JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 39, 1321-1339 (2023)
DOI: 10.6688/JISE.202311_39(6).0005

Link-ability: A Compatibility Prediction Method of Wireless Service based on Markov Process*

SHIYANG MA, YULONG SHEN, XUEWEN DONG,
WEI TONG AND LINGXIAO YANG
School of Computer Science and Technology
Xidian University
Xi'an, 710071 P.R. China
E-mail: 503433614@qq.com; ylshen@mail.xidian.edu.cn;
xwdong@xidian.edu.cn; {wtong; lxyang}@stu.xidian.edu.cn

Compatibility is a decisive and necessary factor which impacts the successfulness and correctness of establishing connections among services. Reliable service parameter predictions on compatibility may not only provide assurance on fulfilling workflow, but also reduce the risk of unexpected wasting on time and expenses. Recent research on service's parameter prediction focus on two aspects: Collaborative Filtering and Matrix Factorization. Both of them faces challenges from data-sparse problem, lack of solid mathematical support, and weakness on tracking dynamic services. In this paper, we propose Link-ability, a parameter to reveal the compatibility of a service. Firstly, we propose an architecture called Topology-Retrievable Service Oriented Architecture (TSOA), which is capable of collecting successful invocation records from service requesters and generate the whole topology of the service pool in a specific period of time. Secondly, we bring up the Link-ability Generation Algorithm (LGA) and Markovian mathematical strategies to generate Link-ability value. The algorithm is supported by solid mathematical proof and capable of solving data-sparse problems. Both TSOA and LGA are upgraded to adapt the circumstances of services exiting the service pool (we define it as a Drop-out). Lastly, we design and conduct two series of experiment to strengthen the reliability and correctness of Link-ability. The result shows that Link-ability considers comprehensive information of the whole topology and bring 54.18% reduction in average error rate against Drop-out scenarios.

Keywords: service composition, compatibility prediciton, Markov process, service oriented architecture, wireless service

1. INTRODUCTION

Wireless services are benefiting more and more people in real-life use cases [1]. Since wireless services come out to cover various kinds of tasks from service requesters, the growing complexity of application scenarios [2] and the demand for customized solutions [3] are becoming major challenges to researchers [4]. Fortunately, scholars are inspired to decompose the composite task into basic tasks, design solutions to basic requirements, and reconstruct them into a final composite strategies [5]. That's how service composition was born [6]. As the name suggests, service composition is a procedure to

Received September 30, 2022; revised October 30, 2022; accepted December 1, 2022.
Communicated by Changqiao Xu.

* The preliminary version was presented at the Networking and Network Applications (NaNA) in 2022.

decompose the complex requirements, select proper services, combine different services together within expected workflow [7] to cover the requirement of a request from a service requester. During the process of service composition, parameter-prediction is an important academic direction because it helps to foretell the effectiveness and functionality of the service composition.

Recently, research works in service parameter prediction mainly focus on two aspects: Collaborative Filtering (CF) [8] based prediction [9, 10] and Matrix Factorization (MF) prediction [11]. In order to further improve the prediction accuracy, some more sophisticated CF-based quality prediction methods were proposed. They can be categoried into two approaches. The first category attempts to dig deeper into the user-service matrix so that more information can be extracted to operate predictions [12–16]. The second category of methods try to request for additional information from users and services to strengthen the reference of prediction [17, 18].

As one of service's property parameters, compatibility [19] deserves more attention in the process of service composition because it guarantees the successfulness and correctness of the workflow within expectations. In real-life use-cases, the market is compatitive, services with same kind compete with each other. If there's no effective evaluation on compatibility of the service, service requesters will be confused and exposing themselves to risk wrong choices, causing failure to the workflow, and an unnecessary waste of expenses and time. Therefore, a reliable compatibility prediction strategy is an inelastic demand of selecting proper services.

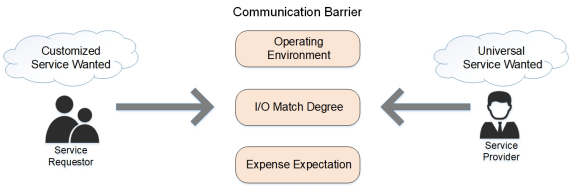


Fig. 1. Compatibility problem.

Existence of compatibility problem is inevitable. As is shown in Fig. 1, firstly, there's a communication barrier between service providers and requesters. In most use-cases, there's no direct communication approaches for providers and requesters. Service requestors can only purchase services within UDDI, not directly from providers. Secondly, it is not realistic to make one on one service for each service requester. Thirdly, service providers and service requesters possess different standpoint. Unlike service provider aims to bring up universal services, service requesters are looking for customized candidate to fulfil the workflow. Every service requester possesses a specific operating environment, a strict I/O match degree, and a specific range of price or time-cost, which is not possible to be satisfied by every candidate services.

To solve the problems above, this paper propose Link-ability, a Markovian Process-based mechanism to evaluate and quantify the compatibility of services. The Markovian perspective focus on the connection between the current service and the coming candidates, which maximum the possibility to fit I/O restrictions. On the other hand, the mechanism collects successful records from other requestors to work on the whole topology of the service pool, which weakens the communication barrier and gives a convincing

reference on the specific operating environment.

Compatibility prediction is quite challenging in three major reasons. Firstly, compatibility is impacted by operating environment [20], and at the same time, operating environment is diverse among different requesters, *i.e.*, two users with the same service may have a totally different experience because of the distinction of the operating environment. Secondly, considering huge cost on time and expenses, it is impractical to inspect each candidate service to make an ultimate selection, especially when the amount of candidate services is way too large. Thirdly, the compatibility of a service is a subjective conception for a user to give remarks on [21], that is to say, giving users the authority to evaluate the compatibility can be misleading because remarks from users are not in the same standard.

