Using Data Mining to Predict and Generate Optimum Multiple Execution Paths Compositions

Osama K. Qtaish[(1)], Zulikha Jamaludin[(2)], and Massudi Mahmuddin[(3)]

(1) School of Computing, College of Arts and Sciences, Universiti Utara Malaysia, 06010, UUM Sintok, Malaysia
E-mail: oqtaish@yahoo.com

(2) School of Computing, College of Arts and Sciences, Universiti Utara Malaysia, 06010, UUM Sintok, Malaysia
E-mail: zulie@uum.edu.my

(3) School of Computing, College of Arts and Sciences, Universiti Utara Malaysia, 06010, UUM Sintok, Malaysia
E-mail: ady@uum.edu.my

ABSTRACT

In multiple execution paths compositions, can we generate solutions that simultaneously optimize all the execution paths, while meeting global QoS constraints imposed by the clients? This paper proposes a runtime path prediction method based on data mining techniqes. The method predicts, at runtime, the execution path that will be followed during the composition's execution based on the information provided by composition requesters, making it possible to compute the optimization by considering only the predicted path. By using our method, it is expected to generate solutions that deliver the best possible QoS ratio, at the same time, minimize the violation of the global constraints. The proposed method is evaluated in terms of its prediction accuracy and scalability.

Keywords: Business process, classifier, data mining, QoS, runtime path prediction, service-oriented architecture, web service composition.

1- INTRODUCTION

One of the main benefits gained from implementing web services and SOA is the ability to compose new functionality out of existing outsourced web services into web service composition [1].

Recently, service composition technology can be used to develop business processes. A business process is comprise of a set of related tasks or activities that been designed to fulfill a specific goal. Each task (also known as abstract web service) can be accomplished by a single outsourced web service hosted by external partners. For example, in the design time, a software engineer of online bank loan application defines the business process by identifying and arranging the abstract services. Based on the semantic descriptions of the abstract services, many functionality equivalents

web services (similar services functionality called candidates) can be discovered for each abstract web service. Then, the service selection can be performed dynamically at runtime by selecting the best outsourced services that can accomplish the abstract services' functionality [2].

One of the most substantial selection factors that can be served as selection criteria between those equivalent services is the QoS criteria including web service's non-functional characteristics such as cost, response time, availability, and reliability [3].

Having QoS characteristics as selection criteria; the process of optimizing business processes aims to select one candidate web service for replacing each abstract web service such that the entire QoS of the business process (hereafter used interchangeably with terms "composite service" and "composition") is optimized while clients (i.e., organizations) QoS requirements are satisfied. These requirements include QoS global constraints and preferences.

On the other hand, compositions operate in highly dynamic environments. In such environments, different possible scenarios may occur at runtime, making the real time compositions facing unanticipated changes. Therefore, it is desirable to support the expectations that can be anticipated by composition engineers. The result is a distinct set of multiple composition paths. Each path represents a scenario that can be followed during the execution of a composition instance. Figure 1 illustrates three different possible execution paths of the services S = {S1, S2,…,S7}.

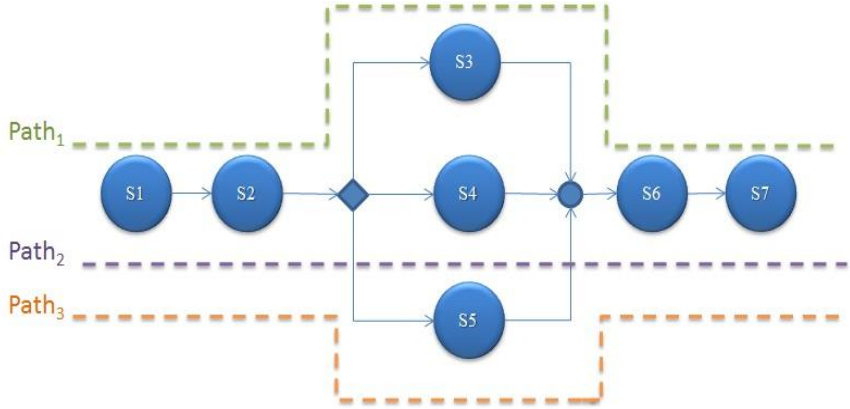


Figure 1 An example of multiple execution paths composition.

When optimizing multiple paths compositions, it is difficult generate a solution that simultaneously optimizes all execution paths involved in the composition at the same time, and satisfy global QoS constraints imposed by clients.

To solve this major issue, this work propses a runtime path prediction method that efficiently suits for optimizing multiple paths compositions. The method based on data mining techniqes and aims to predict, at runtime and just before the actual executions of the compositions, the path that will potentially be executed during the realization of a composition based on the information provided by composition requesters. The information predicted (i.e., the predicted path) using this method will be used by optimization algorithms for optimizing only the predicted path. By using the proposed method, it is expected to generate solutions that deliver the best possible QoS ratio, at the same time, minimize the violations of the global constraints.

2- RELATED WORKS

Two optimization techniqes are proposed to solve the multiple paths compositions problem, namely: the technique of optimizing all paths together, and the technique of optimizing each path separately.

Yu et al. [4], Canfora et al. [5], Wang et al. [6], Gabrel et al. [17] and Lécué [7] are example of approaches using the first technique. In this technique, the optimization is computed assuming that a certain path will be more likely executed than another according to probability of paths execution. The assumptions are based on stochastic information indicating the probability of paths being executed at runtime. Estimation of the paths' probability of executions is estimated either by inspecting the system logs or specified by the composition engineers. Uker and Carpenter [8], [9] present an approach that enables users to bias the optimizations using a set of meta-metrics including execution probability of an activity, previous execution history of each activity, and probability of occurrence. The approach aims to find an approximation solution for each path involved in the composition. A trade-off between the paths is made, which chooses a path to favor by using a set of meta-metrics. For each path, the meta-metrics are computed as the weighted average of the aggregate values of meta-metrics. Then, the selection problem is solved using integer programming solution. However, considering all paths together in computing the optimization may results in suboptimal solution for some execution paths. Even wose, if the composition execution follows the path with the less probability; the QoS requirements imposed by clients may violate.

