Regression-Based Bootstrapping of Web Service Reputation Measurement

Okba Tibermacine, Chouki Tibermacine, Foudil Cherif

Regression-Based Bootstrapping of Web Service Reputation Measurement

Okba Tibermacine*, Chouki Tibermacine† and Foudil Cherif*
*Computer Science Department, Biskra University, Algeria
o.tibermacine@univ-biskra.dz, foud_cherif@yahoo.com
†LIRMM, CNRS and University of Montpellier, France
tibermacin@lirmm.fr

***Abstract*—In the literature, many solutions for measuring the reputation of web services have been proposed. These solutions help in building service recommendation systems. Nonetheless, there are still many challenges that need to be addressed in this context, such as the "cold start" problem, and the lack of estimation of the initial reputation values of newcomer web services. As reputation measurement depends on the previous reputation values, the lack of initial values can subvert the performance of the whole service recommendation system, making it vulnerable to different threats, like the Sybil attack. In this paper, we propose a new bootstrapping mechanism for evaluating the reputation of newcomer web services based on their initial Quality of Service (QoS) attributes, and their similarity with "long-standing" web services. Basically, the technique uses regression models for estimating the unknown reputation values of newcomer services from their known values of QoS attributes. The technique has been experimented on a large set of services, and its performance has been measured using some statistical metrics, such as the coefficient of determination (R^2) and the Mean Square Error (MSE).**

***Keywords*-Reputation Measurement, Quality of Service, Regression Model.**

I. Introduction

Web service recommendation systems provide a precious assistance to users in selecting the best available services for their systems. In order to recommend services, such systems manage different kinds of information about services, among which reputation. This subjective characteristic is an aggregation of feedback ratings from users. It reflects how a given service is perceived by its users.

In the literature, many reputation management models have been proposed to accurately evaluate the reputation of web services based on user feedback ratings [1]–[4]. Although these models have addressed many reputation aspects such as user credibility, time sensitivity, personalized preferences, majority ratings, etc., reputation bootstrapping is still a common neglected aspect [4]–[6].

In fact, reputation bootstrapping (*i.e.*, the mechanism of assigning rates to newly published (newcomer) web services that have no rating history in the system) is an important and challenging issue due to the following reasons:

- The mechanism has to assign fair reputation values to newcomer services to enhance their visibility and to give them chance to compete with similar services during service recommendation and selection phases. Hence, it provides a solution to the "cold start" problem, which describes the situation in which a recommendation system is unable to make a meaningful recommendation due an initial lack of ratings [7].
- The assigned reputation values have to reflect a particular service, and should not be general values about the recommendation system, such as attributing the average reputation in the system [8], or assigning a fixed reputation based on the evaluated rate of maliciousness in the system [9].
- The lack of a correct estimation of the initial reputation values of newcomer services may subvert the whole system, making it vulnerable to different threats [4] (e.g., the Sybil Attack [10].)
- The bootstrapping technique has to provide a solution to the *whitewashing* problem too [5]. Whitewashing (changing identity) occurs when an entity leaves the recommendation system then re-integrates it with a new identity in order to erase its poor reputation that was gathered with its previous identity [9].

Some approaches for reputation bootstrapping have been proposed in the past [8], [9], [11], [12]). Most of them assign the same initial reputation value for every newcomer Web service. Or, they just do not offer a complete solution that considers all the previous challenges.

In this paper, we introduce a new bootstrapping technique for assigning initial estimated reputation values to newcomer Web services. Though reputation is a subjective concept, it reflects users satisfaction about the service's offered quality of service (QoS). It has been observed that fair feedback ratings provided from the majority of honest users are correlated, even with a slight deviation (due to a difference in raters opinions), to the QoS of the consumed services [13]. Hence, we employed in this technique QoS and reputation data of long-standing services, in order to build a reputation estimation model for bootstrapping the reputation of newcomer web services. We mean by "long-standing services" the services that have long feedback records constructed from the collected user feedback ratings.

The technique is based on three main phases:

- **Provider reputation evaluation:** The system assesses the reputation of the new service provider from the reputation of its previously published web services.
- **Reputation estimation from similar web services:** First, the system selects among the long-standing web services those which are similar to the newcomer service. Second, the system builds an equation model based on: i) reputation scores and ii) the QoS of similar services. Finally, the system, based on the established model, estimates a reputation value of the newcomer service.
- **Regression-based reputation estimation:** Likewise to the second phase, the system builds a multiple linear regression model from QoS and reputation data of all long-standing web services. This model enables the estimation of the unknown reputation value of new comer services from their known values of QoS.

The final reputation is assigned to the newcomer web service depending on the results of the three phases.