Our research focus on proposing a reliable compatibility prediction strategy. We firstly propose a topology-retrievable architecture over SOA (Service Oriented Architecture), which is capable of generating the topology structure of the service pool by collecting individual invocation records from requesters. Secondly, we propose a parameter called Link-ability to evaluate compatibility of services, and Link-ability Generation Algorithm to to generate the Link-ability. Then, we advance the proposed strategy to strengthen its feasibility towards Drop-out problems. Lastly we operate two series of experiment to prove our point of view and conclude our research. Table 1 compares our work with some typical service prediction mechanism. In this paper, our contributions are included as follows:

- **Compatibility Parameters: Link-ability**
  In the traditional mechanism of SOA, there is no specific parameter to describe the ability of a specific service to connect other services in a workflow. We create a definition on the compatibility of a service called 'Link-ability'. The Link-ability functions as a parameter that quantize the probability to operate successful connections with other services, and we provide a solid mathematical proof on the uniqueness and correctness on our proposal.

- **Topology-Retrievable SOA (TSOA)**
  Existing SOA has a dull sense of current invocation situations of the whole service pool because the mechanism only records split individual invocation history instead of collecting records together. In this paper, we propose our Topology-Retrievable SOA which inherits the original functions of SOA, and at the same time optimizes the interaction procedure with service requesters to enable TSOA itself to evaluate the current situation of the whole service pool

- **Link-ability Generation Algorithm**
  This paper proposes a Markovian-process-based algorithm to generate the value of Link-ability. The mechanism makes compatibility predictions with invocation records collected by Topology-Retrievable SOA. Besides, the algorithm's reliability is guarenteed by Markovian property. We also propose a theorem to support and reveal rationality of the algorithm.

- **Drop-out Contingency**
  In realistic application scenario, services are dynamic. When a service exit the service pool, it may cause calculation error in Link-ability mechanism. This paper

proposes Drop-out contingencies to fix this calculation error and make the evaluation dynamic and accurate with the timestream.

The rest of this paper is organized as follows: Section 2 presents a new architecture designed for service composition with detailed explanation on its working process and advantages over traditional SOA. Section 3 illustrates our models and mathematical methods on Link-ability. Our experiment will be presented in Section 4 and we conclude our research in the last Section 5.

Table 1. Comparison of existing service prediction mechanism.

Mechanism	Anti-data sparse	Anti-subjective inference	Robustness	Universal	Drop-out contingency	Compatibility Prediction
Ref.[8]	✓	✓	×	×	×	×
Ref.[9]	✓	✓	✓	×	×	×
Ref.[10]	×	✓	×	✓	✓	×
Ref.[11]	✓	×	✓	✓	×	×
Ref.[12]	×	×	×	✓	×	×
Ref.[13]	✓	×	✓	×	×	×
Ref.[14]	✓	×	✓	×	✓	×
Ref.[15]	✓	✓	✓	×	×	×
Ref.[16]	✓	✓	✓	✓	×	×
Link-ability	✓	✓	✓	✓	✓	✓

2. ARCHITECTURE

This chapter proposes the architecture called Topology-Retrieving Service Oriented Architecture (TSOA). The TSOA remains the basic function of SOA, and expands its use cases to compatibility evaluation and provide contingency for permanent drop-out services.

Distinguished from the traditional triangle structure of SOA, we emphasize and strengthen the function of an important component, the service pool. Assuming that in a specific time-slot, service requesters return their successful invocation records, the UDDI has the ability to generate the topology of the service pool. It is quite challenging because in the aspect of traditional service composition, services in the service pool are considered to be discrete and non-associated. This situation changes as we propose our algorithm for UDDI center to generate a topology of the service pool. The generation is supposed to be based on the actual invocation records of users who successfully operate the services and cover their tasks. As we extract the topology of the service pool, we process it in the aspect of Markov Procedure, and operate our calculations (See details in Section 3).

As is shown in Fig. 2, the procedure is divided into four phases: Standard phase, Link-ability generation phase, Invocation phase, and Drop-out & Recalculation phase. The Standard phase contains the basic function of SOA, it guarantees the registration of new services from service requester, uploads new services to the pool, and permits the service requester to access the information of services and make requests. The Link-ability generation phase shoulders the responsibility to make evaluations on the compatibility of services in the pool by calculating Link-ability values. To do so, the UDDI [22] shall collect successful invocation records from requesters [23], and generate the topology matrix of the service pool. With the help of Link-ability generation algorithm, the UDDI send the Link-ability values and ranking of services back to requesters as references. The Invocation phase includes the basic procedure of service selection, invocation, and information

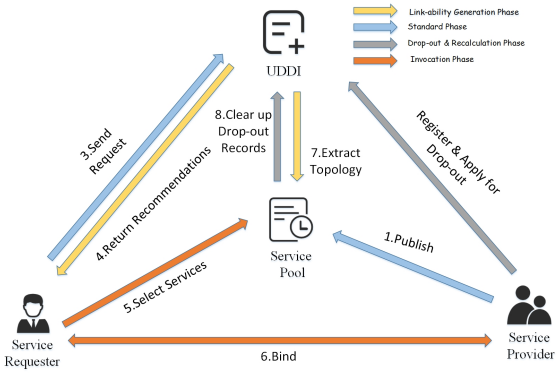


Fig. 2. Topology-retrievable SOA.

uploading. The Drop-out and Recalculation phase provides contingencies for services to make a permanent exit from the pool. When a service provider sends a request to withdraw his service from the pool, the UDDI releases the order to kick certain services out of the pool. Meanwhile, the service pool updates its state to record the change, note the service ID, and regenerate the topology of services in the pool. In the next time-slot, the change brought by drop-out services may cast influence on the Link-ability value, and UDDI center may adjust them due to the change of topology.

3. SYSTEM MODEL

3.1 Markov Chain via Service Composition

Compatibility deserves a high priority when it comes to service selection [24], because it guarantees the correct succession of the dataflow, and expected function of the workflow. In this paper, we mainly focus on the compatibility between two adjacent services. This point of view gives a proper and ingenious interface of the Markov Chain. A Markov Chain [25] is a mathematical system that experiences transitions from one state to another according to certain probabilistic rules. The defining characteristic of a Markov chain is that no matter how the process arrived at its present state [26], the possible future states are fixed. In other words, the probability of transitioning to any particular state is dependent solely on the current state and time. As we mentioned, the Markovian perspective focus on states and transitions. In this paper, services are modeled as states, and connections between services are treated as transitions. Mathematically, connections among services, or transition among states, stands with one necessary restriction, that is, transitions are time-irrelevant. Moreover, invocations only depend on services and transition steps, they can't be predicted. The restriction holds the Markovian property of every service, making the service pool stays as Homogeneous Markov Chain. The finite distribution of a Homogeneous Markov Chain, which presents as Link-ability distributions, can be determined by initial distribution and k-steps transition probability distribution.