The technique of optimizing each path separately is proposed by Zeng, et al. [10], [11]. In this technique, the optimization is computed for each path separately by decomposing the composition into execution paths. If there is a conflict in service selection in some abstract services that are common to multiple execution paths; the system identifies the hot path for the considered web service. However, the actual execution of the composition may not follow the hot path. In this case, suboptimal solutions may resluts for the executed path.

In contrast to the above mentioned techniqes, the proposed path prediction method allows for optimizing only the path that will be followed during the execution. By using this strategy of optimization, it is expected that the resulted solutions deliver the best possible QoS ratio and, at the same time, meet the QoS requirements.

3- METHOD FOR RUNTIME PATH PREDICTION

3-1 Composite service scenario

This section describes the scenarios that will be used throughout this paper to explain the method of predicting the execution path. In this paper, two different scenarios are used to show how generable is the proposed method.

3-1.1 Bank loan business process

The loan business process (composite services) supports by banks can be developed using web service composition technology. Each function of the composite service can be accomplished by a single outsourced web service. The loan composite service illustrated in figure 2 is composed of 22 web services which are connected using sequential (services are executed in sequence order) and conditional structures (among all the branches, only one branch is executed).

In this composite service, a client (i.e., a bank loan's requester) is required to fill online from for requesting a loan. The data provided by the client are forwarded to Check Loan Type web service to determine the loan types. Based on its type, the request is then forwarded to one of the five services: Check Home Loan, Check Educational Loan, Check New Car Loan, Check Personal Loan, or Check Used Car Loan. The request can be: accepted or rejected or approved conditionally in the case of a home loan. Approve (Reject) Home Loan, Approve (Reject) Educational Loan, Approve (Reject) New Car Loan, Approve (Reject) Personal Loan, and Approve (Reject) Used Car Loan are the web services in charge of accepting (rejecting) a loan request. The result of the loan request is then e-mailed to the client. Finally, the loan data is stored in a database by the Archive Application web service.

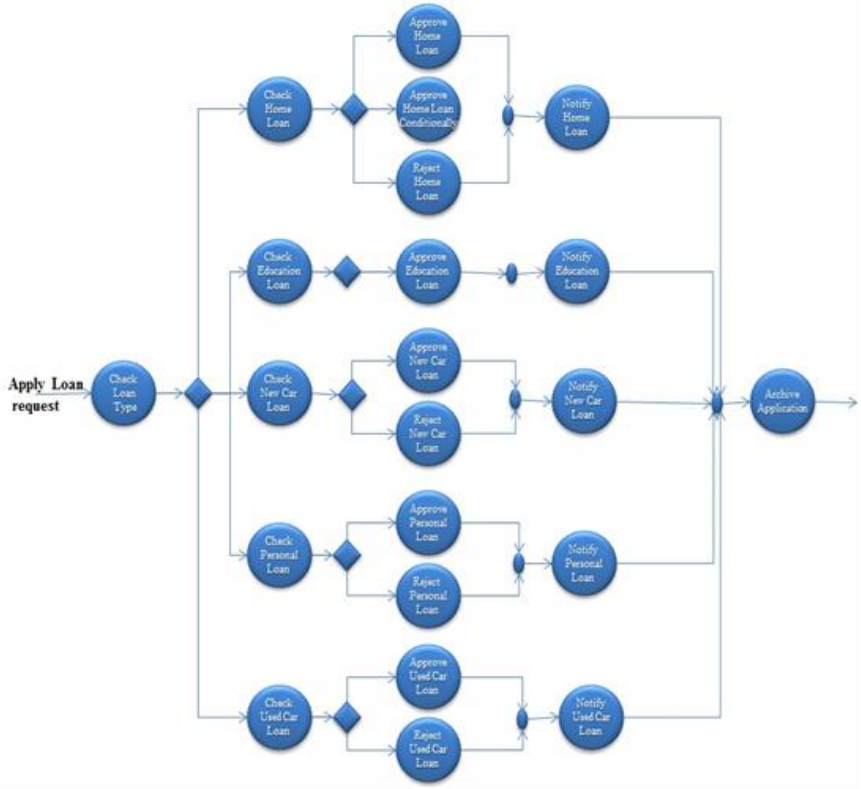


Figure 2 A typical bank loan composite service scenario.

3-1.2 An auto insurance business process

Auto insurance is one of several insurance types sold by insurance companies. A typical auto insurance composite service which represents a multiple paths composite service is illustrated in figure 3. As it is seen in figure 3, the service is composed of 11 web services and represents a multiple paths composite service scenario. It includes 4 different execution paths. The service sells two policies of auto insurance, namely comprehensive and a third party. Comprehensive is the most complete protections for vehicles; it covers the client's vehicle, other vehicles, etc. However, third party insurance covers only the damages that clients may cause for other vehicles. The service requesters are required to fill and apply application forms requesting for auto insurance. Then the information provided by clients is forwarded to Check Policy to determine the requested insurance policy. Based on the policy, the request is forwarded either to Evaluate Comprehensive or to Evaluate Third Party services. The request can be either approved or rejected. Approve Comprehensive and Approve Third Party are the services responsible for approving comprehensive/third party insurances. In contrast, Reject Comprehensive and Reject Third Party are the web services responsible for

rejecting comprehensive/third party insurances. The result of the auto insurance request is then e-mailed to the client. Finally, auto insurance application data is stored in a database by the Archive Application web service.

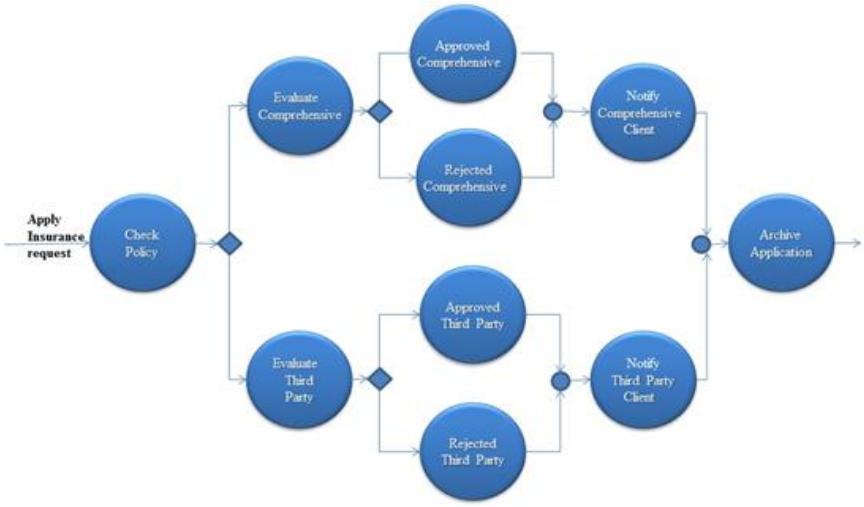


Figure 3 A typical auto insurance composite service scenario.

