach and machine learning technique that is Support Vector Machine (SVM). The SVM is an existing well-known prediction technique. Through our proposed CBR approach, the predictions as well as adaptation of the database workload can be performed. Due to the adaptive nature of the CBR, it can best solve the workload management problem. To the best of our knowledge, these techniques have not been used earlier to solve workload management problems. For experiments, the training and testing data is prepared though executing OLTP and DSS workloads. The evaluation measures such as accuracy, effectiveness, adaptiveness, and significance (see Section 5) are used to validate the proposed AWPP framework. To predict and adapt the workload, data mod-

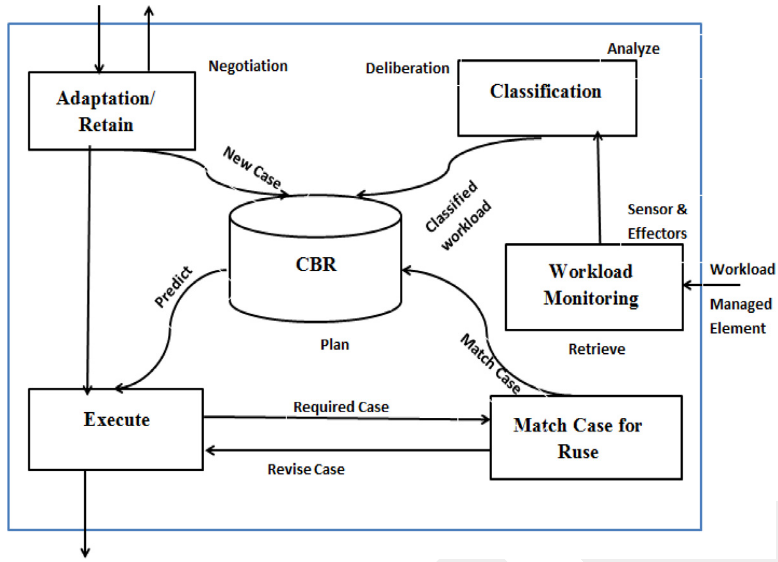


**Fig. 2.** Autonomic workload prediction and adaptation architecture.

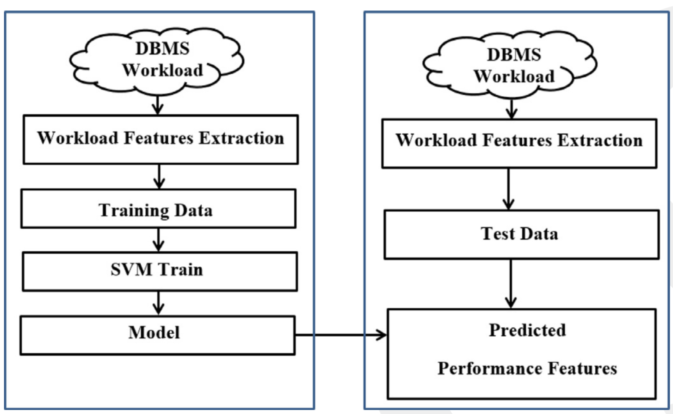


**Fig. 3.** SVM model.

eling through SVM and CBR is presented in Sections 4.1 and 4.2 respectively.

### 4.1. Data modeling through SVM

The SVM technique [32,33] learns through an example for performing classification tasks. After extracting the workload features, the training and testing data are prepared for workload performance prediction. The training data corresponding to the target data values. We train the data through SVM train, a model is generated which provides a prediction for test data, the adaptation could not be performed as SVM is not adaptive. For best classification parameters, we performed 10-fold cross-validations. Various types of 1000 workloads are used for experiments. We applied SVM technique to transform the data in the format specified for SVM and we scale on the data. Training of data is performed in MATLAB through LibSVM tool [34] that is used to solve classification problems. The process of SVM model is shown in Fig. 3. While using SVM, the selection of kernel is important. Kernels can handle linear or nonlinear relations between the class and attribute [33]. In this study, due to the existence of nonlinear relations, we selected Radial Basis Function (RBF) that maps nonlinear relation to higher dimensional space.

### 4.2. Data modeling through CBR

The CBR model is shown in Fig. 4. The CBR is a lazy learning technique which has the ability to adapt new cases in the case-base or knowledge base for future use. The case–base contains or stores the previous cases; similar to training data. In machine learning techniques, it is required to retrain the data. As CBR works on reasoning, so it does not need to re-train data. After extracting WFV, training and testing data are prepared for CBR model using MySQL through execution of a variety of workloads that are stored in the case-base. The knowledge-base stores the previous cases similar to training data that consists of Workload Features Vector (WFV) and Performance Metrics Vector (PMV). The new case represents the test workload. The Case-base is presented as a matrix. The row of the matrix corresponds to case and column corresponds to the features of WFV and PMV. The indexes are used to show cases in the case-base. For the new test case, first, it is matched with the cases of case-base through similarity measure between WFV and cases of training data. A number of distance formulas are applied to find the best similarity cases. The four CBR phases *Retrieve, Reuse, Revise*, and *Retain* are applied to perform workload prediction and adaptation. When exact matching of a new test case with training data is found through similarity, the cases are retrieved, and the most similar case is reused. In this way, the performance of the workload is predicted. When the similar matching does not exist or is lower than the threshold value, the case is revised and is retained by storing in the case-base repository for future use. This provides self-adaptiveness which is necessary for the evolving behavior of the DBMS workloads.

The algorithm for workload performance prediction and adaptation for AWPP using CBR approach is shown in as below.

Performance prediction and adaptation are evaluated by measuring the accuracy and effectiveness criteria [31]. We calculated accuracy and effectiveness (through F-measure) for the proposed AWPP framework. We executed 1000 queries in our experiments and used the values of WFV and PMV for creating the cases and build training and testing data. We labeled the classes and identified 7 classes based on data distribution. Actual classes are in training data and after running the algorithm on testing data we get predicted, classes. Table 2 shows the actual versus predicted class matrix. The rows correspond to actual class and column cor-

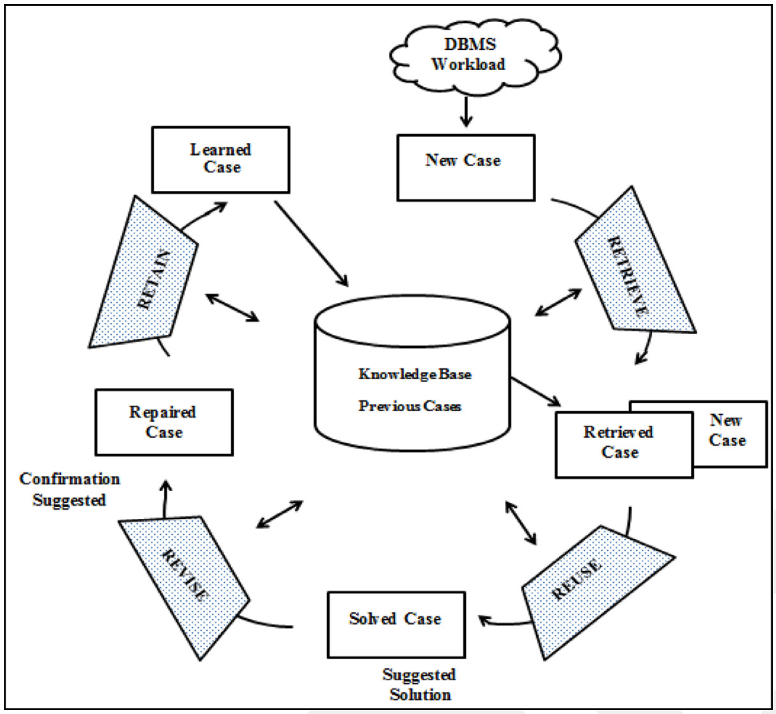


**Fig. 4.** CBR model.

**Table 2**
Actual vs. predicted class.

| | | Predicted class | |
|---|---|---|---|
| | | Positive class | Negative class |
| Actual class | Positive class | ***tp*** (true positive) Correct result | ***fn*** (false negative) Missing result |
| | Negative class | ***fp*** (false positive) Unexpected result | ***tn*** (true negative) Correct absence of result |

responds to the predicted class. The actual and predicted class can be positive or negative which means that when actual class is positive class and predicted class is positive class then this represents true positive ***tp***, i.e., we have correct result and when actual class is positive class and predicted class is negative class then this represent false negative ***fn***, i.e., missing result. Similarly, when the actual class is negative class and predicted is positive class then this represent false positive ***fp***, i.e., unexpected result and when actual class is negative class and predicted class is negative class then this represent true negative ***tn***, i.e., correct absence of result.