We give a concise example to claim our perspective. As is shown in the Fig. 4, S1, S2 and S3 are previous services with successful invocation, and we are about to select a

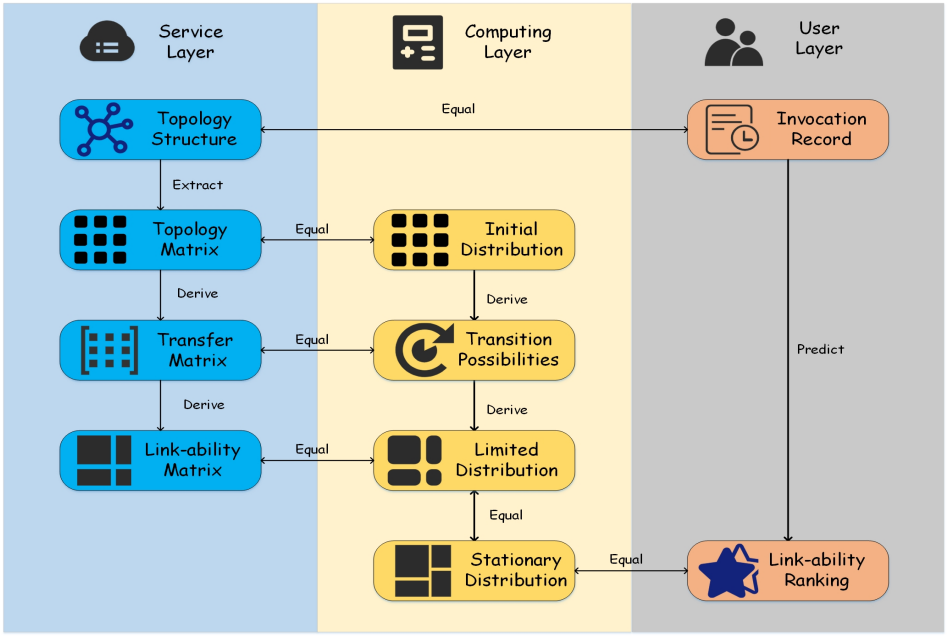


Fig. 3. Mathematical modeling on link-ability calculation.

new service, S5, to establish connection with our present service S4. The selection inherit the characteristic of Markov Chain the past services are treated as a black-box with no awareness of what services are included nor the structure of the previous workflow. In other words, the present service, S4, is the only service that matters the selection of the upcoming service.

In our proposed method, when a service request comes to the UDDI center, the UDDI center generates an adjacent matrix that describe the specific topology structure of the service pool. Next, it operates a set of calculations (we will discuss in the next section) to acquire the value of link-ability and annotate them on services in the pool [27]. Then, the requester receives a list of candidate services with link-ability annotations and operates a selection and finishes the composition procedure. Mathematically, the whole procedure stands with one necessary condition, that is, transitions among services are time-irrelevant. Furthermore, invocations can't be predicted, and only depend on services and transition steps. This condition holds the Markovian property of the every service, making the whole service pool stays as Homogeneous Markov Chain.

The Link-ability model takes the advantage of the Homogeneous Markov Chain. In our proposed model, Link-ability ranking serves as the finite distribution of the Markov chain. But how can we get the finite distribution? In Homogeneous Markov Chain, the computing strategy become crystal clear: if we acquire the initial distribution and the k-step transition probability distribution, the finite distribution, or, Link-ability Rankings, can be determind.

Theroem 1. *Service Link-ability's* $(\{X(n), n = 0, 1, \cdots\})$ *every finite dimensional proba-*

bility distribution can be determined by its initial distribution and k-step transition probabilty distribution.

Proof.

$$\begin{aligned}
&\textit{Given} \quad t_1 < t_2 < \cdots < t_n, t_k \in T; \quad k = 1,2,\cdots,n; \quad i_k \in E;\\
&P\{L(t_1) = i_1, L(t_2) = i_2, \cdots, L(t_n) = i_n,\}\\
&= \sum_{j\in E} \{L(0) = j\} P\{L(t_1) = i_1 | L(0) = j\} P\{L(t_2) = i_2 | L(t_1) = i_1\}\\
&\cdots P\{L(t_n) = i_n | L(t_{n-1}) = i_{n-1}\}\\
&= \sum_{j\in E} p_j p_{ji}^{t_1} p_{i_1 i_2}^{t_2 - t_1} \cdots p_{i_{n-1} i_n}^{t_n - t_{n-1}} \qquad (1)
\end{aligned}$$

The proposed Thereom 1 implies our computing strategy. Firstly, the topology matrix can be generated by collecting invocation record from service requesters, and we use it as initial distribution. Secondly, transfer matrix can be calculated within the strategy proposed in the coming section. The *k*-step transition probability distribution comes from iterations of transfer matrix. Holding the two distributions mentioned, we can finally calculate the Link-ability rankings according to Theorem 1.

Table 2. Notation definition.

Parameters	Definitions
i, j	$service_i, serivce_j$
k	*service requester*$_k$
$R_k[a_{ij}]$	*service requester*$'_k s$ Invocation Record
$T[b_{ij}]$	Topology Matrix of the pool
N_{cs}	number of services with invocation record
N_{is}	number of isolated services
σ	proportion of connected services
In	indegree matrix of every service
D	transfer matrix
A	Link-ability distribution matrix
x, z	tool vectors
cnt	number of steps of iteration
L	Link-ability
E	probability space

3.2 Model and Strategy

This chapter proposes the Link-ability Generation Algorithm shown as Algorithm 1. In Markovian vision, a transition is defined as a successful invocation from service i to service j. The Link-ability, a parameter we proposed in this paper, is designed to quanti-

fy the ability of a service to establish connections with other services, or in Markovian perspective, make transitions to another states.

