THE COST-DRIVEN SERVICES COMPOSITION FOR ENTERPRISE WORKFLOWS IN UNCERTAIN ENVIRONMENT

Honghao Gao[1,2], Qiming Zou[1], Yucong Duan[3], Yuyu Yin[4]
1. Computing Center, Shanghai University, 200444 Shanghai, P.R. China
2. School of Computer Engineering and Science, Shanghai University, 200444 Shanghai, P.R. China
3. College of Information Science and Technology, Hainan University, 570100 Haikou, P.R. China
4. School of Computer, Hangzhou Dianzi University, 310018 Hangzhou, P.R. China
Email: gaohonghao@shu.edu.cn

Abstract

The uncertain feature of Internet imposes high risks on the reliability of service workflow. The risk of failure caused by unavailable services may increase costs when using service workflow based applications. Thus, it is necessary to consider the non-functional factors of service cost and reliability. In this paper, we propose a cost-driven services composition approach for enterprise workflows, where formal verification is employed to recommend appropriate services to abstract workflow. The primary work checks services composition in a quantitative way to make ensure the solution of service configuration to service workflow has the best performance with high reliability and low cost. First, it introduces the service searching approach based on inverted index, and the service recommendation method based on improved Pearson formula. After that, it will return a set of candidate services for instancing workflow. Second, service model and workflow model are defined to formalize the behavior of services composition, which is considered as verification model. Then, the quantitative verification checks each possible plan of services composition by using probabilistic model checking. Our approach provides an effective method to generate the optimal service workflow.

Keywords: Service Workflow, Probabilistic Model Checking, Uncertain Environment, Services Searching and Recommendation, Formal Verifications.

1. INTRODUCTION

Service-Oriented Computing (SOC) has been widely used in modern industry and academia, which has an ability to quickly develop information system using Web services to integrate distributed applications (Papazoglou, 2008; Papazoglou, 2003). However, single service is hard to satisfy complex requirements of business logics, which calls for services composition. As one of possible technologies, service workflow is a flexible way to address this problem. Each activity of business process in abstract workflow can be mapped with services to provide value-added functions to achieve target requirements. Due to the diversity and complexity of services composition, it involves service searching, service selection, services verification to compose multiple individual services together (Tan, 2010; Jiang, 2012).

The open, dynamic and ever-changing features of Internet will lead to workflow failures. The traditional workflow method is stationary and certain, which will be broken down due to the risk of failure may be randomly encountered under this uncertain environment (Gao, 2013). If enterprise workflow has no response mechanism to dynamically configure workflow with new services, the running of service workflow will cause serious performance issues.

The process of service selection selects and composes services into new composite service to support abstract business process of workflow. However, most existing studies (Changyuan, 2010; Aalst, 1999; Zhang, 2011) do not account for risk-aware services composition for workflow. These techniques only require services composition to be consistent with functional behavior claims. In the practical running environment, the risk of service failure shows a probability of service disruptions. Therefore, it is possible that the new recommended plan of services composition will be encountered with failures again. It has several limitations and may cause users inconveniences. To this point, service workflow implemented by different services composition plans contains different risks because component service shows different reliability. Therefore, the performance of service workflow is not only dependent on service function but correlated to the quality of service. To minimize the re-failure risks of new service configuration, we need to select the optimal service workflow with the lowest cost and highest reliability. To this point, the balance between cost and reliability should be considered, which may have significant impact on the efficiency of service workflow.

As core issue, we transform the problem of service composition into the process of applying probabilistic model checking. Each plan is checked in a quantitative way, by which verification results with cost and reliability help us to match better services. The main contributions of this paper are summarized as follows:

1) The service searching method and service recommendation method are used to improve the efficiency

DOI: 10.29268/iciot.2016.0008

of building the candidate service set. These methods can avoid the searching works to be overloaded and degraded.

2) Due to the candidate service set may give different plans of services configurations to abstract workflow, the formal verification is used to check each plan in a quantitative way focusing on evaluating functional and non-functional requirements, such as interface operation, cost and reliability.

The rest of this paper is organized as follows. Section 2 shows the candidate service set generation process. Section 3 introduces formal models and verification process for checking service process. Section 4 reviews related works. Section 5 concludes this paper and gives future research directions.

2. THE SERVICE SELECTION FOR SERVICES COMPOSITION

In this section, formal models are proposed to formalize Web service and abstract workflow. The service searching problems and service recommendation are discussed to generate the service candidate set.

2.1 Formal Models

Definition 1 (Web Service). Web service is defined as tuple ws::=(id,F,I,O), where,

1) id is the identifier for each Web service.

2) F is the functional description for each Web service.

3) $I=\{i_1,i_2,\ldots,i_n\}$ is a set of inputs, which are received from invoker.

3) $O=\{o_1,o_2,\ldots,o_m\}$ is a set of outputs, which are sent back to invoker.

Web service receives inputs I from invoker and sends outputs O to invoker via interface operations. In general, the inputs and outputs are basic information described in WSDL files (Crasso, 2010, Wu, 2009). The functional description F shows the service functionality, which can be considered as the source of function matching.

Definition 2 (Web Service Performance). To formalize non-functional behaviors, the cost and reliability of Web service ws are defined as follows.

1) The reliability of Web service is denoted as $R(ws)\in[0,1]$, which represents the probability of reliability running when a request is sent to invoke service ws.

2) The cost of Web service is denoted as as $C(ws)\in\mathbb{R}$, which shows how much should be paid when user agrees to use service ws.

In uncertain environment, Web service may be unavailable, which will make the software encountered with unpredictable risks. Definition 2 indicates that each service displays both functional and non-functional behaviors. During service invocations, user should prepare for risks of service failure. Thus, many service agreements often predefine different kinds of service costs under different reliability conditions, such as SLA (Service-Level Agreement).