3-2 Composition logs

In Workflow Management Systems (WFMS), the data generated from the execution of business processes are recorded into so-called execution logs. During the execution of a composite service, WFMS stores data including real time information describing the execution and the behaviour of the composite service, web services, and instances. The data stored in these logs are rich with concealed information. One important piece of knowledge that can be extracted from these logs is the subset of the web services that will potentially be execute by composition instance.

3-3 Runtime path prediction method

Data mining techniqes can be employed in order to determine the execution path that will potentially be executed at runtime. Based on data mining, Cardoso [12] proposed a method for predicting the QoS of workflows before they executed or during the execution. Their method is extened and refined here for the purpose of predicting, at runtime, the execution path that will be followed during the composition's execution based on the information provided

by the service requesters. The following limitations are identified to be addressed in this work.

- The mentioned work performs the prediction at design time on information indicating the input (output) values parameters passed (received) to (from) activates. The prediction in this work is performed at runtime based on the information provided by the composite service requester when filling online application.

- In the mentioned work, it is not necessary that all attribute values are stored in logs i.e., there may be some missing information. This is because some activities may not have been invoked by the workflow management system when path mining is started. Using datasets with missing values to train classifiers affects the prediction quality of classifiers. In contrast to the mentioned work, in the proposed prediction method, there are no missing attribute values in the datasets because the stored values parameters are kinds of must entered attributes. These attributes represent personal information and information describing the condition of the service being requested. For example, a policy-type and an auto-model are examples of information that must be provided when requesting auto insurance. Having datasets with no missing values allows improving the prediction quality of the classifiers.

- In the mentioned work, profiles for each process instance are needed to be constructed for training the algorithms. In this work, the training dataset is created in the form of a relational table.

- In the mentioned work, experiments were conducted using one dataset that represent one process scenario. In this work, 10 datasets were used which represent two processes scenarios. The datasets represent different business process domains i.e., auto insurance and bank loan processes.

In the following discussions, the proposed runtime path prediction method is discussed in detail. The method consists of four phases:

The Log Preparation Phase

This phase is adopted from [12]. It includes extending the logs to store information indicating the input (output) values parameters passed (received) to (from) web services and their types such as a loan year, an income, etc. These values are generated at runtime during the execution of composition instances. Each ‘parameter/value’ entry as a data type, a name, and a value, (for example, int production-year=3). In addition, the class path is an extra field needs to be added to the log to store path information.It indicates the path that has been taken by a particular composition instance when the pa-

rameters have been assigned to a specific value set. The class path is associated in order to analyze the choices that have been made (i.e., the paths that have been executed) in the past execution of a composition, and to determine whether the paths that have been taken might be influenced by the information provided by compositions instances.

Preparation of Training Dataset Phase

This phase aims at using the runtime data about instance contained in logs as a training dataset for machine learning algorithms. The training dataset is typically in the form of a relational table in which each row represents one composition instance extracted from logs. Each instance in the training dataset is characterized by the values parameters of a composition requester. In addition, it is labeled with a class indicating the path that has been taken when the parameters have been assigned to a specific value set. In this way, a set of classified data is taken by a learning schema to learn a way of classifying unseen instances. For example, table 1 shows the structures of training dataset for the auto insurance composite service scenario presented in the previous section. As it is seen in table 1, each instance consists of four parameters, namely a policy-type, a manufacture-type, an auto-model, and a production-year. These are associated with a class, namely path indicating the path that has been executed when these parameters have been assigned to a specific value set. Table 2 shows an example of training dataset for the auto insurance composite service. As it was mentioned earlier, there are no missing attribute values in the datasets because the stored values parameters are kinds of must entered attributes.

Table 1 Training dataset structure for auto insurance problem

Dataset structure				Class
Policy-type	Manufacture-type	Auto-model	Production-year	Path

Table 2 Example of auto insurance training dataset

Policy-type	Manufacture-type	Auto-model	Production-year	Path
Comprehensive	Kia	Rio	2004	Path1
Third Party	Volkswagen	Golf	2000	Path3
Comprehensive	Fiat	Punto	1996	Path2

The Learning Phase

This phase aims at building classifiers. Path prediction is treated as a classification problem. Once storing enough information in logs, machine learning algorithms can be used to establish a relationship between the values parameters and the paths taken at runtime.

It is recommended for a learning process to be iteratively refined when the process execution proceeds and more information about composite service execution becomes available. More data yields to build more accurate prediction classifiers.

The output of this phase is classifiers. A classifier is a function used to map unlabeled instance to a labeled by producing a set of classification rules. For example, if the requested policy-type is comprehensive, the manufacture-model is Fiat, and the production-year is less than 2004 then path2 i.e., rejected comprehensive insurance.

The Runtime Path Prediction Phase

This phase aims at performing runtime path prediction based on the information provided by a composition requester. The classifier is now ready for classifying unknown classes, i.e., predict the path that is followed during the execution.

At runtime, a client (i.e., a service requester) for auto insurance is required to fill an application form and apply it to request insurance. The form represents personal data and the data describes the condition of the service being requested. For example, a policy-type, a manufacture-type, an auto-model, and a production-year are examples of such data. Figure 4 illustrates an example of a typical application form for auto insurance request.

The data needed for prediction i.e., a policy-type, a manufacture-type, an auto-model, and a production-year are then collected and fed to a classifier to be classified into target classes, i.e., execution paths.

The output of this phase is the prediction of a certain execution path representing the path that is potentially followed during the execution of the bank loan composite service. This important information i.e., predicted path is utilized by the optimization algorithms in order to optimize the predicted path. The runtime path prediction method is illustrated in figure 5.