Algorithm 1 : Link-ability Generation Algorithm

Input: $R_k[a_{ij}]$: Service requester k's invocation record;
$W[p]$: collection of Drop-out services' ID;
Output: $T[b_{ij}]$: Topology matrix of the service pool;
L: Services' link-ability Rankings;

$k \in [1,m]$; $i,j,p \in [1,n]$; $T[b_{ij}] = 0$; //Initialize topology matrix $T[b_{ij}]$
for each $i \in [1,n]$ **do**
  for each $k \in [1,m]$ **do**
    for each $j \in [1,n]$ **do**
      if $a_{ij} = 1$
then $b_{ij} = 1$
      end if
    end for
  end for
end for//Acquire $T[b_{ij}]$ as the topology matrix of the whole service pool.
for each $i \in [1,n]$ **do**
  if $i \in W$ **then**
$row[i] = 0$;
$colomn[i] = 0$;
  end if
end for// Remove connections made by Drop-out Services
$N_{cs} = sum(any(T[b_{ij}],2))$;
$N_{is} = n - N_{cs}$;
$\sigma = N_{cs}/n$;
$In = sum(T[b_{ij}],1)$;
$D = diag(1/In)$; //Initialize Transfer Matrix
$A = \sigma \cdot T[b_{ij}] \cdot D$;
$x = ones(n,1)/n$;
$z = zeros(n,1)$;
$cnt = 0$;//Initialize iteration steps
for $cnt = 0$ **do**
  if max(abs(x-z)) > 0.00001
then $z = x; x = A \cdot x$; $cnt = cnt + 1$
    // 0.00001 as convergence domain
  else
    $[L, index] = sort(x)$, $L = flipud(L)$, $index = flipud(index)$;
//sort services by the value of link-ability
  end if
end for

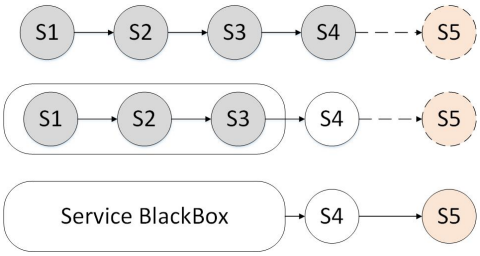


Fig. 4. Service composition within Markovian vision.

Assuming that there are totally n services in the service pool, making the record matrix from a service requester an $n*n$ square matrix. The mechanism collects all of the invocation records and sketch the topology structure of the service pool, which is denoted in the form of adjacent matrix, we call it Topology Matrix, $T[a_{ij}]$ (line 1-10, Algorithm 1). The parameter a_{ij} reflects the connection between two specific services, service i and service j. If they have successful connection history in the workflow, a_{ij} reaches the value 1, otherwise 0. We have $L_i(k)$ for the Link-ability of service i in the time slot k. And we give our calculation method.

(1) Services' initial states are determined by the topology matrix, 0 as no invocation record, 1 as connected with another service, and we consider the whole topology matrix as the initial matrix. Each service with connections are given a standard value called link-ability.

(2) In the next Round, each services with directed connections splits all of its value through the direction to the other end of the connection, and absorb the value from all services which has directed connections to it (line 14-16, Algorithm 1).

(3) Repeat Step (2) until the link-ability of every service converges at a small range. And finally we get the ultimate link-ability of each service (line 17-23, Algorithm 1).

We present our equations as follows.

$$\sigma = \frac{N_{ConnectedService}}{N_{TotalService}} \triangleq \frac{N_{cs}}{n}, \tag{2}$$

$$1-\sigma = \frac{N_{IsotatedService}}{N_{TotalService}} \triangleq \frac{N_{is}}{n}, \tag{3}$$

$$L_i(k) = \sigma \sum_{j=1}^{n} a_{ij} \frac{L_j(k-1)}{n_j} + (1-\sigma)\frac{1}{n}, \tag{4}$$

$$L_{NewMember\ i}(k) = \frac{1}{n}. \tag{5}$$

Eq. (4) explains the link-ability of service i in the Round k. The parameter n_j indicates the total number of services which have directed connections with service j, and other parameters are introduced in the previous section. However, the equation indicates an obvious characteristic of our method: if you want a result in Round k, you'd better check and only check the result in Round $k-1$. In other words, the calculation inherit the characteristic of Markov Chain: the probability of transitioning to any particular state is dependent solely on the current state and time (which we explained as Rounds). Furthermore, the link-ability in the current Round has completely no decisive impact on the next Round because the equation infers the fact that the current link-ability is decided by other services which have directed connections to the specific service in the last Round, not by itself. It's only the topology that matters.

3.3 Drop-out Contingencies

In practical use-cases, services come and go. The service pool maintains dynamic balance with new services joining and previous services exiting. Reasons to exit the pool can be various, such as permanent break-down, punishment from illegal behavior, withdrawn by service providers, *etc*. No matter what the reason is, exiting the pool is a normal and common behavior. We define an exit for a service in a specific time-slot as a Drop-out, and a service which makes a drop-out is called a Drop-out service.

Definition 1. ***Drop-out Service***
If a service exists in time-slot $k-1$, and exit the service pool in time-slot k, we call it a Drop-out service in the specific time-slot k.

Drop-out behavior causes miscalculation in Link-ability. In our proposed mechanism, the Link-ability value is calculated with the topology structure of the pool in the last time-slot. Assuming that in time-slot k, service j make a drop-out in a time-spot t_{s1}. Before t_{s1}, service j exists, along with its connections with other services. After t_{s1}, service j permanently vanishes from the pool, and there is no chance to establish any new connections. However, the problem is that the mechanism notes down connection records of the whole time-slot, even remains the records of drop-out services. When it comes to calculate the Link-ability value of time-slot $k+1$, the result will be interfered by drop-out services in time-slot k. Because drop-out services are gone, but their records remain, which will impact all the services connected before.