The remaining of the paper is organized as follows: in Section II, we present the reputation bootstrapping model. Phases one, two and three are detailed respectively in Sections III, IV, and V. We show the results of our experiments in Section VI. We discuss the related work in Section VII, before concluding the paper in Section VIII.

II. Reputation bootstrapping model

Rather than assigning a default initial reputation value to newly posted web services, we propose a reputation bootstrapping model that assigns appropriate reputation values depending on their initial QoS. Thereby, the assigned reputation values give more chances for these services to be recommended and selected. Moreover, the bootstrapping model offers a solution for the "cold start" and the "whitewashing" problems. So, from the one hand we avoid fixed initial values, and from the other hand, even if a service leaves the system and comes back later with another identity (Name, URL, Provider Domain, etc.), it will get a reputation value that approximates to its reputation values before leaving the system.

We suppose that the bootstrapping model belongs to a reputation management framework for web service recommendation. This framework enables the collection of feedback ratings from service users, the assessment of reputation values of web services, and the monitoring (or the collection) of the QoS of web services. Moreover, the system indexes Web service descriptions (WSDL documents) and keeps information of all services, including dead or left services.

At the initial phase, when a new Web service S_i arrives to the system, we assume that it comes with a set of initial QoS vector $Q_{S_i}^{init} =< Q_{i,1}, Q_{i,2}, ..., Q_{i,k} >$. These QoS values are provided during publication time by the service provider ($Pr(S_i)$) as the advertised QoS. They can be established by the system after a short period of service testing and monitoring (The system can use one of the approaches proposed in this survey [14]). Then, to bootstrap the reputation of the new service S_i, the system goes through three phases: i) provider reputation evaluation, which is explained in Section III, ii) reputation estimation from similar services, which is described in Section IV, and iii) reputation estimation from multiple regression models built from QoS and reputation values of long-standing services in the system, which is detailed in Section V.

In Algorithm 1, we present the process that covers the three phases to estimate the reputation of the new web services S_i. First, the system checks whether the provider of the service is known by the system (Line 1 in Algorithm 1), that is, the service provider belongs to the list of providers *ProviderList* that have previously published services in the system. In the positive case, the system computes the reputation of this provider, denoted *prReputation*, based on the reputation of its long-standing services (Line 2). We present details on how we calculate the reputation of the provider in Section III.

Afterwards, the system seeks for long-standing services that provide similar functionalities to the new web service (Line 3). To evaluate the similarity between web services, we use the approach proposed by Tibermacine *et al.* [15]. If the *simServiceSet*, which denotes the set of similar service, is not empty, the system builds an equation model from the QoS vectors and reputation values of similar services (Line 5 in the Algorithm), as it is detailed in Section IV. The system estimates from the built model a reputation value, denoted *mReputation*, for the newcomer service based on its initial QoS vector. The maximum value between the provider reputation and the estimated reputation (*mReputation*) is assigned to the newcomer service. The choice of the maximum value (an optimistic strategy) is motivated by the fact that if a provider has a good reputation, it is likely that its new service will have a good reputation too. If the estimated reputation is better, then we give a chance to this service to be selected and evaluated by users, independently of its provider reputation.

Besides, when the provider of the service is also new to the system, we check if it is a whitewashing situation (lines 12-17 in Algorithm 1). The system retrieves all similar long-standing services and compares their similarity scores with the new service (similarity scores range between 0 and 1, where 1 means that services are totally similar and 0 otherwise). If the highest similarity score equates to one, and the similar service has left the system, then, the provider of the new service becomes suspicious, and we assign the (old) reputation of the service that has left to the new service (line 17 in the algorithm). Otherwise, we go to phase two, where the system builds an equation model from QoS and reputation values of similar services. The estimated reputation is assigned to the newcomer service.

When the newcomer service and its provider are both new, and there are no similar services in the system, we go to phase three (detailed in Section V). The system builds a multiple linear regression model from QoS vectors and reputation values of all long-standing web services in the system (line 23 in the algorithm). Likewise to phase two, the model gives also an estimation of service reputation based on the service initial QoS. The estimated value is assigned to the newcomer web service.