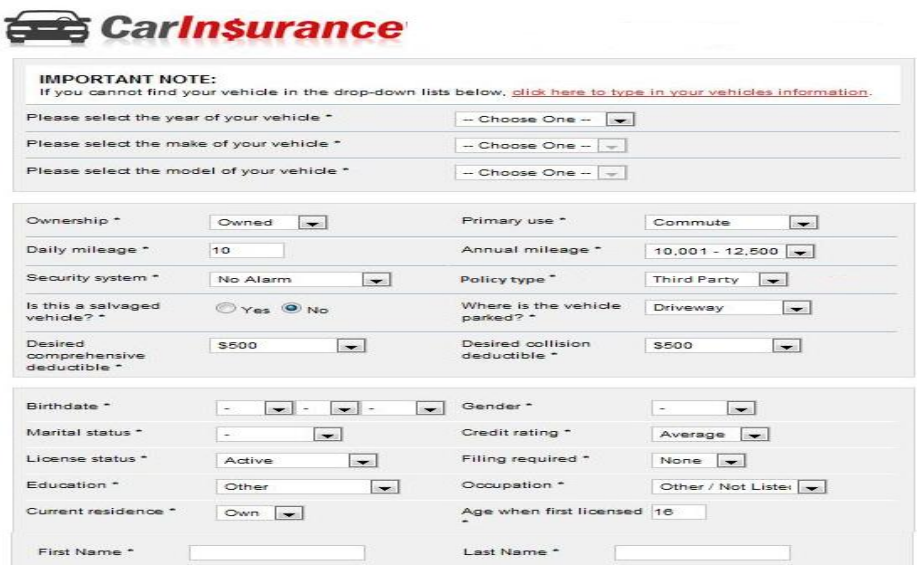


Figure 4 A typical online application form for requesting auto insurance.

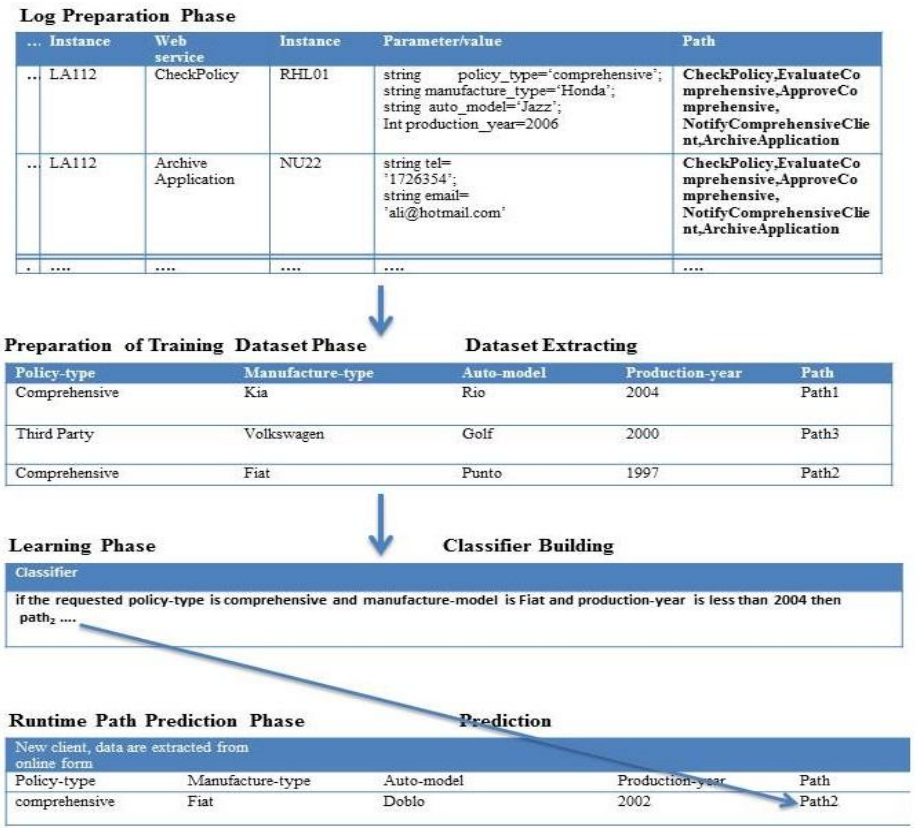


Figure 5 The runtime path prediction method

4- EXPERIMENTS

The purpose behind this experiment is: (1) to validate the accuracy of path prediction when several machine learning algorithms are applied to several different datasets, (2) to study how will the prediction method scale with a rising number of execution paths' involvement.

4-1 Data set description

The first dataset represented an auto insurance problem and consists of 826 instances (i.e., all runtime data related to the auto insurance requesters). The data was collected from a major insurance company. The dataset characterized by four attributes, namely a policy-type, a manufacture-type, an auto-model, and a production-year. The attributes policy-type, manufacture-type, and auto-model are nominal while the production-year is numeric. For example, a policy-type attribute can take comprehensive and third party values.

Beside the auto insurance dataset, several datasets were required for evaluating the scalability of the path prediction method. Each dataset should include different numbers of paths involvement. For this purpose, a bank loan problem was used to create 9 datasets representing variable numbers of paths ranging from 2 up to 10 paths. Paths are identified based on the bank loan composite service illustrated in figure 2 which included 10 paths. To effectively compare between the learning algorithms when evaluating the scalability, each dataset has an equal number of service instance i.e., 1000 instances; therefore, an equal size of subsets (i.e., 100 instances) can be obtained in each iteration of the 10-fold cross-validation method. Table 3 lists the 9 datasets used for evaluating the scalability of the path prediction method. The loan datasets are characterized by four attributes, namely income, loan-amount, a loan-type, and a loan year. The attribute income, loan-amount, and loan-years are numeric whereas the attribute loan-type is nominal. The attribute loan-type can take the finite set of values: a home loan, an education loan, a new car loan, a personal loan and a used car loan. These types are the most common loan types.