Strategies to erase the risk of miscalculation caused by drop-out services have to focus on three aspects, correcting the miscalculation, taking care of services' ID, and stabilizing the irreducible property of the transfer distribution. The deviation comes from remaining connection records of drop-out services. Systematically, every service must get the permission from UDDI [28] to make a drop-out. In other words, drop-out services' ID can be located by tracking UDDI records. Once the mechanism finishes collecting drop-out services' ID, we erase all of the connection record involved with them (Algorithm 1, line 11-16).

Mathematically, this step generates twice the number of drop-out services null vectors, which increase the potential to eliminate the irreducible property of the Markovian process. In order to overcome this problem, we force the drop-out services to operate a random walk [29]. Specifically, the transition doesn't stop as the last service finishes

its job, instead, it launches a random jump to another service [30]. There is no specific restriction on the random jump (such as the next target), and we consider it an equal probability to jump into other services. This strategy completely eliminate the presence of zero row vectors in the transition matrix, and the potential to break the irreducible property is overcome by our proposed Random Walk strategy.

3.4 Link-ability Generation Algorithm

How does invocation records predict the Link-ability rankings? Shown in Fig. 2, the mathematical strategy is to sketch the initial distribution of the Markov process, and finally reach the stationary distribution. Theorem 1 guarantees the correctness of the strategy. To realize the prediction, we propose the Link-ability Generation Algorithm (LGA), shown in Algorithm 1. The LGA consists of two parts. Shown in Fig. 5, in the first part, the LGA collects invocation records, generate the topology of the service pool, and transform it into the initial distribution (Algorithm 1, line 1-10). The second part of LGA shoulders the responsibility to derive the one-step transition probability distribution, and operate iterations until the finite distribution reach the convergence domain (Algorithm 1, line 17-28). Based on these two fundamental functions, there are still some details to be explained. Firstly, the whole strategy stands within one important condition: the Markov Process must be irreducible and homogenous. Mathematically, it means no zero vectors in the distribution. To hold the whole procedure obey this condition, we assume every service make a random walk (Algorithm 1, line 17-21), which means a random transition to other services with equal possibility. This random-walk assumption eliminates the zero elements in the distribution matrix, and keep the whole Markov process irreducible and homogenous. Besides, eliminating zero elements also prevents data-sparse problem in iterations. Secondly, in real-life use cases, services may drop-out of the pool permanently. These services exit the pool but leaves their previous records in the last time-slot. This behavior generates errors in predicting compatibility and zero vectors to interrupt the irreducibility of the Markov process. To overcome this problem, we operate a scan of every service to pick out the drop-out services, and eliminate their records (Algorithm 1, line 11-16). This strategy erases the negative impact brought by drop-out services.

4. EXPERIMENT

Practice is the sole criterion for testing truth. In this section, we bring Link-ability mechanism to practical use-cases. We observe Link-ability's functional performance in two aspects: the ability to reveal compatibility, and the robustness against drop-out contingencies. The experiment we conduct is based on the proposed Topology-Retrievable SOA and Link-ability Generation Algorithm.

4.1 Experimental Settings

Procedure: The procedure of the experiment runs like this: As is shown in Fig. 5, invocation records from service requesters in a specific time-slot will be simulated as an input of Link-ability Generation Algorithm (LGA). The LGA has two phases to process the records. In Phase 1, the LGA collects the invocation records and sketch the topology of the service pool. In Phase 2, LGA generates Link-ability values and rank them into a

list called Link-ability Rankings. Meanwhile, in order to operate a cross reference against Link-ability, we bring up 'Reference Value'(RV), which can be extracted from invocation records.

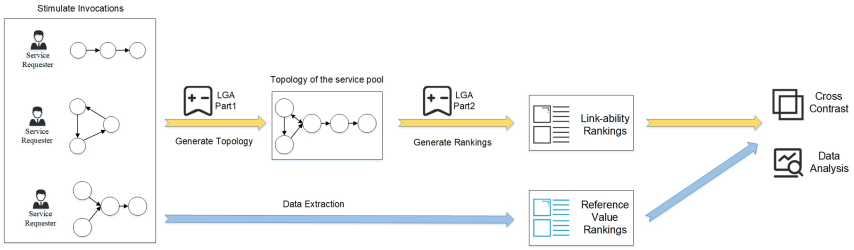
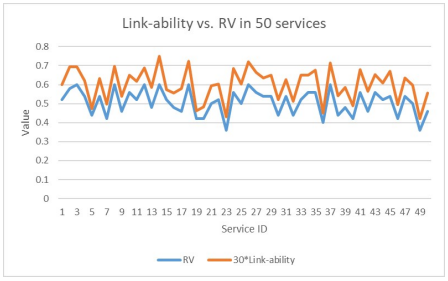
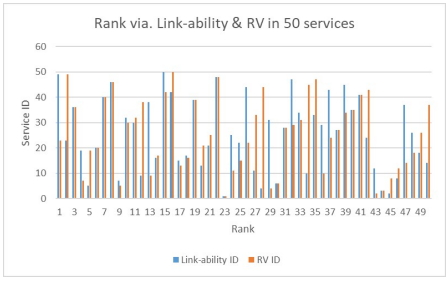


Fig. 5. Experimental procedure.

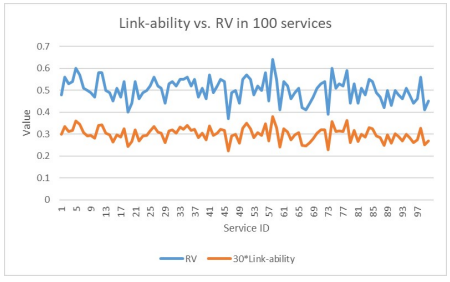
Dataset & Environment: We have referenced service datasets from WS-Dream and QWS, both of them provides reliable distributed assessment mechanism for Web services. To assure the completeness of the experiment, we stimulate a consecutive time-stream with a specific length of time-slot. The simulation of invocation records and LGA are operated on MATLAB R2016b. The cross-reference, data analysis and figures generation are conducted with the help of MATLAB R2016b and Microsoft Excel 2016. All experiments were conducted on a machine with Intel Dual Core i7 3.6GHz CPU, 16G RAM, running Windows 10×64 Professional.