Input: S_i new service
Output: Reputation
Begin
1: **if** (Provider(S_i) $\in$ *ProviderList*) **then**
2: prReputation = providerReputation(Provider(S_i));
3: simServiceSet = getSimilarServices(S_i,*ServiceList*);
4: **if** (simServiceSet$\neq \Phi$) **then**
5: eModel = buildEquationalModel(simServiceSet);
6: mReputation = estimateReputation(Qos(S_i),eModel);
7: **reputation**= Max(prReputation,mReputation);
8: **else**
9: **reputation**= prReputation;
10: **end if**
11: **else**
12: simServiceSet = getSimilarServices(S_i,*ServiceList*);
13: **if** (simServiceSet$\neq \Phi$) **then**
14: SimVector = Similarities(S_i, simServiceSet);
15: aService =HighestScoreService(SimVector);
16: **if** (Max(simVector)==1 && hasLeft(aService)) **then**
17: **reputation** =getReputation(aService);
18: **else**
19: eModel = buildEquationalModel(simServiceSet);
20: **reputation** = estimateReputation(Qos(S_i),eModel);
21: **end if**
22: **else**
23: rModel = BuildRegressionModel(serviceList);
24: **reputation** = estimateReputation(Qos(S_i),rModel);
25: **end if**
26: **end if**
End

Algorithm 1: Reputation bootstrapping algorithm

III. Provider reputation

The reputation of a given provider mainly depends on the quality of its offered services, thus on their reputation. In this phase, we assess the reputation of a provider as the average of the reputation scores of its services. Given a provide Pr_x, let $Services(Pr_x) = S_1, S_2, ..., S_n$ be the set of web services provided by Pr_x. The reputation of this provider is assessed as follows:

$$RP(Pr_x) = \begin{cases} \frac{(\Sigma_{i=1}^{n} R(S_i))}{n} & if \quad Services(Pr_x) \neq \phi \\ 0 & Otherwise \end{cases} \tag{1}$$

where,

- $R(S_i)$ is the reputation of service S_i that belongs to the provider's service set ($S_i \in services(Pr_x)$).
- ϕ denotes an empty set.

The reputation of a new provider, in case of the introduction of a new service, is set to 0. This value means that the provider is by default not trusted. However the reputation of the provider is updated automatically once the reputation of one of its services has been modified.

IV. Reputation estimation from similar services

Since a user rates similar web services based on the same criteria, it could be possible to estimate the reputation of newcomer web service based on reputation values of its similar services, which are aggregations of user feedback ratings. In this phase, we present how to estimate, according to QoS values, the reputation of a newcomer service S_i. For illustration, we suppose that service S_i comes with three initial QoS values: i) response time, denoted T_i or in general form $Q_{i,1}$; ii) availability, denoted A_i or $Q_{i,2}$; and iii) price, denoted P_i or $Q_{i,3}$.

To estimate the reputation of service S_i, we follow the next steps:

1) *Selecting similar services:* First, the system selects from its database long-standing services that are similar to service S_i. We use the approach proposed by Tibermacine *et al.* [15] to assess the similarity between the new web service and the long-standing services. The approach assesses the similarity between two services by comparing their WSDL files using several lexical and semantic metrics. The results of the similarity assessment are scores that range between 0 and 1, where 0 represents a total dissimilarity and 1 a total similarity. Compared web services with a similarity score greater or equal than a fixed threshold (e.g., 0.75) are considered as similar. The result of this step is a set of similar services denoted by $simServiceSet$.
2) *Preparing QoS data:* Second, the system retrieves QoS and reputation values of each service in $simServiceSet$. Let $S_j \in simServiceSet$ (j=1,..,m) be a similar service. Each similar service S_j has a QoS vector $Q_{S_j} =< T_j, A_j, P_j, R_j >$. Where R_j denotes the reputation of service S_j calculated from user feedback ratings. Table I groups the collected data. Besides, the newcomer service S_i has the vector $Q_{S_i}^{init} =< T_i, A_i, P_i, R_i? >$. Where T_i, A_i and P_i represent the initial QoS values, and $R_i?$ is the unknown reputation value.
Afterwards, the system scales all QoS values in the interval $[0, 1]$. Thus, each QoS value ($QosVal$),which is T_j, A_j, or P_j (j=1,..,m) in Table I, is replaced by $NewQosVal$ as follows :

$$NewQosVal = \frac{QosVal - MinVal}{MaxVal - MinVal} \tag{2}$$

where, $MinVal$ and $MaxVal$ are respectively the minimum and maximum recorded value in the system

Service	Response Time (T)	Availability (A)	Price (P)	Reputation (R)
S_1	T_1	A_1	P_1	R_1
S_2	T_2	A_2	P_2	R_2
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
S_m	T_m	A_m	P_m	R_m

Table I
GENERIC FORM OF SERVICE QOS VECTORS AND REPUTATION

	S_τ	S_i
Response Time (T)	$T_\tau = \frac{\Sigma_{i=1}^{m} T_i}{m}$	T_i
Availability (A)	$A_\tau = \frac{\Sigma_{i=1}^{m} A_i}{m}$	A_i
Price (P)	$P_\tau = \frac{\Sigma_{i=1}^{m} P_i}{m}$	P_i
Reputation (R)	$R_\tau = \frac{\Sigma_{i=1}^{m} R_i}{m}$	$R_i?$