Where ***tp*** is true positive, ***fp*** is false positive, ***fn*** is false negative and ***tn*** is true negative

Eqs. (1)–(4) presents the Precision, Recall, F-measure and Accuracy formulas used.

Precision formula is as follows

$$\boldsymbol{Precision} = \frac{\boldsymbol{tp}}{\boldsymbol{tp} + \boldsymbol{fp}} \tag{1}$$

and Recall is calculated through the

$$\boldsymbol{Recall} = \frac{\boldsymbol{tp}}{\boldsymbol{tp} + \boldsymbol{fn}} \tag{2}$$

F-measure is calculated as

$$\boldsymbol{F - Measure} = 2.\frac{\boldsymbol{precision}\ .\ \boldsymbol{recall}}{\boldsymbol{precision} + \boldsymbol{recall}} \tag{3}$$

Accuracy is calculated as

$$\boldsymbol{Accuracy} = \frac{\boldsymbol{tp} + \boldsymbol{tn}}{\boldsymbol{tp} + \boldsymbol{tn} + \boldsymbol{fp} + \boldsymbol{fn}} \tag{4}$$

In Algorithm 1, we calculated the similarity between test case and training case of the case-base. The selection of a similarity measure depends on the type of data being used, how data is distributed and domain of the problem. To calculate the similarity, we have used different similarity and distance measures which are Spearman distance, Minkowski distance, Hamming distance, Euclidean distance, Correlation, Chebychev distance, Cosine distance, and Jaccard similarity.

The Spearman distance [53] is used to retrieve similar cases from the case-base. The Spearman distance is the square of the Euclidean distance between two WFVs where n denotes the total number of workload features. It calculates the distance that shows the similarity of the test case and the stored case in the case-base using the formula as shown in Eq. (5).

$$d_{ij} = \sum_{k=1}^{n} (x_{ik} - y_{jk})^2 \tag{5}$$

Where $\boldsymbol{d_{ij}}$ represents distance between $i$th and $j$th cases with respect to all features of WFV.

The Minkowski distance [53] defines a distance between two points in a normed vector space. There are two special cases for the value of p in Eq. (6), for p = 1 and p = 2, it works like the Manhattan distance and the Euclidean distance respectively where p is a positive integer. The Minkowski distance between the test case and case stored in the case-base is shown in Eq. (6).

$$d_{ij} = \left(\sum_{k=1}^{n} \left|x_{ik} - x_{jk}\right|^{p}\right)^{(1/p)} \tag{6}$$

Where $\boldsymbol{d_{ij}}$ represents distance between $i$th and $j$th cases with respect to all p features of WFV.

The Hamming distance [53] between two WFVs is equal to the number of positions where they have distinct values or where the coefficient of two cases, test case and training case, differ as shown

**Algorithm 1** Performance prediction and adaptation for AWPP.

***Input*: *Workload* (*WFV*)**
***Output*: *Performance metric* (*PFV*)**
***Extract features from input workload*.**
***Calculate similarity between all workload attributes of testing data and attributes of training data stored in case – base*.**
***If matching cases found in the case – base // Retrieve is performed***
    ***If similarity percentage*** > 80
        ***Reuse is performed and Performance metrics (PFV) are predicted***
    ***Else if similarity percentage*** ⟨80 ***and*** ⟩60
        ***Revise is performed after learning and Performance metrics (PFV) are predicted***
        ***Adapt the revise case for future use || Retain is performed***
    ***End if***
***End if || Prediction solution does not exist*.**

in Eq. (7).

$$\boldsymbol{d_{ij}} = \sum_{\boldsymbol{k}=0}^{\boldsymbol{n}-1} \left[ \boldsymbol{y}_{\boldsymbol{i},\boldsymbol{k}} \neq \boldsymbol{y}_{\boldsymbol{j},\boldsymbol{k}} \right] \tag{7}$$

Where $\boldsymbol{d_{ij}}$ represents distance between $i$th and $j$th cases with respect to all features of WFV.

Similarly, the Euclidean distance [54] is applied as a similarity measure between the current case and any previous case of case-base. It calculates the distance by using the Euclidean distance formula as shown in Eq. (8).

$$\boldsymbol{d_{ij}} = \sqrt{(\boldsymbol{xi}_1 - \boldsymbol{xj}_1)^2 + (\boldsymbol{xi}_2 - \boldsymbol{xj}_2)^2 + \ldots + (\boldsymbol{xi_n} - \boldsymbol{xj_n})^2} \tag{8}$$

Where i = ($\boldsymbol{xi}_1$, $\boldsymbol{xi}_2$, ..., $\boldsymbol{xi_n}$) and j = ($\boldsymbol{xj}_1$, $\boldsymbol{xj}_2$, ..., $\boldsymbol{xj_n}$) are two n-dimensional data objects and $\boldsymbol{d_{ij}}$ represents distance between $i$th and $j$th cases with respect to all features of WFV.

The Correlation coefficient [53] is another similarity measurement rather than distance measurement, which measures the angular separation having values in the range [−1, +1] between the test case and stored cases of the case-base, as shown in Eq. (9).

$$\boldsymbol{S_{ij}} = \frac{\sum_{k=1}^{n} (x_{ik} - \overline{x_i})(x_{jk} - \overline{x_j})}{\sqrt{\sum_{k=1}^{n} (x_{ik} - \overline{x_i})^2 \sum_{r=1}^{n} (x_{jr} - \overline{x_j})^2}} \tag{9}$$

Where $\overline{x_i} = \frac{1}{n} \sum_{k=1}^{n} \overline{x_{ik}}$ and $\overline{x_j} = \frac{1}{n} \sum_{k=1}^{n} \overline{x_{jk}}$ and $d_{ij}$ represents distance between $i$th and $j$th cases with respect to all features of WFV.

The Chebychev distance [53] calculates the maximum of absolute differences between the test case and the cases stored in the case-base using the formula as shown in Eq. (10).

$$d_{ij} = {}_{k}^{max}\left( \left| x_{ik} - x_{jk} \right| \right) \tag{10}$$

Where $d_{ij}$ represents distance between $i$th and $j$th cases with respect to all features of WFV.

The Cosine distance similarity measures the cosine angle between the test case and stored case of the case-base. The value of Cosine function is 1 when the angle is 0 and Cosine value is less than 1 for other values of angle. When the angle between the WFVs shortens, the cosine value approaches 1, meaning that the two WFVs are getting closer, the similarity of WFVs increases [53] as shown in Eq. (11).

$$S_{ij} = \frac{\sum_{k=1}^{n} x_{ik} x_{jk}}{\sqrt{\sum_{k=1}^{n} (x_{ik})^2 \sum_{r=1}^{n} (x_{jr})^2}} \tag{11}$$

Where $s_{ij}$ represents similarity between $i$th and $j$th cases with respect to all features of WFV.