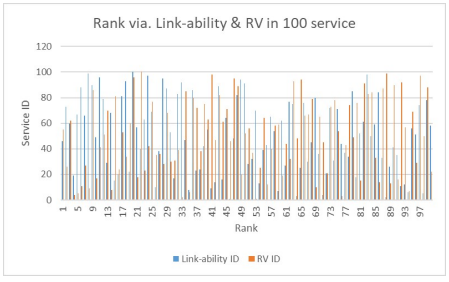
(a) Link-ability vs. RV in 50 services.



(b) Rank deviation in 50 services.



(c) Link-ability vs. RV in 100 services.



(d) Rank deviation in 100 services.

Fig. 6. Link-ability vs. Reference Value.

4.2 Experimental Design & Feature Analysis

4.2.1 Conceptional distinction: Reference value vs. link-ability

Link-ability doesn't simply count the number of successful connections in a time-slot, moreover, it also reflects the potential of connecting more services in the next pick. To prove that, we propose a parameter called Reference Value to be the competitive index against Link-ability. Then, we stimulate the invocation record in different scales of the service pool, which contains 50 and 100 services.

Reference Value (*RV*): We define a service's Reference Value (*RV*), as a proportion of services that connect successfully to this specific service in a time-slot.

$$RV_i \triangleq r_i = \frac{N_{success_i}}{N_{TotalService}} \tag{6}$$

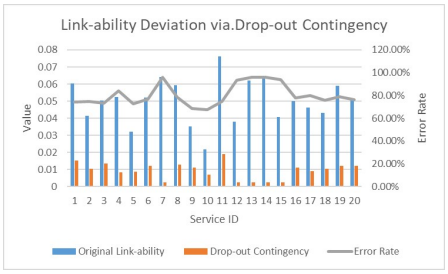
$N_{success_i}$ presents a specific number of services that have successfully connected to *service i* in a time slot. One thing to be stressed here that $N_{success_i}$ doesn't count repeat connections with the same service, which means, r_i is totally different from typical $SuccessRate$. As we defined r_i, r represents the distribution of every service in the service pool in a specific time-slot.

Unlike *RV* simply recording connections with adjacent services, Link-ability covers all the compatibility information of every service in the service pool connected in multi-level by tracing connections. The Link-ability rankings are much more reliable to be the reference for service selection.

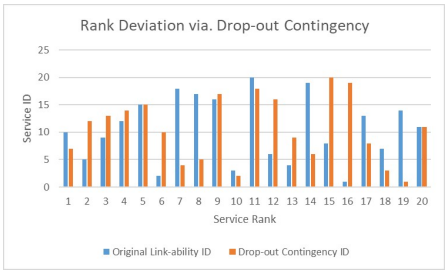
As mentioned above, we stimulate and monitor the invocation of 50 and 100 services in a specific time-slot. Both Link-ability value and Reference Value are generated according to the invocation record. In order to make more clarity to the comparison, Link-ability value is amplified by 30 times. The amplification doesn't interfere the result of observation because absolute deviation doesn't impact rankings among services, only relative deviation influence the choice of service requesters. Shown in Figs. 6 (a) and (c), although both *RV* and Link-ability value share a similar outline of distribution, internal rankings via Link-ability value reveal huge difference from *RV*. Figs. 6 (b) and (d) display the internal rankings among both *RV* and Link-ability value. In a size of 50 services, 30% services share same ranks within *RV* and Link-ability value. When the size of the pool grows up, the percentage quickly falls because of the growing complexity of the topology of the pool. In principle, the *RV* simply count the connections built with specific target service, it doesn't reflect the compatibility of involved services. Unlike *RV*, the Link-ability records not only the compatibility of adjacent service connected, but also the adjacent services' connections, which ends until the whole topology is considered.

4.2.2 Contingencies against drop-out services

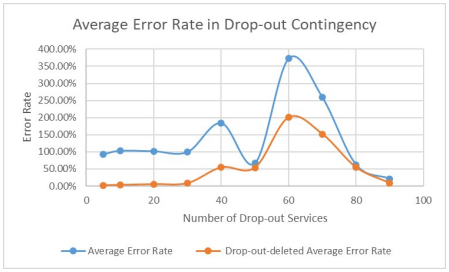
The second series of experiment mainly focus on contingencies against Drop-out problems. What impact will Drop-out problem cast on the link-ability of these services? How much accuracy will the updated mechanism lift from the original mechanism? What are the variation tendencies of Link-ability and its relevant variables when more service participate the drop-out move? Based on these purpose, we architect several experiment to fulfill our hypothesis.



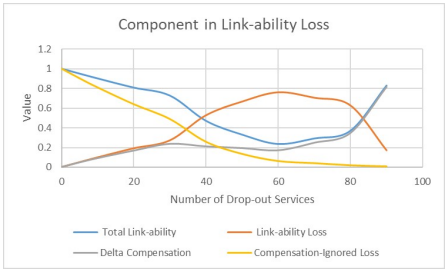
(a) Deviation in drop-out contingency.



(b) Rank deviation in drop-out contingency.



(c) Error rate analysis.



(d) Component analysis in link-ability.

Fig. 7. Link-ability & drop-out contingencies.

In order to reveal the basic impact brought by drop-out services, we first demonstrate a standard simulation with 20 services total, and 5 services which drops out (25% drop-out rate). We observe the float of Link-ability value and ranking alterations.

Figs. 7 (a) and (b) present the Link-ability value deviation within 20 services. As the demonstration suggests, after the drop-out, both Link-ability value and ranks face great changes. Every service in the pool are influenced: their Link-ability value falls 80.57% in average and ranks are completely different from the previous evaluation. In the perspective of a service requester, the change of rankings influence the most when selecting services. **In other words, the Drop-out disrupt the rankings of candidate services, which totally change the priority of service selection.**

4.2.3 Observation in variable tendencies

We simulate a growing percentage of drop-out services in a scale of 100 services and observe the variation tendencies and numerical relationship of Link-ability and its relevant variables: Average Error (*AE*), Drop-out-deleted Average Error (*DAE*), Link-ability Loss (*LL*), Delta Compensation (*DC*), and Compensation-Ignored Loss (*CIL*).