Table II
MEAN QOS VALUES

for that QoS metric. Note that, some of the QoS metrics have values which are interpreted inversely, i.e. the higher is the value, the lower is the quality. This includes execution time and price. Thus, the scaled values $NewQosVal$, for these type of QoS, is calculated as follows:

$$NewQosVal = 1 - (\frac{QosVal - MinVal}{MaxVal - MinVal}) \quad (3)$$

3) *Solving an equation system:* We assume that reputation and QoS are collinear. Thus, from the scaled data of Table I, we define the equation system of m equations with 3 variables (QoS attributes) as follows:

$$R(S_k) = \begin{cases} T_1.X_1 + A_1.X_2 + P_1.X_3 = R_1 \\ T_2.X_1 + A_2.X_2 + P_2.X_3 = R_2 \\ T_3.X_1 + A_3.X_2 + P_3.X_3 = R_3 \\ \vdots \quad\quad\quad\quad\quad\quad \vdots \\ T_m.X_1 + A_m.X_2 + P_m.X_3 = R_m \end{cases} \quad (4)$$

where, X_1, X_2 and X_3 are the variables, the coefficients T_i, A_i and P_i are the QoS values, and R_i is the reputation value ($R(S_i)$). Each equation in the system represents the relation between QoS values and the reputation of each service. Finally, we solve the equation system to find values of X_1, X_2 and X_3.

4) *Evaluating reputation:* Finally, Once the equation system is solved, we can find the reputation of the newcomer service $R_i?$ by direct application of the values of X_1, X_2 and X_3 in the following equation:

$$R_i? = T_i.X_1 + A_i.X_2 + P_i.X_3 \quad (5)$$

In case the equation system has no solution for the m equations, we eliminate from the system one equation, which includes the oldest assessed value of reputation, and we solve the equation-system again. We keep eliminating equations and solving the system till we find a solution, and m is still greater or equals 3. In case we could not get a solution for the equation system, we assess the reputation of the new web service as follows:

- *Construct the mean vector of similar-services' QoS*: For instance, column S_τ in Table II represents the mean vector for the QoS values in Table I. And, column S_i represents the vector that holds QoS data of the new service.
- *Compute the reputation $R_i?$ of the new service S_i* by applying the following formula:

$$R_i? = \frac{R_\tau}{3} \times (\frac{T_i}{T_\tau} + \frac{A_i}{A_\tau} + \frac{P_i}{P_\tau}) \quad (6)$$

This formula stipulates that the ratio of the newcomer service reputation to the reputation of its similar services is equal to the average of the ratios of the QoS attributes of the newcomer service to those of its similar services.

V. REGRESSION-BASED REPUTATION ESTIMATION

The third phase in the reputation bootstrapping technique is the construction of a multiple regression model, using QoS and reputation data of all long-standing web services. This model serves as an estimation of reputation for new services that has no previous interactions, nor they have known providers.

By definition, multiple regressions are statistical techniques used for predicting unknown Y values (dependent variable) corresponding to a set of X values (independent variables). In our study, the multiple regression is expected to give a model that could relate the reputation value of long-standing services to their QoS metrics, that is, we consider the dependent variable Y to represent the reputation of services as a function of multiple QoS attributes (independent variables) such as *response time, availability, throughput, latency, price,* etc. So, if we have n long-standing services in the system (S_j , $j = 1, 2, ..., n$), and each service S_j has a QoS vector $Q_{S_j} =< Q_{j,1}, Q_{j,2}, ..., Q_{j,k} >$ that holds k QoS metrics, and each service S_j has a reputation value $R(S_j)$ denoted R_j. The relationships between reputation (dependent variable) and QoS metrics (independent variables) can be expressed by the following equation:

Number	Quality	Description	Unit
1	Response time	Time taken to send a request and receive a response	ms
2	Availability	Number of successful invocations / total invocations	%
3	Throughput	Total Number of invocations for a given period of time	invocations/second
4	Successability	Number of response messages / number of request messages	%
5	Reliability	Number of error messages / total messages	%
6	Compliance	The extent to which a WSDL document follows WSDL specification	%
7	Best Practices	The extent to which a Web service follows WS-I Basic Profile	%
8	Latency	Time taken for the server to process a given request	ms
9	Documentation	Measure of documentation (i.e. description tags) in WSDL	%

Table III
QOS METRICS SELECTED FROM QWS DATASET

$$\underbrace{\begin{pmatrix}\beta_1\\ \beta_2\\ \vdots\\ \beta_n\end{pmatrix}}_{\beta}\underbrace{\begin{pmatrix}Q_{1,1} & Q_{1,2} & \cdots & Q_{1,m}\\ Q_{2,1} & Q_{2,2} & \cdots & Q_{2,m}\\ \vdots & \vdots & \ddots & \vdots\\ Q_{n,1} & Q_{n,2} & \cdots & Q_{n,m}\end{pmatrix}}_{X}+\underbrace{\begin{pmatrix}\varepsilon_1\\ \varepsilon_2\\ \vdots\\ \varepsilon_n\end{pmatrix}}_{\varepsilon}=\underbrace{\begin{pmatrix}R_1\\ R_2\\ \vdots\\ R_n\end{pmatrix}}_{Y} \tag{7}$$