For all created datasets i.e., 10 datasets, the most informative attributes were selected for each dataset which was determined after conducting preliminary tests. In addition, a class, namely path was added as an extra field for each instance in the datasets for the purpose of path prediction. It indicates that the path has been followed by each instance. The class path of auto insurance dataset can take a finite set of values: path1, path2, path3, and path4. These four paths are contained in the auto insurance composite service as it is seen from figure 3. A detailed description of each path is presented in table 4. On the other hand, the class path in loan datasets can take a finite set of values: *Path1, Path2 … Path10* as it is seen in figure 2. A detailed description of each path is presented in table 5.

The class path is labeled based on the instance data and the decision that has been made when evaluating the instance (i.e., either approve or reject a process request). For example, assume that third party insurance has been requested by auto insurance's requester, and the request is rejected. Then as it is seen for table 4, the class path is labeled as path4. Figure 6 shows a few recoreds from Dataset5.

Table 3 Datasets description for bank loan composite service.

Dataset	No. of Paths	No. of instances	Loan type	Included classes(paths)
Dataset1	2 paths	1000	New Car	$Path_5$,$Path_6$
Dataset2	3 paths	1000	New Car Education	$Path_4$,$Path_5$,$Path_6$
Dataset3	4 paths	1000	New Car Personal	$Path_5$,$path_6$,$Path_6$,$Path_8$
Dataset4	5 paths	1000	New Car Home	$Path_1$,$Path_2$,$Path_3$, $Path_5$,$Path_6$
Dataset5	6 paths	1000	New Car Home Education	$Path_1$,$Path_2$,$Path_3$,$Path_4$, $Path_5$,$Path_6$
Dataset6	6 paths	1000	New Car Home Personal	$Path_1$,$Path_2$,$Path_3$,$Path_5$,$Path_6$, $Path_6$,$Path_8$
Dataset6	8 paths	1000	New Car Home Education Personal	$Path_1$,$Path_2$,$Path_3$,$Path_4$,$Path_5$, $Path_6$,$Path_6$,$Path_8$
Dataset8	9 paths	1000	New Car Home Personal Used Car	$Path_1$,$Path_2$,$Path_3$,$Path_5$,$Path_6$, $Path_6$,$Path_8$,$Path_9$,$Path_{10}$
Dataset9	10 paths	1000	New Car Home Education Personal Used Car	$Path_1$,$Path_2$,$Path_3$,$Path_4$,$Path_5$, $Path_6$,$Path_6$,$Path_8$,$Path_9$,$Path_{10}$

Table 4 Path description for auto insurance composite service.

Path	Policy Type /Decision	Path description
$Path_1$	Comprehensive Approved	CheckPolicy,EvaluateComprehensive, RejectComprehensive,NotifyComprehensiveClient, ArchiveApplication
$Path_2$	Comprehensive Rejected	CheckPolicy,EvaluateComprehensive ,ApproveComprehensive,NotifyComprehensiveClient ,ArchiveApplication
$Path_3$	Third Party Approved	CheckPolicy,EvaluateThirdParty, ApproveThirdParty, NotifyThirdPartyClient,ArchiveApplication
$Path_4$	Third Party Rejected	CheckPolicy,EvaluateThirdParty, RejectThirdParty,NotifyThirdPartyClient, ArchiveApplication

Table 5 Path Description for bank loan composite service.

Path	Loan Type /Decision	Description
$Path_1$	Home Approved	CheckLoanType,CheckHomeLoan,ApproveHomeLoa n, NotifyHomeLoanClient,ArchiveApplication
$Path_2$	Home Approved Conditionally	CheckLoanType,CheckHomeLoan,RejectHomeLoan, NotifyHomeLoanClient,ArchiveApplication
$Path_3$	Home Rejected	CheckLoanType,CheckHomeLoan,ApproveHomeLoa nConditionaly, NotifyHomeLoanClient,ArchiveApplication
$Path_4$	Education Approved	CheckLoanType,CheckEducationLoan,ApproveEduca tionLoan, NotifyEducationLoanClient,ArchiveApplication
$Path_5$	New Car Approved	CheckLoanType,CheckNewCarLoan,ApproveNewCar Loan, NotifyNewCarLoanClient,ArchiveApplication
$Path_6$	New Car Rejected	CheckLoanType,CheckNewCarLoan,RejectNewCarL oan, NotifyNewCarLoanClient,ArchiveApplication
$Path_6$	Personal Approved	CheckLoanType,CheckPersonalLoan,ApprovePerson alLoan, NotifyPersonalLoanClient,ArchiveApplication
$Path_8$	Personal Rejected	CheckLoanType,CheckPersonalLoan,RejectPersonal Loan, NotifyPersonalLoanClient,ArchiveApplication
$Path_9$	Used Car Approved	CheckLoanType,CheckUsedCarLoan,ApproveUsedC arLoan, NotifyUsedCarLoanClient,ArchiveApplication

$Path_{10}$	Used Car Rejected	CheckLoanType,CheckUsedCarLoan,RejectUsedCar Loan, NotifyUsedCarLoanClient,ArchiveApplication

```
@relation 'dataset5'
@attribute loan-amount numeric
@attribute loan-year numeric
@attribute income numeric
@attribute loan-type {Home,'New Car',Education}
@attribute path {Path1,Path5,Path2,Path6,Path4,Path3}
@data
49574.7,7,3415,Home,Path1
15983.45,3,2948,'New Car',Path5
114488.55,16,2467,Home,Path2
50708,1,3709,'New Car',Path6
24769.85,1,2982,'New Car',Path6
21751.2,13,3197,Home,Path1
12114.2,6,3665,'New Car',Path5
33544,3,3043,'New Car',Path5
77080.35,8,2797,Home,Path1
```

Figure 6 A sample of Dataset5.

4- 2 Data mining algorithms

Different supervised learning methods can be used to carry out path. Among these algorithms, Naïve Base (NB), J48, and Sequential Minimal Optimization (SMO) methods are selected to be experimented. These algorithms are one of the most well-known algorithms in the data mining community.

J48 algorithm is Weka's (2004) implementation of the C4.5 decision tree learner [14]. Since finding an optimal solution tree is a multi-objective problem, it uses a heuristic approach to generate suboptimal decision trees. J48 is chosen because it is a good representative of a symbolic method.