Shown in Figs. 7 (c) and (d), with the increase of Drop-out services, the AE and DAE don't float monotonically. The variation tendency is impacted by the topology generated by services invocation records, and connections involved by Drop-out services. The more connections the Drop-out service involved, the more *AE* is generated.In general, as the proportion of Drop-out service increases, *AE* **is greater than** *DAE* all the time. If the total number of service is fixed, **total Link-ability decreases monotonically while the number of drop-out service increases**. Specifically, Compensation-Ignored Loss, Total

Link-ability, and Link-ability Loss follow this numerical relationship:

$$TL = CIL + LL. \tag{7}$$

4.2.4 Performance analysis

How much reduction in Error Rate does the Drop-out Contingency bring? We simulate different proportion of Drop-out services as 5%, 10%, 20%, 30%, 40%, 50%, 60%, 70%, 80%, 90% among 1000 services and repeat the random drop-out for 1000 times. In order to stick to reality, we assume a reverse probability on different propotion of Drop-out occasions.The result is shown in Fig. 8.

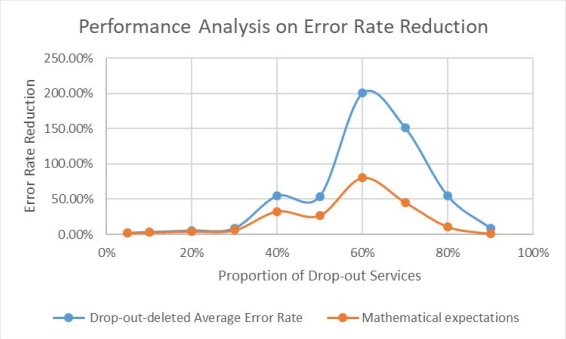


Fig. 8. Performance analysis on error rate reduction.

The Drop-out contingency bring 54.18% reduction in Average Error Rate, 211.54% reduction in Error Rate in Mathematical Expectation.

5. CONCLUSION

We define a parameter 'Link-ability' which is literally dedicated to describe the ability of a service to connect with other services. Starting with proposing a Topology-Retrievable SOA, this paper gives a detailed description on working procedure within the new architecture. Then we put forward a series of mathematical strategy to generate Link-ability and upgrade the mechanism with contingencies to correct calculations when service exit the service pool. Lastly, we perform two series of experiment to strengthen that Link-ability is capable of recording compatibility of the target service by considering all the topology of the pool, and at the same time, can reduce error rate by 54.18% in average when services exit the pool.

REFERENCES

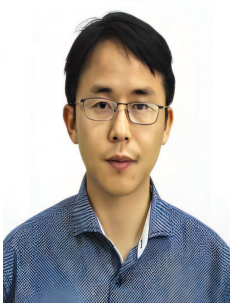
1. S. Berrani, A. Yachir, S. Mahmoudi, B. Djamaa, and M. Aissani, "Towards a new semantic model for service-based iot applications," *Journal of Information Science and Engineering*, Vol. 38, 2022, pp. 38-39.

2. H. Xiao, C. Xu, Z. Feng, R. Ding, S. Yang, L. Zhong, J. Liang, and G.-M. Muntean, "A transcoding-enabled 360° vr video caching and delivery framework for edge-enhanced next-generation wireless networks," *IEEE Journal on Selected Areas in Communications*, Vol. 40, 2022, pp. 1615-1631.
3. T. Peng, H. Wang, C. Liang, P. Dong, Y. Wei, J. Yu, and L. Zhang, "Value-aware cache replacement in edge networks for internet of things," *Transactions on Emerging Telecommunications Technologies*, Vol. 32, 2021, p. e4261.
4. Y. Shen, T. Zhang, Y. Wang, H. Wang, and X. Jiang, "Microthings: A generic IoT architecture for flexible data aggregation and scalable service cooperation," *IEEE Communications Magazine*, Vol. 55, 2017, pp. 86-93.
5. J. Thomas, M. Thomas, and G. Ghinea, "Modeling of web services flow," in *Proceedings of IEEE International Conference on E-Commerce*, 2003, pp. 391-398.
6. P. P. Ray, "A survey on internet of things architectures," *Journal of King Saud University-Computer and Information Sciences*, Vol. 30, 2018, pp. 291-319.
7. Y. A. Ridhawi and A. Karmouch, "Decentralized plan-free semantic-based service composition in mobile networks," *IEEE Transactions on Services Computing*, Vol. 8, 2015, pp. 17-31.
8. L. Yao, Q. Z. Sheng, A. Segev, and J. Yu, "Recommending web services via combining collaborative filtering with content-based features," in *Proceedings of IEEE 20th International Conference on Web Services*, 2013, pp. 42-49.
9. M. Frank, D. Drikakis, and V. Charissis, "Machine-learning methods for computational science and engineering," *Computation*, Vol. 8, 2020, p. 15.
10. C. Wu, W. Qiu, X. Wang, Z. Zheng, and X. Yang, "Time-aware and sparsity-tolerant QoS prediction based on collaborative filtering," in *Proceedings of IEEE International Conference on Web Services*, 2016, pp. 637-640.
11. X. Zhu, X.-Y. Jing, D. Wu, Z. He, J. Cao, D. Yue, and L. Wang, "Similarity-maintaining privacy preservation and location-aware low-rank matrix factorization for qos prediction based web service recommendation," *IEEE Transactions on Services Computing*, Vol. 14, 2021, pp. 889-902.
12. L. Shao, J. Zhang, Y. Wei, J. Zhao, B. Xie, and H. Mei, "Personalized qos prediction forweb services via collaborative filtering," in *Proceedings of IEEE International Conference on Web Services*, 2007, pp. 439-446.
13. Z. Zheng, H. Ma, M. R. Lyu, and I. King, "WSRec: A collaborative filtering based web service recommender system," in *Proceedings of IEEE International Conference on Web Services*, 2009, pp. 437-444.
14. Y. Jiang, J. Liu, M. Tang, and X. Liu, "An effective web service recommendation method based on personalized collaborative filtering," in *Proceedings of IEEE International Conference on Web Services*, 2011, pp. 211-218.
15. L. Ding, G. Kang, J. Liu, Y. Xiao, and B. Cao, "Qos prediction for web services via combining multi-component graph convolutional collaborative filtering and deep factorization machine," in *Proceedings of IEEE International Conference on Web Services*, 2021, pp. 551-559.
16. Y. Xiao, G. Kang, J. Liu, B. Cao, and L. Ding, "WSGCN4SLP: Weighted signed graph convolutional network for service link prediction," in *Proceedings of IEEE International Conference on Web Services*, 2021, pp. 135-144.