Where :

- X is the design matrix that packs all regressors (predictors) $q_{i,j}, i = 1, ..., n$ and $j = 1, ..., m$.
- β is the regression coefficient vector (called also slop vector).
- ε is the error vector. Error terms $\epsilon_i, i = 1, .., n$ capture all the factors which influence the dependent variable $(R_i, i = 1, ..., n)$ other than regressors $(X_{i,j}, i = 1, ..., n$ and $j = 1, ..., m)$.

The multiple regression of the model can be simplified to:

$$Y = \beta_0 + \beta_1 Q_1 + \beta_2 Q_2 + ... + \beta_k Q_k + \epsilon \tag{8}$$

Where,

- Y is the response (estimated reputation) of the linear combination of the model terms.
- β_i $(i = 1, ..., k)$ represents the unknown coefficients.
- ϵ is the error term.

At the end of phase three, the system uses solved values of the unknown coefficients (β_i $(i = 1, ..., k)$, and the error term (ε), to estimate the reputation of the new comer services based on its initial QoS vector.

VI. EVALUATION

To evaluate the proposed bootstrapping technique, we conducted our experiment on a set of real web services data extracted from WSDream [16] and QWS [17] datasets. The aim of the experiment is to study the feasibility of the proposed technique, and to guarantee its efficiency in estimating reputation values of newcomer Web services. We focused on phase three (the construction of the multiple regression model), which is the worst (and somehow the most general) case. We evaluated the regression model regarding to its coefficient of determination (R^2) and its F-significance. Moreover, we tested the performance of the model by comparing estimated reputation values and reputation values assessed during a simulation, where we pretend that newcomer services are old in the system, and they have received feedback ratings from different users (both honest and malicious users are considered). Comparison results are reported by the Mean Absolute Error (MAE), and the Percentage-Error metrics.

A. Data description and preparation

WSDream dataset holds 5825 web service QoS data evaluated by 339 users in different geographical locations (we have chosen the dataset 2 in WSDream). The dataset holds 339*5825*2 (2: response time and throughput). QWS dataset holds 365 web services with 9 QoS metrics listed in Table III.

To use a maximum number of QoS metrics with different monitored values of response time and throughput, we selected the services that belong to the two datasets. We matched web services based on their URIs, Names, and WSDL file size. We obtained 409 services that constitute the intersection set. Each service in this set has 7 fixed QoS metrics from QWS, and 2 QoS metrics (response time and throughput) that vary based on the observation of 339 users from WSDream. This final web service set is used for experimentation.

B. Feedback rating simulation

Due to the current limited availability of feedback rating data, many web service reputation management approaches (e.g., [6], [9], [13]) have used simulation for generating user feedback ratings for assessing service reputation values. Likewise, we have built a Java program that simulates the

Regression Statistics	
Multiple R	0.886815467
R Square	0.786441673
Adjusted R Square	0.781072888
Standard Error	0.272880252
Observations	368

ANOVA	df	SS	MS	F	Significance F
Regression	9	98.16965139	10.90773904	146.4841128	2.8767E-114
Residual	358	26.6579802	0.074463632		
Total	367	124.8276316			

	Coefficients	Standard Error	t Stat	P-value	Lower 95%	Upper 95%	Lower 95.0%	Upper 95.0%
Intercept (ε)	2.374640371	0.219162569	10.83506359	7.52464E-24	1.943632519	2.805648222	1.943632519	2.805648222
Response Time	-0.000346194	0.00529994	-0.065320349	0.09479554	-0.010769122	0.010076734	-0.010769122	0.010076734
Throughput	-0.000396292	0.000137447	-2.883247406	0.004173789	-0.000666596	-0.000125988	-0.000666596	-0.000125988
Availability	0.009825812	0.005457871	1.800301188	0.072654752	-0.000907706	0.02055933	-0.000907706	0.02055933
Successability	0.013471355	0.005074587	2.654670304	0.008292994	0.003491609	0.0234511	0.003491609	0.0234511
Reliability	-0.000504619	0.002832241	-0.178169433	0.85869069	-0.00607454	0.005065303	-0.00607454	0.005065303
Compliance	0.020737736	0.001661812	12.47899235	6.38821E-30	0.017469597	0.024005876	0.017469597	0.024005876
Best Practice	-0.005847026	0.002303954	-2.537822196	0.011577834	-0.010378011	-0.001316041	-0.010378011	-0.001316041
Latency	-0.000348913	9.14902E-05	-3.813661459	0.000161212	-0.000528838	-0.000168987	-0.000528838	-0.000168987
Documentation	0.007401319	0.000420714	17.59226209	2.2905E-50	0.006573936	0.008228701	0.006573936	0.008228701