Naïve Bayes (NB) classifier technique is based on the so-called Bayesian theorem. It is work by analysing the relationship between the dependent variable and the independent variable, and for each relationship, a conditional probability is derived. NB is chosen because it is a good representative of a probabilistic method.

SMO [15] is an improved training algorithm for SVM (Support Vector Machines) [16]. Usually, a very large quadratic programming (QP) problem of the solution is required to train SVM. SMO breaks down a large QP problem into a series of smaller QP problems. SMO improves its scaling and computation time significantly because the utilization of the smallest possible QP problems that are solved quickly and analytically. SMO is chosen because its success in text mining domain.

4-3 Simulation tool

The experiments are conducted utilizing WEKA, which is popular open source software developed at the University of Waikato in New Zealand. In academia, WEKA is one of the most well-known data mining systems.

WEKA provides implementations for J48, NB, and SMO as well as a wide variety of learning algorithms. Thus, there is no need to manually write the algorithms 'code. In a simple way, the algorithms are easily applied to the generated datasets.

4-4 The performance evaluation

The runtime path prediction method is considered successful if the prediction accuracy is high. In this work, the prediction accuracy is the primary measure for evaluating the prediction method. Beside the accuracy measure, the precision and recall criteria and the number of correctly/incorrectly classified instances are also considered.

The commonly used technique ''10-fold cross validation" is used for assessing the classifiers. This technique is suitable in prediction to estimate the future prediction accuracy of a classifier. .

In this method, the dataset is split into 10 mutually executive subsets of approximately equal size. A machine learning algorithm is trained and tested 10 times; at each time it is tested on 1 of the 10 subsets and trained using the 9 remaining subsets (i.e., each subsets being once the test set and reaming subsets being the training set). The iteration is necessary to ensure that all instances in the dataset are part of the test and train subsets. The 10 results are then averaged to give the overall result.

4-5 Accuracy of path prediction

It is crucial to have high prediction accuracy when predicting the execution paths because the optimization process depends on the predicted path. Any false prediction means that the optimization results in solutions that may have low QoS ratio or may violate the global constraints. Therefore, the first experiment aimed at validating the accuracy of the path prediction. The experiments

were conducted using three selected learning algorithms, namely J48, NB, and SMO. These algorithms are applied to the auto insurance dataset which contains 826 instances.

The 10 prediction accuracy results are presented in figure 7. Table 6 depicts the average results obtained for the various measures.

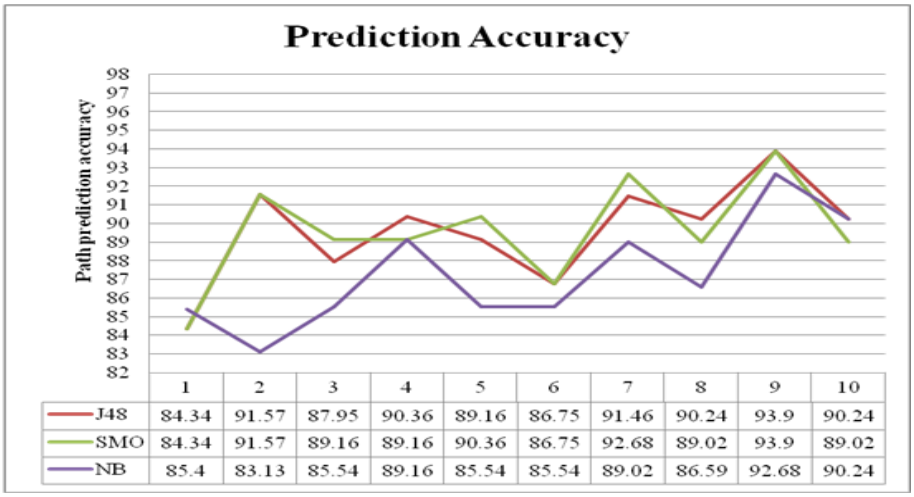


Figure 7 The prediction accuracy per each fold achieved using J48, NB, and SMO classifiers when applied to auto insurance dataset.

Table 6 Evaluation criteria results achieved by using the J48, NB, and SMO classifiers when applied to auto insurance dataset.

Evaluation Criteria	Classifiers		
	J48	SMO	NB
Prediction Accuracy	89.60	89.60	86.28
Precision	0.90	0.90	0.88
Recall	0.94	0.95	0.95

The results presented in table 6 indicate that all the selected classifiers achieved promising accuracy when predicting the execution paths. This is expected because learning algorithms in the proposed method are trained on the most informative attributes of instances executions. This allows classifiers for better learning and consequently improves the prediction quality. Further-

more, in the proposed method, the attributes used for learning are a kind of must entered attributes that are provided by service requesters i.e., no missing attributes values. Building classifiers using datasets with no missing values allows improving the prediction quality of these classifiers.

As it is presented in table 6, both J48 and SMO classifiers achieve the highest accuracy prediction, i.e. 89.60. The lowest accuracy is achieved by using NB classifiers i.e., 86.28. It is observed from figure 7 that both J48 and SMO produce prediction models with the best accuracies in 9 out of 10 tests than NB. NB outperforms both J48 and SMO in the first test only.

Comparing the precision and recall results of all classifiers, the high precision results of J48 and SMO indicate that both have less proportion of negatives cases that were incorrectly classified as positive. For recall results, the results indicate that SMO and NB have less proportion of negatives cases which were classified correctly.

In the form of a bar diagram, figure 8 illustrates the number of correctly/incorrectly classified instances using all classifiers. It is seen that the total number of instances i.e., 826 is equal in the three cases since the same dataset is used in the experiment. As it is seen in figure 8, both J48 and SMO classifiers are able to correctly classify 740 instances out of 826 instances. Only 86 instances are incorrectly classified by these classifiers. However, 721 instances are correctly classified by NB and 105 instances are incorrectly classified.