The Jaccard distance similarity [54] measures dissimilarity between test case and the stored case of the case-base by the size of the intersection divided by the size of the union of the WFVs as shown in Eq. (12).

$$S_{ij} = \left| \frac{x_i \cap x_j}{x_i \cup x_j} \right| \tag{12}$$

Where $s_{ij}$ represents similarity between $i$th and $j$th cases with respect to all features of WFV.

## 5. Results and discussion

This section provides the details of the setup developed for the experiments. The experiments were performed on Windows 7, processor Cor 2 Duo 3.0 GHz and hard disk of 280GB, 2GB RAM. Database MySQL version 5.1 [37] is used in this work. The standard dataset TPC-H of size 1GB is used that is defined by Transaction Processing Council (TPC) [38]. Machine learning and lazy learning techniques such as CBR, SVM, and others are used to perform experiments and we used the same dataset for all the classification techniques. The TPC-H and TPC-C like data are generated using DB-Gen (Dbt2) for populating the workload [39]. The TPC-H and TPC-C like queries were the representative workloads that were executed. For our experiments, more than 1000 workloads were executed on MySQL database and noted the status variable values. We used 22 queries which are fixed distinct the TPC-H standard queries in our experiments available in [55] and also the TPC-C transactions available in [56]. For extending our experiments to build a dataset of 1000 queries, we designed and executed similar queries, we called them the TPC-H like queries, by varying selection predicates, many variations of joins, aggregations, different constraints etc. After execution of queries, we obtained the status variables of the WFV and PMV. The training and testing data is prepared from WFV and PMV and developed the case-base. The case consists of WFV and corresponding PMV class to whom it belongs. We defined the classes in the training and testing data for the performance metrics. Section 5.1 presents the experiments and results of the workload performance prediction and Section 5.2 presents the results of the workload adaptation through SVM and CBR approach. In the end, ranking of existing classifiers including the proposed CBR approach is provided and post hoc tests are performed for validating the results.

### 5.1. Workload performance prediction

To predict the workload performance prediction in AWPP, we used machine learning techniques. The classification is applied for workload performance prediction. We prepared training and testing by executing 1000 queries. We make the data distribution into 7 classes and then divided 70% training and 30% testing data. We applied classification and prediction on training and testing for workload performance metrics predictions. Classification problem can be a binary class or multi-class. As in this work, there are 7 classes, so this corresponds to the multi-class problem and there

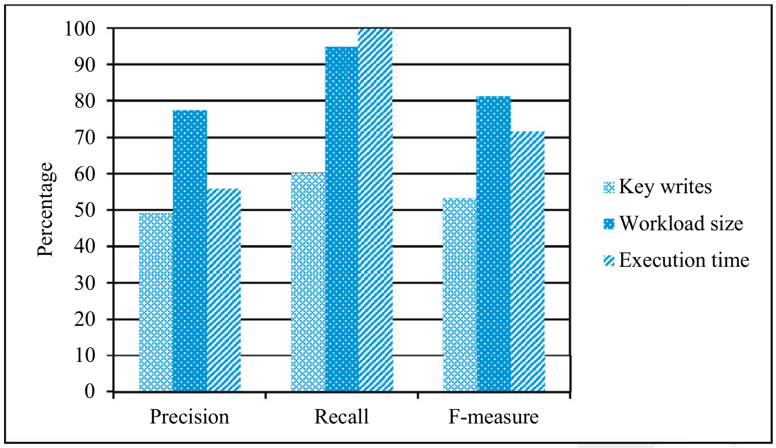


**Fig. 5.** Precision, recall and F-measure using SVM.

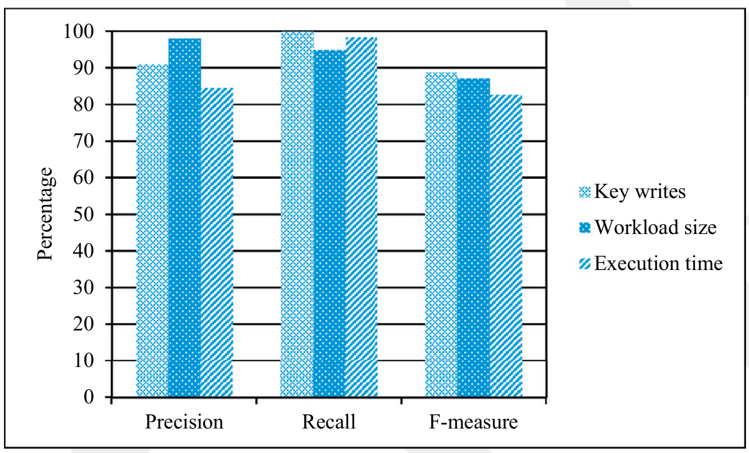


**Fig. 6.** Precision, recall and F-measure using CBR.

are many approaches to handle classification tasks of multi-class nature [45]. One of the approaches is building one-verses-all and other is one-verses-one. In one-verses-one, the target class becomes that class which is selected by maximum classifiers and due to this, we consider this approach. Our PMV consists of 11 performance metrics. We obtained the optimization parameters by performing 10-fold cross-validation through LibSVM using RBF kernel. The parameter obtained are C = 8, RBF kernel gamma parameter - G = 0.0625 and training is performed. The both Sections 5.1.1 and 5.1.2 present experiments performed using SVM and CBR approach respectively. The Section 5.1.3 provides a comparison of the proposed CBR approach with a number of machine learning techniques.

#### 5.1.1. Experiments performed through SVM

The experiments were performed on 11 performance metrics using SVM technique and precision, recall, and f-measure are calculated. Here we are presenting only three metrics which are *Key writes, Workload size,* and *Execution time* as shown in Fig. 5. The precision, recall, and f-measure for the metric *Key writes* were recorded 49%, 60%, and 53% respectively. For the *Workload size* the values for precision, recall, and f-measure were 77%, 95%, and 81% respectively. Similarly, for the metric *Execution time* 56%, 100%, and 72% were the values for the precision, recall, and f-measure respectively. The precision, recall, and f-measure for the three selected performance metrics produced good results using the well-known SVM classification technique.

#### 5.1.2. Experiments performed through proposed CBR

To compare the SVM results with our proposed approach, the experiments were performed on the proposed CBR approach for workload performance prediction and adaptation. Here we are presenting the same metric *Key writes, Workload size,* and *Execution time* as shown in Fig. 6. For the *Key writes* metric, the precision, recall, and f-measure were 90%, 100%, and 89% respectively. The precision, recall, and f-measure for the *Workload size* metric were 98%, 95%, and 87% respectively and for the *Execution time* 84% precision, 98% recall and 83% f-measure were recorded. It can be seen that by using the proposed CBR approach, precision, recall and f-measure values are increased and produced better results than the well-known SVM classification technique. The reason behind is that the CBR works on reasoning with the existing cases whereas the SVM works on the learning which is performed once.