Estimated reputation = 2.374640371 - (0.000346194 * Response Time) - (0.00396292 * Throughput) + (0.009825812 * Availability) + (0.013471355 * successability) - (0.000504619 * Reliability) + (0.020737736 * Compliance) - (0.005847026 * Best Practice) - (0.000348913 * Latency) + (0.007401319 * Documentation)

Figure 1. Output from the regression data analysis tool

interaction between the selected (409) web services and (339) users.

Each service has an actual performance (overall quality) level (from 1 to 10), denoted *PerfVal*, that represents in a scale of 10 how good is the overall quality provided by the service. PerfVal is calculated based on a utility function (i.e., a single scalar metric to quantify quality perception) of the delivered service, as suggested in [3]. However, in our work, we propose to calculate the utility function with the root mean square, which is a measure of the magnitude, of the scaled QoS metrics. Thus, *PerfVal* of service S_i is assessed as follows:

$$PerfVal(S_i) = 10 \times \sqrt{\frac{\sum_{j=1}^{k} Scal(Q_{i,j})^2}{k}} \quad (9)$$

Where, k is the number of used QoS metrics (Q_j, $j = 1, .., k$). And, $Scal(Q_{i,j})$ is the scaling function, which is defined by Eq. 10, if the quality is positive (i.e., the higher is the value the higher is the quality), and by 1 - the same formula otherwise.

$$Scal(Q_{i,j}) = \frac{Q_{i,j} - Min(Q_j)}{Max(Q_i) - Min(Q_j)} \quad (10)$$

where, $Min(Q_j)$ and $Max(Q_j)$ are respectively the minimum and maximum recorded values of the quality Q_j.

The program simulates two kinds of users: honest and malicious users. Honest users randomly rate a service based on its *PerfVal* within the interval $[Max(0, \textit{PerfVal} - 2), Min(\textit{PerfVal}+2, 10)]$, e.g, if *PerfVal*=7, honest feedback ratings could be 5, 6, 7, 8, and 9. the deviation with ± 2 from *PerfVal* represents natural variation between user opinions. Malicious users randomly rate the same service outside the previous interval, always on scale of 10. For instance, if *PerfVal*=7, malicious feedback ratings could be 0, 1 ,2, 3, 4, and 10.

In this simulation, we consider 10% of users are malicious users. Finally, the reputation of web services are calculated as the mean of collected feedback ratings. The final reputation values are the average of 10 round-simulation results.

C. Multiple regression model building

We build the regression model using the QoS and reputation data of 90% of web services (Training set). The remaining 10% of services are used as service-test set (i.e., services considered as newcomers). We have used Microsoft Excel Regression data analysis tool to build our multiple regression model, because it provides a detailed analysis results about the regression model as it is shown in Figure 1. The same model is also generated by our Java program to estimate the reputation of the elements of the test set.

As we can see from Figure 1, the correlation coefficient (Multiple R=0.8869) indicates a positive relationship between reputation value and QoS data (where, 1 indicates a perfect positive relationship). We see also that $R^2 = 0.7864$ which means that 78.64% of the values fit the model.

Finally, since the significance-F (p-value) = 2.8767e-114 < 0.05, we conclude that the regression model is a significantly good fit (i.e, the calculated linear equation fits our data). For estimating web service reputation, we use the formula depicted in the bottom of Figure 1.

D. Reputation estimation and evaluation

We test the efficiency of the multiple regression model in the estimation of web service reputation, using test web services data (the remaining 10% of web services). We

Malicious Density	R	MAE	Percentage Error
10%	0.776797	0.29042	5.06319
20%	0.760624	0.29270	5.04946
30%	0.75072	0.26922	4.57920

Table IV
R^2, MEA AND PE COMPARISON BY VARYING MALICIOUS DENSITY

compare the estimation reputation value of each service obtained from the model, with the reputation value calculated from user feedback ratings collected during multi-round simulation. The **Mean Absolute Error** (MAE) between the estimated and calculated reputation values of the 41 test web services is equal to **0.2651**, with a **Percentage Error** (PE) of **4.656%**. A comparison between estimated and calculated reputation values of a sample of test services is shown in Figure 2. We have run multiple experiment rounds by varying the density of malicious users (i.e., rate of malicious users in the system). We have built regression models from the obtained results using the same training set (90% of services). Moreover, we tested regression models using the same test service set. Table IV lists the recorded R^2, MAE and Percentage Errors. As we can see, with the variation of malicious density, the regression models provide good results (e.g., R^2= 0.7507 with Malicious density = 30%). In addition, the obtained MAE and PE values indicate that there is a slight variation of the estimated reputation values. This variation is caused by the generated models.