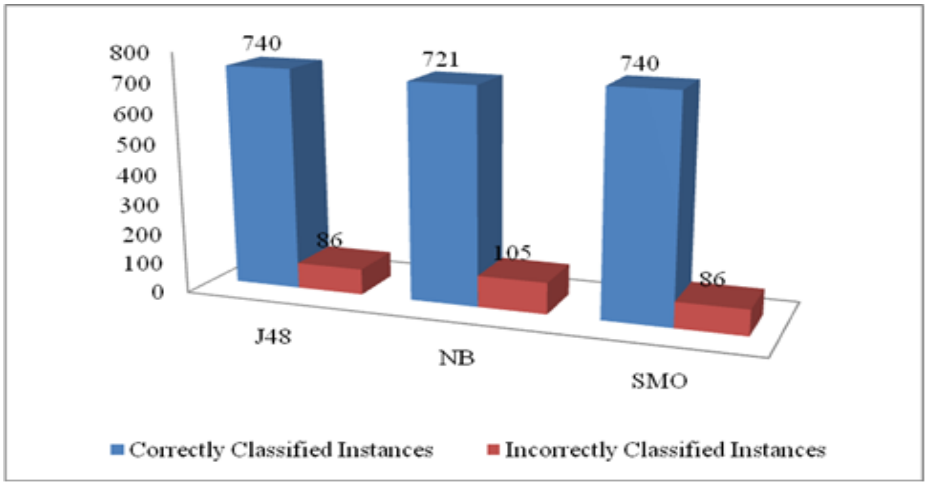


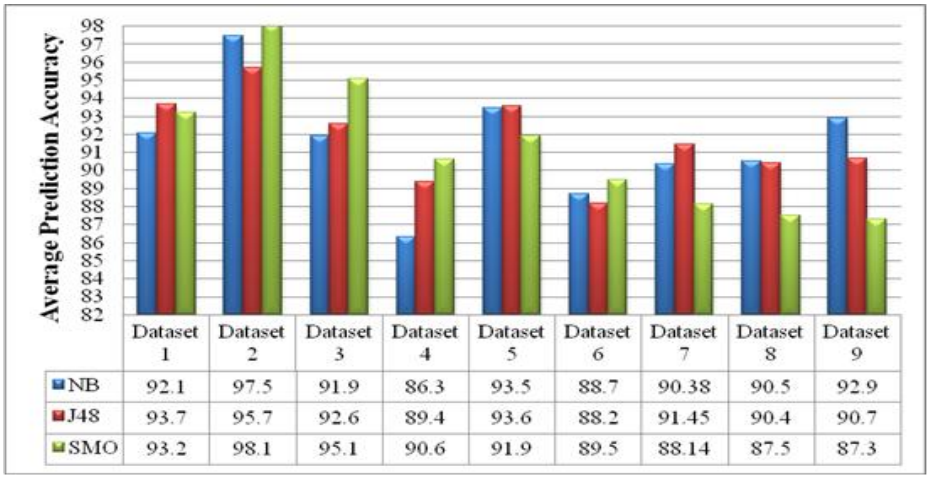
Figure 8 A number of correctly/incorrectly classified instances achieved by using J48, NB, and SMO classifiers when applied to auto insurance dataset.

Having such encouraging results of prediction accuracy contributes to the generation of high QoS ratio solutions and minimizes the constraints violated number of the generated solutions.

4-6 Scalability of the Prediction Method

The number of execution paths involved in compositions varies between one composite service and another one making us wonder about the prediction method's ability to accurately predict the paths when having a growing number of paths involvements. Therefore, the second experiment aimed at studying how the prediction method scaled with a rising number of execution paths involvement. For this purpose, 9 datasets representing the bank loan process were used for this experiment. Table 3 shows the datasets. For fair comparison, each dataset contained 1000 instances. The instances represented a bank loan process that involved execution paths ranging from 2 up to10 paths. The experiments were conducted using J48, NB, and SMO.

The average of the prediction accuracy for the three classifiers when applied to the 9 datasets is illustrated in figure 9.



	Dataset 1	Dataset 2	Dataset 3	Dataset 4	Dataset 5	Dataset 6	Dataset 7	Dataset 8	Dataset 9
NB	92.1	97.5	91.9	86.3	93.5	88.7	90.38	90.5	92.9
J48	93.7	95.7	92.6	89.4	93.6	88.2	91.45	90.4	90.7
SMO	93.2	98.1	95.1	90.6	91.9	89.5	88.14	87.5	87.3

Figure 9 An average prediction accuracy acheived by using NB, J48, and SMO classifiers when applied to 9 different datasets

As it is illustrated in figure 9, there are ups and downs in the accuracy results for all classifiers which make us wonder about the reasons behind the variance in the prediction accuracy results. It is seen that these ups and downs are not related to the number of paths involvement. Take Dataset1 and Dataset5 as examples. The Dataset1 involves 2 paths while Dataset5 involves 6 paths. The average prediction accuracy of all classifiers when applied to Dataset1 is 93 which is equal to the average prediction accuracies for all classifiers when applied to Dataset5 which is 93. Furthermore, take Dataset4, which involves 5 paths, and Dataset9 which involves 10 paths, the average prediction accuracy of all classifiers when applied to Datsset4 i.e., 88.8 is less than Dataset9 i.e., 90.3. Even that Dataset9 includes 10 paths; it has higher aver-

age prediction accuracy than Dataset4 which includes 5 paths. Based on these findings, it is valid to conclude that the rising number of classes' involvements does not affect the prediction accuracy of the classifiers. In other words, there is no relationship between the number of classes' involvements in the classification process and the prediction accuracy of the classifier.

However, what causes the ups and downs in the accuracy results is a question that is still needed to be answered? To answer this question, a comparison between the Dataset2, which has the maximum average accuracy i.e., 96.1, and the Dataset4, which has the minimum i.e., 88.8, is needed to be conducted. A test experiment was conducted and aimed at studying these datasets. SMO classifier was chosen to be applied on these datasets since it had the maximum prediction accuracy when applied to these datasets i.e., it had satiable results. Having got the experiment results, it was noticed that the precision and the recall results for some classes (i.e., paths) were either very low or very high in comparison with other classes in the same dataset. Table 7 presents the precision and the recall results for Dataset2 while table 8 presents the precision and the recall results for Dataset4. As it is seen in table 7, the precision and the recall results for path4 are very high i.e., 1. It indicates that there are no incorrectly classification for this class i.e., path4. For Dataset4, in comparison with the other classes in the same dataset, the precision and the recall results for path2 are low 0.666 and 0.686 respectively as it is presented in table 8.