We compared the f-measure of performance metrics obtained through SVM and proposed CBR as shown in Fig. 7 and it can be seen that in comparison with SVM the proposed CBR produced the better results. All the performance metrics were effective, some metrics were more effective as they yield 100% f-measure that are *Query cost, Byte received* and *Key read*. Likewise, other metrics are approaching 80% or above. The proposed CBR approach produced better results due to its lazy learning ability, as it stores the training data in the form of solution cases in the case-base and reasoning is performed by the test case. However, it works slowly because it has to perform reasoning for every test case. The data is localized in CBR, therefore, generalization takes time for every turn. In eager learning, like SVM, the learning data model is created once at the start and performs predictions on the basis of

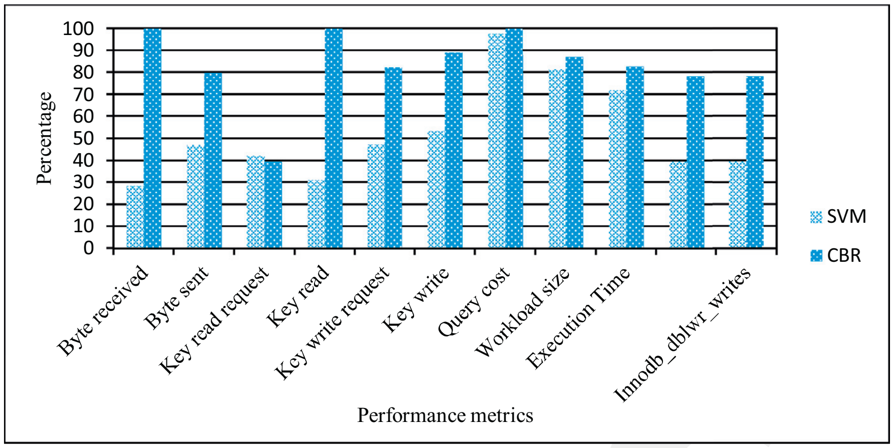


**Fig. 7.** F-measure comparison between CBR and SVM.

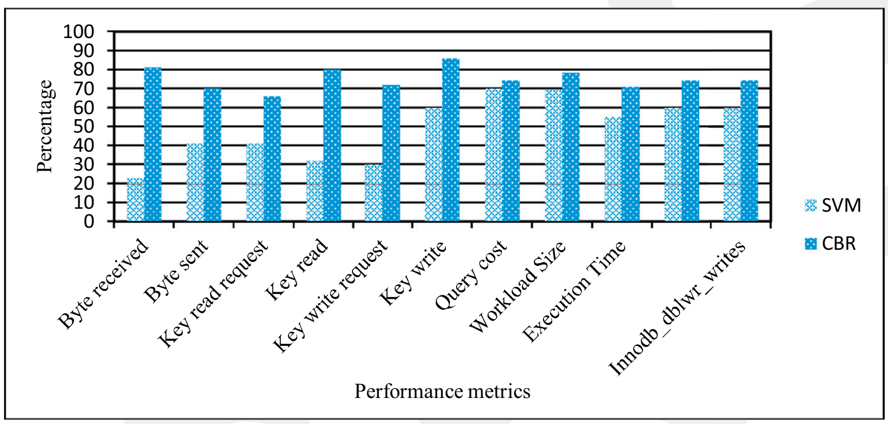


**Fig. 8.** Accuracy comparison between CBR and SVM.

**Table 3**
Performance measure comparison of *Key writes* through different algorithms.

| Algorithm/Performance measure | Simple Cart | Naive Bayes | Bayes Net | J48 | SVM | CBR |
|---|---|---|---|---|---|---|
| Precision (%) | 53 | 26 | 43 | 59 | 49 | 91 |
| Recall (%) | 71 | 35 | 46 | 70 | 60 | 100 |
| F-measure (%) | 56 | 28 | 38 | 61 | 53 | 89 |
| Accuracy (%) | 53 | 47 | 53 | 57 | 60 | 86 |

that learning. The CBR works well for a period of time, after that its performance may degrade. To improve the learning method, the calibration of the learnt model could improve the performance of proposed CBR. The variation of F-measure can be seen among all performance metrics because of data generated through execution of a variety of queries or workloads of different nature and complexities have a different distribution of values. The attributes having higher values depict that they have higher *tp* values and less *tn*, *fn*, and *fp* values (Table 2).

The accuracy of the CBR approach is also calculated as shown in Fig. 8 We used a number of distance formulas and seen that cosine distance provided better results. It is observed that in comparison with SVM the CBR results were more accurate in predicting the performance of the workload. Overall the accuracy was more than 70% in most performance metrics and some metrics such as *Byte received, Key Read* and *Key writes* have 92%, 76%, and 88% accuracy.

### 5.1.3. Proposed CBR approach comparison with other machine learning approaches

For further investigation, we compared the proposed CBR approach with other machine learning techniques which includes Naïve Bayes (NB), Simple Cart (SC), J48, and Bayes Net (BN). The Weka tool [43,44] is used for applying other machine learning techniques. For training and testing data, 10-fold cross-validation is used. Tables 3–5 provided the performance measure comparison of the three selected metrics *Key writes, Workload size* and *Execution time.* For the performance, metric *Key write* the precision, recall and f-measure produced by CBR were the best f-measure 89% and accuracy 86%, however, J48 and SVM results were satisfactory. For the performance metrics *Workload size* the Bayes net algorithm produced 73% f-measure and 62% accuracy and through SVM the f-measure and accuracy recorded 81% and 60%, however, CBR provided the best measures that is 89% f-measure and 86% accuracy. The CBR, due to its continuous learning capability produced the best results which are 83%and 71% f-measure and accuracy for *Execution time* in comparison with J48 algorithm produced 73% and 57% f-measure and accuracy respectively. We can see that overall the proposed CBR has the best results in term of precision, recall, f-measure, and accuracy in comparison with other selected machine learning techniques.

In CBR appraoch, we tested all the distance formulas as stated in Section 4.2 for getting the best similarity results and obtained the precision, recall, and f-measure for the performance metrics. In this work, we observed that in comparison with other distance

**Table 4**
Performance measure comparison of *Workload size* through different algorithms.

| Algorithm/Performance measure | Simple Cart | Naive Bayes | Bayes Net | J48 | SVM | CBR |
|---|---|---|---|---|---|---|
| Precision (%) | 57 | 59 | 75 | 56 | 77 | 98 |
| Recall (%) | 70 | 94 | 79 | 67 | 95 | 95 |
| F-measure (%) | 60 | 71 | 73 | 59 | 81 | 87 |
| Accuracy (%) | 63 | 60 | 62 | 60 | 69 | 83 |

**Table 5**
Performance measure comparison of *Execution time* through different algorithms.

| Algorithm/Performance measure | Simple Cart | Naive Bayes | Bayes Net | J48 | SVM | CBR |
|---|---|---|---|---|---|---|
| Precision (%) | 57 | 57 | 61 | 57 | 56 | 84 |
| Recall (%) | 59 | 100 | 58 | 100 | 100 | 98 |
| F-measure (%) | 54 | 73 | 53 | 73 | 72 | 83 |
| Accuracy (%) | 58 | 57 | 59 | 57 | 55 | 71 |