17. X. Chen, X. Liu, Z. Huang, and H. Sun, "RegionKNN: A scalable hybrid collaborative filtering algorithm for personalized web service recommendation," in *Proceedings of IEEE International Conference on Web Services*, 2010, pp. 9-16.
18. W. Lo, J. Yin, S. Deng, Y. Li, and Z. Wu, "Collaborative web service qos prediction with location-based regularization," in *Proceedings of IEEE 19th International Conference on Web Services*, 2012, pp. 464-471.
19. C.-W. Lin, "Poster: Formal qos compatibility verification for components on time-sensitive networking," in *Proceedings of IEEE Vehicular Networking Conference*, 2018, pp. 1-2.
20. A. Pastukh, E. Deviatkin, V. Tikhvinskiy, and A. Kulakaeva, "Compatibility studies between 5G IoT networks and fixed service in the 6425-7125 mhz band," in *Proceedings of International Conference on Engineering Management of Communication and Technology*, 2021, pp. 1-4.
21. Y. Jin, W. Guo, and Y. Zhang, "A time-aware dynamic service quality prediction approach for services," *Tsinghua Science and Technology*, Vol. 25, 2020, pp. 227-238.
22. Y. Lee, "QoS management for soa by synchronizing quality context in UDDI," in *Proceedings of the 2nd International Conference on Future Generation Communication and Networking Symposia*, Vol. 1, 2008, pp. 17-22.
23. G. Saez, A. Sliva, and M. Blake, "Web services-based data management: evaluating the performance of UDDI registries," in *Proceedings of IEEE International Conference on Web Services*, 2004, pp. 830-831.
24. W. Tong, X. Dong, Y. Shen, and X. Jiang, "A hierarchical sharding protocol for multi-domain iot blockchains," in *Proceedings of IEEE International Conference on Communications*, 2019, pp. 1-6.
25. D. A. Alfenas, D. P. Shibata, J. Jose, and M. R. P. Barretto, "Adaptive markov systems: Formulation and framework," *IEEE Latin America Transactions*, Vol. 12, 2014, pp. 1271-1277.
26. S. Dong, Z. Wu, Y. Pan, H. Su, and Y. Liu, "Hidden-markov-model-based asynchronous filter design of nonlinear markov jump systems in continuous-time domain," *IEEE Transactions on Cybernetics*, Vol. 49, 2019, pp. 2294-2304.
27. M. Kaouan, D. Bouchiha, S. M. Benslimane, and S. Boukli-Hacene, "Towards service ontology for web services storage and discovery," in *Proceedings of the 4th International Symposium on Informatics and its Applications*, 2020, pp. 1-6.
28. K. Tamilarasi and M. Ramakrishnan, "Design of an intelligent search engine-based UDDI for web service discovery," in *Proceedings of International Conference on Recent Trends in Information Technology*, 2012, pp. 520-525.
29. S. Simi and A. Sherin, "A centrality based random walk approach for topology formation in networks," in *Proceedings of International Conference on Wireless Communications, Signal Processing and Networking*, 2017, pp. 1468-1472.
30. C. Xiumei, B. Jingwei, W. Yan, and C. Qiaoqiao, "Semi-supervised image segmentation based on k- means algorithm and random walk," in *Proceedings of IEEE Symposium Series on Computational Intelligence*, 2019, pp. 2853-2856.

Shiyang Ma received the BE degree in Telecommunications Engineering from Xidian University, Xi'an, China, in 2015 and is currently working toward the Ph.D. degree in the School of Computer Science and Technology, Xidian University. His current research interests are semantic technology, service computing, IoT, and blockchain.

Yulong Shen received the BS and MS degrees in Computer Science and Ph.D. degree in Cryptography from Xidian University, Xi'an, China, in 2002, 2005, and 2008, respectively. He is currently a Professor at the School of Computer Science and Technology, Xidian University, China. He is also an Associate Director of the Shaanxi Key Laboratory of Network and System Security and a member of the State Key Laboratory of Integrated Services networks Xidian University, China. He has also served on the technical program committees of several international conferences, including ICEBE, INCoS, CIS and SOWN. His research interests include wireless network security and cloud computing security.



Xuewen Dong received the BE, MS and Ph.D. degrees in Computer Science and Technology from Xidian University, Xi'an, China, in 2003, 2006 and 2011, respectively. From 2016 to 2017, he was a Visiting Scholar with Oklahoma State University, OK, USA. He is currently a Professor with the School of Computer Science, Xidian University. His research interests include cognitive radio network, blockchain, wireless network security and privacy.

Wei Tong received the Ph.D. degree in Cyberspace Security from Xidian University, China, in 2022. He is currently a Joint Post-Doctoral of the Hangzhou Institute of Technology and the School of Computer Science and Technology, Xidian University. His current research interest is blockchain technology and application.

Lingxiao Yang received the B.E. degree in Network Engineering from Xidian University in 2018. He is a member of the Shaanxi Key Laboratory of Network and System Security. He is currently pursuing the Ph.D. degree at the School of Computer Science and Technology, Xidian University, China. His research interests include blockchain and machine learning.