Table 7 Precision and Recall results for Dataset2.

Class	Precision	Recall
Path4	1	1
Path5	0.938	0.916
Path6	0.886	0.916

Table 8 Precision and Recall results for Dataset4.

Class	Precision	Recall
Path1	0.925	0.936
Path2	0.666	0.686
Path3	0.946	0.918
Path5	0.92	0.905
Path6	0.851	0.862

As it is seen in table 3, Dataset2 represents two loan types, namely a new car, which involves 2 paths (i.e., path5 and path6), and education, which involves 1 path (i.e., path4). It is clear that the simple structure of education (i.e., only one class belongs to the education loan type) is the reason behind the very

high results of precision and recall for this class. These high results contribute in achieving high accuracy results compared with the complex home loan structure, where path2, which consists of 3 paths, belongs to the home loan type. Investigating other datasets, which have low average prediction accuracy such as Dataset6 88.8, and Dataset8 89.5, showed that the presence of home loan type (i.e., path2) is the reason behind the low average accuracy results. Based on these findings, it can be concluded that the accuracy highly depends on the structure of compositions.

Based on the results of the this test experiment, it is valid to say that the proposed approach is suitable for any compositions regardless of the number of execution paths involvements. However, the structure of the business process plays an important role in the prediction accuracy results.

5- CONCLUSION

The proposed path prediction method predicts the execution paths that will be followed during the executions of compositions based on the information provided by service requesters. The method allows for optimizing only the path that will be followed during the execution, yielding for solutions delivering the best possible QoS, at the same time, satisfying global constraints.
The results obtained from evaluating the prediction accuracy indicate that the J48, SMO, and NB classifiers achieved promising accuracy prediction i.e., 89.60, 89.60, and 86.28 respectively. These excellent results yield for the generation of high QoS ratio solutions and minimizing the constraints' violation of the generated solutions.

The results also indicate that the rising number of classes' involvements doesn't affect the prediction accuracy of the classthere, making the proposed method suitable for any compositions regardless of the number of execution paths' involvements. However, the structure of the business process plays an important role in the prediction accuracy results.

REFERENCES

[1] F. Rosenberg, P. Leitner, A. Michlmayr, P. Celikovic, and S. Dustdar, "Towards Composition as a Service - A Quality of Service Driven Approach," Proc. 2009 IEEE International Conference on Data Engineering, IEEE Computer Society, pp. 1733-1740, 2009.

[2] D. Ardagna, and B. Pernici, "Global and Local QoS Guarantee in Web Service Selection," Business Process Management Workshops, C. Bussler and A. Haller, eds., Heidelberg: Springer Berlin, pp. 32-46, 2006.

[3] O.K. Qtaish, and Z.B. Jamaludin, "QoS criteria for distinguishing the competing web services," Proc. 2011 International Conference on Data Engineering and Internet Technology (DEIT), Bali, IEEE Computer Society, 2011.

[4] T. Yu, Y. Zhang, and K.-J. Lin, “Efficient algorithms for Web services selection with end-to-end QoS constraints,” ACM Transactions on the Web (TWEB), Vol. 1(1), 2007.

[5] G. Canfora, M.D. Penta, R. Esposito, and M.L Villani, “An approach for QoS-aware service composition based on genetic algorithms,” Proc. 2005 conference on Genetic and evolutionary computation, Washington, ACM, 2005, 1069-1075. Washington, ACM, pp. 1069-1075, 2005.

[6] R. Wang, C.-H. Chi, and J. Deng, “A Fast Heuristic Algorithm for the Composite Web Service Selection,” Advances in Data and Web Management, Q. Li, L. Feng, J. Pei, S. Wang, X. Zhou and Q.-M. Zhu, eds., Heidelberg: Springer Berlin, PP. 506-518, 2009.

[7] F. Lécué, “Optimizing QoS-Aware Semantic Web Service Composition”, The Semantic Web - ISWC 2009, A. Bernstein, D. Karger, T. Heath, L. Feigenbaum, D. Maynard, E. Motta, and K. Thirunarayan, eds., Heidelberg: Springer Berlin, PP. 375-391, 2009.

[8] R. Ukor, and A. Carpenter, “On Modelled Flexibility and Service Selection Optimisation,” Proc. 9th Workshop on Business Process Modeling, Development and Support, Montpellier, 335, 2008.

[9] R. Ukor, and A. Carpenter, “Flexible Service Selection Optimization Using Meta-Metrics,” Proc. 2009 Congress on Services – I, IEEE Computer Society, pp. 593-598, 2009.

[10] L. Zeng, B. Benatallah, M. Dumas, J. Kalagnanam, and Q.Z Sheng, “Quality driven web services composition,” Proc.12th international conference on World Wide Web, Budapest, ACM, pp. 411-421, 2003,.

[11] L. Zeng, B. Benatallah, A.H.H Ngu, M. Duma, J. Kalagnanam, and H. Chang, “QoS-aware middleware for Web services composition,” IEEE Transactions on Software Engineering, Vol. 30(5), pp. 311-327, 2004.

[12] J. Cardoso, “Applying Data Mining Algorithms to Calculate the Quality of Service of Workflow Processes,” Intelligent Techniques and Tools for Novel System Architectures, P. Chountas, I. Petrounias and J. Kacprzyk, eds., Heidelberg: Springer Berlin, PP. 3-18, 2008.

[13] S. Sumathi, and S. Esakkirajan, fundamentals of relational database management systems, Heidelberg: Springer Berlin, 2007.

[14] J.R. Quinlan, C4.5: programs for machine learning, San Francisco: Morgan Kaufmann Publishers Inc., 1993.

[15] J.C. Platt, “Fast training of support vector machines using sequential minimal optimization,” Advances in kernel methods, B. Scholkopf, C. J. C. Burges and A. J. Smola, eds., MIT Press, PP. 185-208, 1999.

[16] C. Cortes, and V. Vapnik, “Support-vector networks,” Machine Learning, Vol. 20(3), PP. 273-297, 1995.

[17] V. Gabrel, M. Maude, and M. Cecile, "A new linear program for QoS-aware web service composition based on complex workflow", 2013.