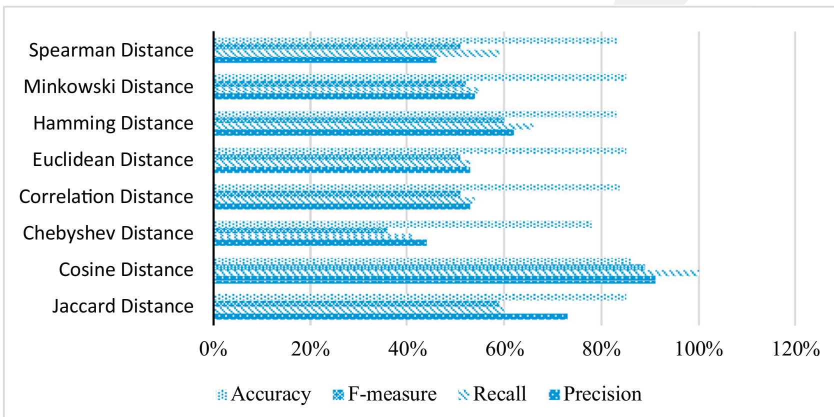


**Fig. 9.** Performance measure comparison of *Key writes* through different similarity measures.

formulas the Cosine similarity has produced better results. As in our study, due to high dimensional data, we have used RBF kernel in SVM modeling for handling the high dimensional data. For handling high dimensional data in the proposed CBR modeling, the Cosine similarity performed better in comparison with other distance and similarity measures due to the angular value of Cosine, when the angular value is small, the cases have high similarity. However, the Euclidean distance is not good for the high dimensional data. Further, the data is normalized to the interval [1, −1], therefore, the cosine performed well as compared to other distance and similarity measured used in this study. Moreover, when the two WFVs are common, the Cosine similarity also affects. In Cosine similarity, the number of common features is divided by the total number of possible features, whereas in Jaccard similarity, the number of common features is divided by the number of features that exist in at least one of the two WFVs. In Cosine similarity, when the dimensionality grows the correlation distribution between WFVs gets decreasing and may approach to zero (that indicates high similarity). For growing dimensionality, the significance of small correlations increases. Cosine similarity is good at capturing the similarity of patterns of feature changes, at the same time disregarding the absolute amplitude of the compared feature vectors. Due to high dimensionality of data and data distribution in our study, the Cosine similarity produced the best measures such as 91% precision, 100% recall, 89% f-measure, and 86% accuracy, however, other similarity measures have produced accuracy about 80% and precision, recall and f-measure values about 60% or less which is not good as shown in Fig. 9.

The results of the performance metric *Workload Size* show that the Cosine similarity measure produced the best results which are 98% precision, 95% recall, 87% f-measure, and 83% accuracy as compared to other similarity measures that produced very small values about 40% such as Minkowski, Hamming, Euclidean and Chebyshev which scored lowest in our work as shown in Fig. 10.

Similarly, the results of the Cosine distance similarity for the performance metric *Execution time* show the best precision, recall, f-measure, and accuracy as 84%, 98%, 83% and 71% respectively whereas for most of the other similarity measure the performance measure values were recorded about 40% that is lowest in our study as shown in Fig. 11, that is why they are not appropriate to be selected as similarity measure. For our proposed CBR approach, we selected Cosine similarity measure, as it produced better results for all the performance metrics. The Cosine similarity measures the Cosine angle between two WFVs of test case within the case-base. The value of Cosine function is 1 when the angle is 0 and less than 1 for other value of angle. The cosine angel approaches 1 when the angle between the vectors shortens which means that the similarity of two vectors increases. In our case, the nature of data allows Cosine similarity to best suit it and produces the maximum similarity results.

### 5.2. Workload adaptation

For workload adaptation, a number of experiments were performed by applying SVM. For SVM experimentation, we used the LibSVM tool. SVM is a good classification technique, however, it has no adaptation ability. The CBR having adaptation ability in nature is used in the experiments. A number of workloads are executed using SVM and CBR and f-measure is observed before and after the adaptation. Before and after adaptation, values of the f-measure using the CBR is recorded as 79% and 100% in comparison with the

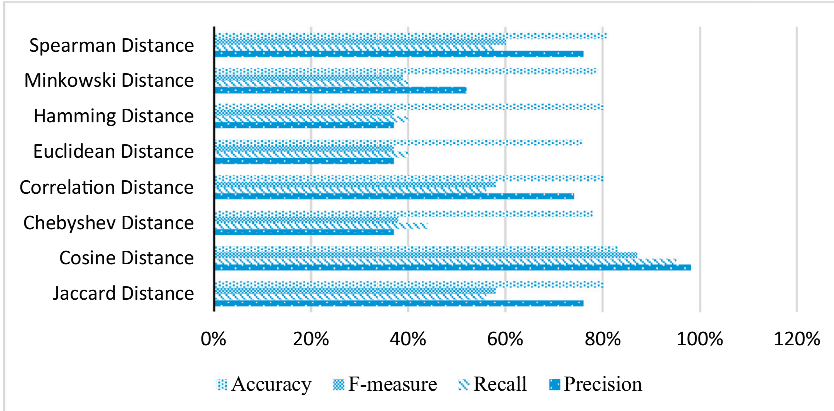


**Fig. 10.** Performance measure comparison of *Workload size* through different similarity measures.

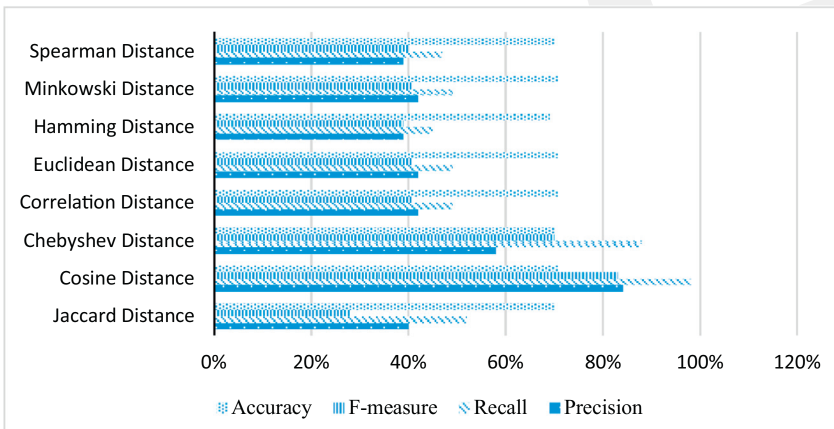


**Fig. 11.** Performance measure comparison of *Execution time* through different similarity measures.

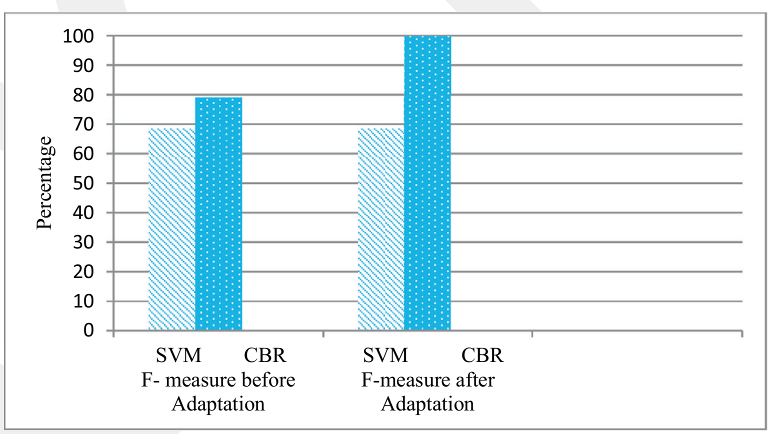


**Fig. 12.** Adaptation comparison through CBR and SVM.

values of f-measure for the SVM both as 69% respectively as shown in Fig. 12, which depicts that our proposed CBR approach achieved the adaptive capability for workload management.

Similarity, increase in the accuracy of workload adaptation before and after adaptation is observed as shown in Fig. 13.

Before adaptation, the accuracy is recorded as 58% and 68% for SVM and CBR respectively. After adaptation, the accuracy of CBR increased to 83% in comparison with SVM which is not adaptive. An increase of 15% is observed after adaptation. We observed that CBR outperformed the machine learning techniques in performance prediction and adaptation. Our contribution is that the proposed CBR approach produced better results for prediction and adaptation in comparison with the well-known machine learning techniques in terms of effectiveness and accuracy.