Finally, we may safely use the estimated reputation value from the built regression model for bootstrapping the reputation of the newcomer web services. Because, even with greater deviation, the use of these bootstrapped reputation values is still better than assigning initial high (maximum), low (minimal), or any fixed (*e.g.,* average) reputation values.

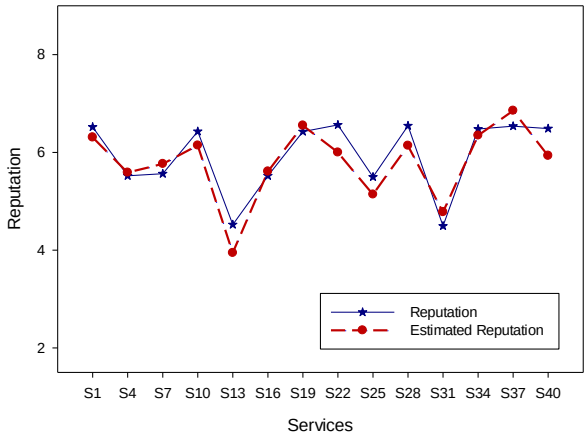

Figure 2. Calculated and estimated reputation of a sample of test services

VII. RELATED WORK

Though many studies have been conducted on trust and reputation for Cloud and Web service systems (e.g. [1], [2], [4]), a little effort has been dedicated to the study of reputation bootstrapping.

In fact, most of the proposed reputation approaches are neglecting this aspect, and only a few of the them provide default bootstrapping techniques [11], [18], [19]. i.e., they assign a default constant reputation score, such as 0, high (the maximum) or no reputation, to all newcomers. For instance, Zacharia *et al.* [18] give the minimum possible trust value. However, in such situation, newcomers may never have chance to get selected. Even in the case of assigning high trust values, the problem of "whitewashers" (i.e. malicious participants leave the system and come back with new identities) could raise.

The framework proposed by Jin-Dian *et al.* [20] assigns the provider reputation to its newly posted services. The authors suggest to assess the reputation of the provider based on its past experiences. However, the problem raises again if the provider is a newcomer in the recommendation system.

Feldman and Chuang [12] propose a solution for bootstrapping reputation of newcomer services based on its probability of deceiving. This probability is computed by collecting all transaction information of the newcomer's first-time interaction. Moreover, this approach is community-based, and newcomer reputation is adjusted to the reputation of others. However, initial reputation scores are still not fair and they do not reflect the actual reputation of newcomers.

Malik and Bouguettaya [9] propose two bootstrapping techniques for establishing reputation of newcomer web services. The first is an adaptive technique that assigns the initial reputation value based on the rate of maliciousness in the system. The second approach assigns a default reputation score to a newcomer service, where the former purchase the initial reputation from the community provider. Or, the community requests some evaluators (elder service with high reputation) to evaluate the newcomer service in a short period of time. This work has some limitations. In the first technique the reputation of a specific web service is related to the maliciousness rate in the community, which seems highly penalizing or rewarding based on a factor that is unrelated to the service itself. In the second technique, the contribution and the impact of requesters on the reputation of web service is very high, which raises the problem of the trust of evaluators themselves.

Huang *et al.* [8] propose an equitable trustworthy mechanism that enables new services to startup and grow in an ecosystem environment. The mechanism distinguishes between novice and mature services during service recommendation. The approach considers two trust bootstrapping strategies: i) default strategy where they assign to the newcomer a default initial trust value, and ii) an adaptive boot-

strapping strategy where they assign to the newcomer the average trust value in the system. However, the first strategy do not provide any solution for the cold-start problem nor to the whitewashing problem in case of assigning a high value. Moreover, the second technique assigns the average trust in the system to newcomer services, which is not an accurate solution (e.g. the case where the average is high and the service is bad or the inverse).

The main differences between our solution and the previous solutions are summarized as follows:

- Instead of assigning the same reputation value to all newcomer web services, we propose to estimate the initial reputation value of the newcomer service based on its provider reputation, reputation of its similar long-standing services, and its initial QoS values.
- When the previous values are missing, we make an estimation using a multiple regression model built from the reputation and QoS values of long-standing services.
- We propose a solution to overcome the whitewashing problem based on the similarity of the newcomer service with registered services that have left the system.