For proposing a new method all pairwise comparison is useful to perform on all classifiers [47,48]. For the post-hoc test, non-parametric procedures are performed for detecting pairwise difference significantly. Non-parametric tests are strong tests as compared to parametric test because the normal distribution of variance is not assumed in non-parametric and it rejects null hypoth-

**Table 6**
Comparison of workload performance metrics and classifiers.

| Metrics\ Classifiers | Byte received | Byte sent | Key read Key read request | Key Key read | Key-write request | Key-write | Query cost | WL Size | ET | Innodb_dblwr_ pages_written | Innodb_ dblwr_writes |
|---|---|---|---|---|---|---|---|---|---|---|---|
| CBR | 1.000 | 0.212 | 0.114 | 0.240 | 0.136 | 0.592 | 0.305 | 0.579 | 0.206 | 0.171 | 0.171 |
| SVM | 1.000 | 0.350 | 0.240 | 0.280 | 0.310 | 0.660 | 0.360 | 0.680 | 0.240 | 0.240 | 0.170 |
| Simple Cart | 0.379 | 0.453 | 0.597 | 0.379 | 0.379 | 0.658 | 0.858 | 0.604 | 0.537 | 0.951 | 0.951 |
| Naive Bayes | 0.375 | 0.346 | 0.494 | 0.375 | 0.375 | 0.685 | 0.858 | 0.714 | 0.726 | 0.951 | 0.951 |
| Bayes Net | 0.378 | 0.512 | 0.568 | 0.378 | 0.378 | 0.376 | 0.858 | 0.733 | 0.529 | 0.951 | 0.951 |
| J48 | 0.397 | 0.541 | 0.646 | 0.397 | 0.397 | 0.605 | 0.858 | 0.593 | 0.726 | 0.951 | 0.951 |

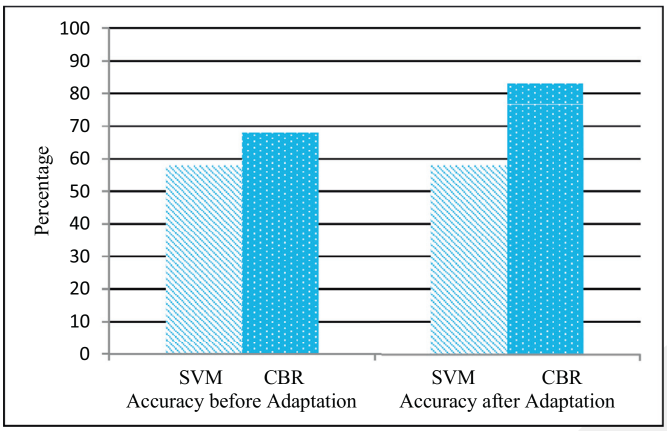


**Fig. 13.** Accuracy comparison of CBR and SVM.

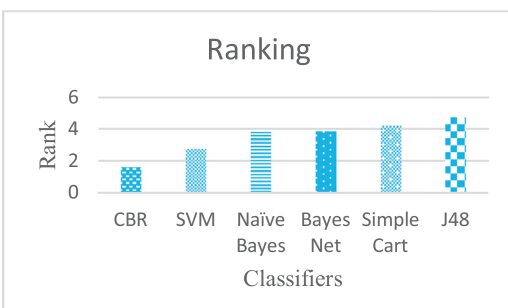


**Fig. 14.** Algorithm and Ranking by Friedman procedure.

**Table 7**
Best, average, worst classifiers.

| | |
|---|---|
| Best | CBR |
| Average | BN, NB, and SVM |
| Worst | SC and J48 |

esis more often. For the performing experiments, we used KEEL software [49] that takes input in CSV format and give output in the form of LATEX document. The 6 classifiers including CBR and 11 performance metrics are compared as shown in Table 6. We noted the hypothesis rejection and Adjusted $p$-value (APV).

The classifiers are compared to obtain the average ranks that provide a fair comparison of the classifiers [46]. The non-parametric procedure, Friedman test [50,51] provides a ranking of classifiers in a way that best classifiers get rank one and then two and three etc. The Friedman procedure is used to calculate the average ranks as shown in Fig. 14.

Based on the ranking values taken from Fig. 14, three types of classifiers that are Best, Average and Worst are shown in Table 7 which show that CBR is the best classifier among them.

For testing *nxn* comparison, the post-hoc and the Friedman tests are applied that provides more information about the hypotheses and the corresponding $p$-values ae computed. The Tables 8(a) and 8(b) show the $p$-values for $\alpha = 0.05$ for Shaffer and Holm procedures. The corresponding probability ($p$-value) can be found from normal distribution table through z value and is compared with level of significance $\alpha$. The test difference is due to the $\alpha$ value adjustment for compensation for many comparisons.

We made a null hypothesis which states that the performance of classifiers is same and observed differences are random. We divided the variability such as variability between the data sets, variability among the classifiers and error variability. The null hypothesis is rejected when variability between the classifiers is greater than the error variability. Due to the existence of a difference between the classifiers, for finding the actual difference post-hoc test is being performed.

The Holm, Nemenyi, Shaffer, and Bergman procedures obtained the adjusted $p$-value (APV) through adjustment of $p$-values. The Tables 9(a) and 9(b) show the adjusted $p$-values and provide information whether the hypothesis will be retained or rejected. In this study, in our example, the null hypotheses are rejected by Holm, Nemenyi, Shaffer, and Bergman procedures. The hypotheses with unadjusted $p$-value 0.003846 are rejected by Holm's procedure. The hypotheses with unadjusted $p$-value 0.003333 are rejected by Shaffer's procedure. Similarly, the hypotheses with unadjusted $p$-value 0.003333 are rejected by the Nemenyi's procedure. The hypotheses ($i = 12–15$) which are CBR vs. Bayes Net, CBR vs. Naive Bayes, CBR vs. J48 and CBR vs. Simple Cart are rejected by Bergmann's procedure. The hypothesis is significant or not can be determined by $p$-value. For the small $p$-value, the evidence becomes strong against the null hypothesis. The $p$-value provides the probability error within a comparison and does not consider other comparisons of the family. However, the APV considers all test are performed and APV can be compared with significant level.

## 6. Conclusions and future work

This study provides an Autonomic Workload Performance Prediction (AWPP) framework for performance prediction and adaptation of the DBMS workload. We mapped our workload management problem to the standard AC architecture to make it autonomic by incorporating three AC characteristics, self-inspection, self-prediction, and self-adaptation. In this way, the burden of DBA could be relieved using autonomic characteristics of the AWPP framework. DBA just performs initial training of data and other activities of the workload management are handled autonomically, so DBA can find more time to perform other useful DBMS activities. The CBR approach is applied to the AWPP framework and is compared with machine learning techniques such as SVM, Naive Bayes, and Simple Cart etc. The results are validated by applying advanced testing procedures and ranks of classifiers are obtained using the Friedman test which declared the CBR as the best classifier. The results show that the proposed CBR approach for the AWPP framework outperformed the SVM and other machine learn-

**Table 8 (a)**
$p$-values for $\alpha = 0.05$ for Shaffer and Holm.

| i<br>Hypothesis | 1<br>CBR vs. J48 | 2<br>CBR vs. SC | 3<br>CBR vs. BN | 4<br>CBR vs. NB | 5<br>SVM vs. J48 | 6<br>SVM vs. SC | 7<br>CBR vs. SVM |
|---|---|---|---|---|---|---|---|
| z = (R0-Ri)/SE | 3.932 | 3.305 | 2.849 | 2.792 | 2.450 | 1.823 | 1.481 |
| $p$-value | 0.000 | 0.001 | 0.004 | 0.005 | 0.014 | 0.068 | 0.138 |
| $p$-Holm | 0.003 | 0.004 | 0.004 | 0.004 | 0.005 | 0.005 | 0.006 |
| $p$-Shaffer | 0.003 | 0.005 | 0.005 | 0.005 | 0.005 | 0.005 | 0.006 |