VIII. Conclusion

Initial reputation values assigned to newcomer web services have their impact on the performance of web service recommendation systems. In this paper, we proposed a technique that bootstraps reputation of newcomer web services. The technique depends on a multiple regression model for the estimation of the initial reputation scores. This model is build with QoS and reputation values of long-standing web services in the system. Moreover, when providers are known by the recommendation system, the technique evaluates the initial reputation of the new services starting from their provider reputation, and an estimated reputation value obtained from similar web services. The experimental results showed the practicability and the efficiency of the proposed solution. As a future work, we would like to investigate empirically the concrete correlation kind between reputation and QoS (to not make the hypothesis of collinearity). This will potentially enable us to obtain more accurate results.

References

[1] T. H. Noor, Q. Z. Sheng, S. Zeadally, and J. Yu, "Trust management of services in cloud environments: Obstacles and solutions," *ACM Comp. Surveys*, vol. 46, no. 1, p. 12, 2013.

[2] F. Hendrikx, K. Bubendorfer, and R. Chard, "Reputation systems: A survey and taxonomy," *Journal of Parallel and Distributed Computing*, vol. 75, pp. 184–197, 2015.

[3] N. Limam and R. Boutaba, "Assessing software service quality and trustworthiness at selection time," *IEEE Transactions on Software Engineering,*, vol. 36, no. 4, pp. 559–574, 2010.

[4] F. G. Mármol and M. Q. Kuhnen, "Reputation-based web service orchestration in cloud computing: A survey," *Concurrency and Computation: Practice and Experience*, 2013.

[5] Z. M. Aljazzaf, "Trust-based service selection," Ph.D. dissertation, The University of Western Ontario, 2011.

[6] H. T. Nguyen, J. Yang, and W. Zhao, "Bootstrapping trust and reputation for web services," in *Commerce and Enterprise Computing (CEC), 2012 IEEE 14th International Conference on*. IEEE, 2012, pp. 41–48.

[7] J. B. Schafer, D. Frankowski, J. Herlocker, and S. Sen, "Collaborative filtering recommender systems," in *The adaptive web*. Springer, 2007, pp. 291–324.

[8] K. Huang, Y. Liu, S. Nepal, Y. Fan, S. Chen, and W. Tan, "A novel equitable trustworthy mechanism for service recommendation in the evolving service ecosystem," in *Service-Oriented Computing*. Springer, 2014, pp. 510–517.

[9] Z. Malik and A. Bouguettaya, "Reputation bootstrapping for trust establishment among web services," *Internet Computing, IEEE*, vol. 13, no. 1, pp. 40–47, 2009.

[10] J. R. Douceur, "The sybil attack," in *Peer-to-peer Systems*. Springer, 2002, pp. 251–260.

[11] M. Chen, L. He, X. Cai, and W. Xia, "Trust evaluation model for composite service based on subjective logic," in *Proc. of IIHMSP'08*. IEEE, 2008, pp. 1482–1485.

[12] M. Feldman and J. Chuang, "The evolution of cooperation under cheap pseudonyms," in *Proc. of CEC'05*. IEEE, 2005, pp. 284–291.

[13] Z. Malik and A. Bouguettaya, "Rateweb: Reputation assessment for trust establishment among web services," *VLDB Journal*, vol. 18, no. 4, pp. 885–911, 2009.

[14] M. H. Hasan, J. Jaafar, and M. F. Hassan, "Monitoring web services quality of service: a literature review," *Artificial Intelligence Review*, vol. 42, no. 4, pp. 835–850, 2014.

[15] O. Tibermacine, C. Tibermacine, and F. Cherif, "A practical approach to the measurement of similarity between wsdl-based web services," *Revue des Nouvelles Technologies de l'Information*, vol. 6th French-speaking Conference on Software Architectures, RNTI-L-7, pp. 03–18, 2014.

[16] Z. Zheng, Y. Zhang, and M. R. Lyu, "Distributed qos evaluation for real-world web services," in *Proc. of ICWS'10*. IEEE, 2010, pp. 83–90.

[17] E. Al-Masri and Q. H. Mahmoud, "Qos-based discovery and ranking of web services," in *Computer Communications and Networks, 2007. ICCCN 2007. Proceedings of 16th International Conference on*. IEEE, 2007, pp. 529–534.

[18] G. Zacharia, A. Moukas, and P. Maes, "Collaborative reputation mechanisms for electronic marketplaces," *Decision Support Systems*, vol. 29, no. 4, pp. 371–388, 2000.

[19] T. D. Huynh, N. R. Jennings, and N. R. Shadbolt, "Certified reputation: how an agent can trust a stranger," in *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*. ACM, 2006, pp. 1217–1224.

[20] S. Jin-dian, G. He-qing, and G. Yin, "An adaptive trust model of web services," *Wuhan University Journal of Natural Sciences*, vol. 10, no. 1, pp. 21–25, 2005.