**Table 8 (b)**
$p$-values for $\alpha = 0.05$ for Shaffer and Holm.

| i<br>Hypothesis | 8<br>SVM vs. BN | 9<br>SVM vs. NB | 10<br>NB vs. J48 | 11<br>BN vs. J48 | 12<br>SC vs. J48 | 13<br>SC vs. NB | 14<br>SC vs. BN | 15<br>NB vs. BN |
|---|---|---|---|---|---|---|---|---|
| z = (R0-Ri)/SE | 1.368 | 1.311 | 1.140 | 1.083 | 0.627 | 0.513 | 0.456 | 0.057 |
| $p$-value | 0.171 | 0.190 | 0.254 | 0.279 | 0.531 | 0.608 | 0.649 | 0.955 |
| $p$-Holm | 0.006 | 0.007 | 0.008 | 0.010 | 0.013 | 0.017 | 0.025 | 0.050 |
| $p$-Shaffer | 0.006 | 0.007 | 0.008 | 0.010 | 0.013 | 0.017 | 0.025 | 0.050 |

**Table 9 (a)**
APVs calculated by Bergman, Holm, Shaffer, and Nemenyi.

| i<br>Hypothesis | 1<br>CBR vs. J48 | 2<br>CBR vs. SC | 3<br>CBR vs. BN | 4<br>CBR vs. NB | 5<br>SVM vs. J48 | 6<br>SVM vs. SC | 7<br>CBR vs. SVM |
|---|---|---|---|---|---|---|---|
| $P$-value | 0.000 | 0.001 | 0.004 | 0.005 | 0.014 | 0.068 | 0.138 |
| APV-Holm | 0.001 | 0.013 | 0.057 | 0.063 | 0.157 | 0.682 | 1.246 |
| APV- Nemenyi | 0.001 | 0.014 | 0.066 | 0.079 | 0.214 | 1.024 | 2.077 |
| APV-Shaffer | 0.001 | 0.010 | 0.044 | 0.052 | 0.143 | 0.682 | 0.969 |
| APV-Bergman | 0.001 | 0.010 | 0.031 | 0.037 | 0.143 | 0.409 | 0.969 |

**Table 9 (b)**
APVs calculated by Bergman, Holm, Shaffer, and Nemenyi.

| i<br>Hypothesis | 8<br>SVM vs. BN | 9<br>SVM vs. NB | 10<br>NB vs. J48 | 11<br>BN vs. J48 | 12<br>SC vs. J48 | 13<br>SC vs. NB | 14<br>SC vs. BN | 15<br>NB vs. BN |
|---|---|---|---|---|---|---|---|---|
| $P$-value | 0.171 | 0.190 | 0.254 | 0.279 | 0.531 | 0.608 | 0.649 | 0.955 |
| APV-Holm | 1.372 | 1.372 | 1.527 | 1.527 | 2.123 | 2.123 | 2.123 | 2.123 |
| APV- Nemenyi | 2.572 | 2.850 | 3.817 | 4.185 | 7.962 | 9.121 | 9.728 | 14.318 |
| APV-Shaffer | 1.200 | 1.330 | 1.527 | 1.527 | 2.123 | 2.123 | 2.123 | 2.123 |
| APV-Bergman | 0.969 | 0.969 | 1.527 | 1.527 | 1.527 | 1.824 | 1.824 | 1.824 |

ing techniques for workload prediction and adaptation. Since the CBR model can adapt new cases and it does not require retraining of data in contrast to the SVM model, thus it best fits our problem.

Our future work includes autonomic maintenance and calibration of case-base to control its growing size. For the calibration of learning model other machine learning techniques need to be investigated. Additional autonomic characteristics such as self-configuration, self-healing and self-optimization can be used to enhance the functionality of the CBR model. The AWPP framework could be tested with other DBMSs, such as DB2, SQL Server and Oracle to examine their performance. The AWPP framework could also be tested for performance tuning in the data warehouse environment. This study initiates a new step towards predictive and adaptive autonomic framework for DBMS workload. It also opened the doors for research enhancements in the proposed framework.

## Acknowledgments

This research work is supported by COMSATS Institute of Information Technology (CIIT) Islamabad, Pakistan through research productivity funds. We also acknowledge the respectable anonymous reviewers for their valuable suggestions and comments that helped us to improve the quality of the paper.

## References

[1] A. Mateen, B. Raza, M. Sher, M.M. Awais, N. Mustapha, Workload management: a technology perspective with respect to self-* characteristics, Artif. Intell. Rev. 41 (4) (2014) 463–489.

[2] B. Raza, A. Mateen, M.M. Awais, M. Sher, Survey on autonomic workload management: algorithms, techniques, and models, J. Comput. 3 (7) (2011) 29–38.

[3] M. Parashar, H. Salim, Autonomic Computing: An Overview, in: LNCS, 3566, Springer-Verlag, Berlin Heidelberg, 2005, pp. 247–259.

[4] J. Koehler, C. Giblin, D. Gantenbein, R. Hauser, On Autonomic Computing Architectures, 2003 IBM Research, Zurich Research Laboratory.

[5] S.R. White, J.E. Hanson, I. Whalley, D.M. Chess, J.O. Kephart, An architectural approach to autonomic computing, in: Proceedings of the IEEE International Conference on Autonomic Computing (ICAC'04), 2004, pp. 2–9.

[6] A. Ahmed, D.J. Brown, A. Gegov, Dynamic resource allocation through workload prediction for energy efficient computing, in: Advances in Computational Intelligence Systems, Springer International Publishing, 2017, pp. 35–44.

[7] C. Rosas, A. Sikora, J. Jorba, A. Moreno, E. César, Improving performance on data-intensive applications using a load balancing methodology based on divisible load theory, Int. J. Parallel Program. (2014) 1–25.

[8] C. Chi, Y. Zhou, X. Ye, Performance prediction for performance-sensitive queries based on algorithmic complexity, Tsinghua Sci. Technol. 18 (6) (2013) 618–628.

[9] S. Venkataraman, Z. Yang, M. Franklin, B. Recht, I. Stoica, Ernest: efficient performance prediction for large-scale advanced analytics, in: 13th USENIX Symposium on Networked Systems Design and Implementation (NSDI 16), USENIX Association, 2016, pp. 363–378.

[10] S. Lee, J.S. Meredith, J.S. Vetter, Compass: A framework for automated performance modeling and prediction, in: Proceedings of the 29th ACM on International Conference on Supercomputing, ACM, 2015, pp. 405–414.

[11] A. Bhattacharyya, T. Hoefler, in: Pemogen: Automatic Adaptive Performance Modeling During Program Runtime, PACT '14, ACM, New York, NY, USA, 2014, pp. 393–404.

[12] R. Hasan, Predicting SPARQL query performance and explaining linked data, in: European Semantic Web Conference, Springer International Publishing, 2014, pp. 795–805.

[13] W. Wu, Y. Chi, S. Zhu, J. Tatemura, H. Hacigümüs, J.F. Naughton, Predicting query execution time: Are optimizer cost models really unusable? in: IEEE 29th International Conference on Data Engineering (ICDE), 2013, pp. 1081–1092.

[14] W. Wu, Y. Chi, H. Hacígümüş, J.F. Naughton, Towards predicting query execution time for concurrent and dynamic database workloads, Proc. VLDB Endowment 6 (10) (2013) 925–936.
[15] R. Singhal, M. Nambiar, Predicting SQL query execution time for large data volume, in: Proceedings of the 20th International Database Engineering & Applications Symposium, ACM., 2016, pp. 378–385.
[16] S. Elnaffar, P. Martin, R. Horman, Automatically classifying database workloads, in: ACM Proceeding of the Conference on Information and Knowledge Management, 2002, p. 2.
[17] H. Li, X. Hou, M. Chen, Z. Dai, M. Zhu, M. Huang, Performance prediction for concurrent workloads in distributed database systems, in: International Conference on Algorithms and Architectures for Parallel Processing, Springer International Publishing, 2015, pp. 626–639.
[18] N.R. Herbst, N. Huber, S. Kounev, E. Amrehn, Self-adaptive workload classification and forecasting for proactive resource provisioning, Concurrency Comput. 26 (2014) 2053–2078.
[19] A. Ganapathi, H. Kuno, U. Dayal, J.L. Wiener, A. Fox, M. Jordan, D. Patterson, Predicting multiple metrics for queries: Better decisions enabled by machine learning, in: International Conference on Data Engineering (ICDE), 2009, pp. 592–603.
[20] M. Akdere, U. Cetintemel, M. Riondato, E. Upfal, S.B. Zdonik, Learning-based query performance modeling and prediction, in: IEEE 28th International Conference on Data Engineering (ICDE), 2011, pp. 390–401.
[21] D.A. Menasce D. Barbará, R. Dodge, Preserving QoS of E-commerce Sites through Self-Tuning: A Performance Model Approach, in: Proceedings of the 3rd ACM Conference on Electronic Commerce, Tampa, Florida, USA, 2001, pp. 224–234.
[22] D.A. Menasce, M.N. Bennani, On the use of performance models to design self-managing computer systems, in: Proceedings of Computer Measurement Group Conference, Dallas, TX. USA., 2003, pp. 1–9.
[23] A. Khattab, A. Algergawy, A. Sarhan, MAG: A performance evaluation framework for database systems, Knowl.-Based Syst. 85 (2015) 245–255.
[24] S.S. Lightstone, G. Lohman, D. Zilio, Toward autonomic computing with DB2 universal database, SIGMOD Rec. 31 (3) (2002) 55–61.
[25] A. Horzyk, E. Dudek-Dyduch, Effectiveness of artificial neural Networks adaptation according to time period of training data acquisition, ISDA (2005) 130–135.
[26] B. Niu, P. Martin, W. Powley, R. Horman, P. Bird, Workload adaptation in autonomic DBMSs, in: Proceedings of the Conference of the Center for Advanced Studies on Collaborative research (CASCON), ACM, NY, USA, 2006, pp. 161–173.
[27] S.F. Rodd, U.P. Kulkarni, Adaptive self-tuning techniques for performance tuning of database systems: a fuzzy-based approach with tuning moderation, Soft Comput. 19 (7) (2015) 2039–2045.
[28] B. Niu, P. Martin, W. Powley, Towards autonomic workload management in DBMSs, J. Database Manag. 20 (3) (2009) 1–17.
[29] S. Qian, S. Wang, Research on workload adaptation architecture for DBMS, in: International Symposium on Intelligence Information Processing and Trusted Computing, 2010, pp. 382–385.
[30] A. Khoshkbarforoushha, R. Ranjan, Resource and performance distribution prediction for large scale analytics queries, in: Proceedings of the 7th ACM/SPEC on International Conference on Performance Engineering, 2016, pp. 49–54.
[31] X.X. Wu, W. Banzhaf, The use of computational intelligence in intrusion detection systems: A review, Appl. Soft Comput. 10 (1) (2010) 1–35.
[32] N. Cristianini, J. Shawe-Taylor, An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods, Cambridge university press, 2000.
[33] C.W. Hsu, C.C. Chang, C.J. LinA, A practical guide to support vector classification, (2003) 1–16.
[34] C. Chang, C. Lin, LIBSVM: a library for support vector machine, ACM Trans. Intell. Syst. Technol. 2 (3) (2011) 27.
[35] C.K. Reisbeck, R.C. Schank, Inside Case-Based Reasoning, First Ed., Lawrence Erlbaum Associates, Hillsdale, NJ, US, 1989.
[36] A. Aamodt, E. Plaza, Case-based reasoning: Foundational issues, methodological variations, and system approaches, AI Commun. 7 (1) (1994) 39–59.
[37] MySQL Reference Manual 5.6, http://www.mysql.com/, 2017 (accessed on 24 March 2017).
[38] TPC Benchmark Standard Specification, http://www.tpc.org/, 2017 (accessed 10 March 2018).
[39] Database test 2 (Dbt2), Database Test Suite, http://osdldbt.sourceforge.net/, 2017 (accessed on 24 March 2017).
[40] B. Mozafari, C. Curino, A. Jindal, S. Madden, Performance and resource modeling in highly concurrent OLTP workloads, in: Proceedings of the ACM SIGMOD International Conference on Management of Data, 2013, pp. 301–312.
[41] C. Gupta, A. Mehta, PQR: Predicting query execution times for autonomous workload management, in: the Proceedings of the International Conference on Autonomic Computing, 2008, pp. 13–22.
[42] S. Agrawal, S. Chaudhuri, L. Kollar, A. Marathe, Database tuning advisor for Microsoft SQL server 2005, in: Proceedings of the 30th International Conference on Very Large Data Bases (VLDB), Canada, 2004, pp. 1110–1121.
[43] I.H. Witten, E. Frank, Data Mining: Practical Machine Learning Tools and Techniques, Second Ed., Morgan Kaufmann, San Francisco, 2005.
[44] R.D. Was, D.L. Neal, Weka Machine Learning Project: Cow Culling, The University of Waikato, Computer Science Department, Hamilton, New Zealand, 1994 Technical report.
[45] K. Duan, S.S. Keerthi, Which is the best multiclass SVM method? An empirical study, in: Proceedings of Multiple Classifier Systems, 2005, pp. 278–285.
[46] J. Demsar, Statistical comparisons of classifiers over multiple data sets, J. Mach. Learn. Res. 7 (2006) 1–30.
[47] S. Garcia, F. Herrera, An extension of statistical comparisons of classifiers over multiple data sets, J. Mach. Learn. Res. 9 (2008) 2677–2694.
[48] S. Garcia, A. Fernandez, J. Luengo, F. Herrera, Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence: Experimental Analysis of Power, Inf. Sci. 180 (2010) 2044–2064.
[49] KEEL Software. Available at http://www.keel.es., 2017 (accessed 13 Sep 2017).
[50] M. Friedman, The use of ranks to avoid the assumption of normality implicit in the analysis of variance, J. Am. Stat. Assoc. 32 (1937) 675–701.
[51] M. Friedman, A comparison of alternative tests of significance for the problem of m rankings, Ann. Math. Stat. 11 (1940) 86–92.
[52] H. Arboleda, A. Paz, M. Jiménez, G. Tamura, Development and instrumentation of a framework for the generation and management of self-adaptive enterprise applications, Ingeniería y Universidad 20 (2) (2016) 303–333.
[53] S. Bandyopadhyay, S. Saha, Unsupervised Classification: Similarity Measures, Classical and Metaheuristic Approaches and Applications, Springer Science & Business Media, 2012.
[54] J. Han, M. Kamber, Data Mining: Concepts and Techniques, Morgan Kaufmann Publishers, 2002.
[55] TPC BenchmarkH, 2018. (Decision Support) Standard Specificaiton (accessed 10 March 2018) http://www.tpc.org/tpc_documents_current_versions/pdf/tpc-h_v2.17.3.pdf/ .
[56] TPC BenchmarkC, Standard Specificaiton, 2018 (accessed 10 March 2018). http://www.tpc.org/tpc_documents_current_versions/pdf/tpc-c_v5.11.0.pdf/ .