Definition 3 (Business Workflow Model). Business workflow describes business logics in abstract way. It is defined as tuple BWM::=(N,C,T,s,e), where,

1) N is a set of logic tasks, which can be mapped with different services.

2) C is a set of control conditions for workflow, e.g., execution probabilities of branch and loops structures.

3) $T\subseteq(C\times N)\cup(N\times N)\cup(N\times C)$ is a set of transitions, which is divided into three types that $C\times N$, $N\times N$, and $N\times C$.

4) $s\in N$ is the starting node of logic task, and $e\in N$ is the ending node of logic task.

Definition 4 (Workflow Requirements). Workflow includes two requirement types. One is the functional requirement, which is used to the service matching. The other is non-functional requirement, which is used to the performance evaluation.

1) The functional requirement $\mathcal{I}(n)=I^w$ is in the form of inputs for each logic task $n\in N$. The functional requirement $\mathcal{O}(n)=O^w$ is in the form of outputs for each logic task $n\in N$.

During comparing services, the candidate service should have similar functions with same inputs and outputs. For Web service ws, if it is satisfied to functional requirement of logic node n of workflow, then it should satisfy that $ws.I\subseteq\mathcal{I}(n)\wedge ws.O\supseteq\mathcal{O}(n)$.

2) The non-functional requirement consists of local requirement and global requirement. The local requirement focuses on logic task that $\mathcal{R}(n)$ and $\mathcal{C}(n)$. The global requirement focuses on workflow that $\mathcal{R}$ and $\mathcal{C}$.

For Web service ws, if it is satisfied to non-functional requirement of logic node n in workflow, then it should satisfy that $R(ws)\geq\mathcal{R}(n)\wedge C(ws)\leq\mathcal{C}(n)$. For checking $\mathcal{R}$ and $\mathcal{C}$, the method in literature (Hwang, 2007) is used to compute the total cost and reliability of service workflow.

2.2 Problem Definitions and Solution

Given a set of interfaces operations with inputs and outputs, the matching work of service searching is to compare each service from service repository. The complexity is O(N*M) in the worst, where N is the number of logic task nodes of workflow and M is the service number in service repository. It will be a huge task when the number of workflow nodes and service number are largely increased. Thus, we use the inverted index method to build service index in order to improve the searching efficiency.

Definition 5 (The Service Index). The inverted index for service is defined as tuple SI::=(k,S,f) where,

1) k is the set of keywords, in which inputs and outputs of interface operations are used as service identifications.

2) S is a set of services, which is related to id of Web service.

3) $f(k) \subseteq 2^S$ defines that services with same keyword k can be grouped into a set.

The service index includes a set of interfaces labeling with inputs or outputs. Due to service index is the one-to-many pattern, each index is mapped with different services to show that they can support at least one same interface.

Table 1. An Example of Service Index

K	*S*				
I1	S1	S3	S4	S6	
I2	S3	S4	S7	S10	S11
I3	S1	S5			
O1	S1	S4	S7	S1	
O2	S1	S2	S3	S5	S7
O3	S3	S4	S5	S11	

The service searching aims to find a set of candidate services according to interface operation. This service index method improves the traditional service searching methods because it can quickly return target services. For example, there are 6 indexes {I1,I2,I3,O1,O2,O3} in Table 1. If we want to search services which contains Output O2, it will return {S1,S2,S3,S5,S7} as candidate services without comparing other services.

Definition 6 (The Functional Consistency). Let $CS=\{cs_1,cs_2,\ldots,cs_n\}$ be the candidate service set, and $cs_i \in CS$ be mapped to $n_j \in N$. The functional consistency requires

$$\forall n \in N\ (\forall i \in I(n),\ \exists cs \in CS\ \exists i' \in cs.I\ \wedge\ i=i')$$
$$\wedge$$
$$\forall n \in N\ (\forall o \in O(n),\ \exists cs \in CS\ \exists o' \in cs.O\ \wedge\ o=o') \quad (1)$$

The functional consistency indicates that each input or output of task node of workflow should be simulated by corresponding service in the candidate service set. However, there will be a large number of candidate services, which calls for refining the current set to return suitable services.

Definition 7 (The Refinement of Candidate Service Set). Suppose the target interface set is $\{op_1,op_2,\ldots,op_n\}$. The refinement process of candidate service set is divided into two steps.

1) The intersection function is used to build an initial set

$$CS = \bigcap_{i=1}^{n} f(op_i) \quad (2)$$

This function removes redundant services. For example, let {I1,I2,O3} be target interfaces. From table 1, there are three sets that f(I1)={(I1,S1), (I1,S3), (I1,S4), (I1,S6)}, f(I2)={(I2,S3), (I2,S4), (I2,S7), (I2,S10), (I2,S11)}, f(O3)={(O3,S3), (O3,S4), (O3,S5), (O3,S11)}. We can obtain the candidate service set CS={S3,S4}.

2) The multi-objective selection function is used to delete services which cannot satisfy the non-functional requirements.

$$CS = CS \setminus \{ws \in CS \mid R(ws) < \delta_R(n) \vee C(ws) > \delta_C(n)\} \quad (3)$$

With the help of multi-objective selection function, the candidate service set can be refined by eliminating services whose has low reliability or high cost. Thus, it can guarantee the high feasibility of services composition.

To improve searching efficiency, the other way is the service recommendation which is different from general service searching. In some cases, dynamic services composition should be completed in seconds. Thus, recommending related services to workflow based on the user usage records can save the service searching time. In the following, we will discuss the improved service recommendation method to select services to support business process of abstract workflow.

Definition 8 (The Service Recommendation). Suppose the usage frequency of services in service repository will be recorded and updated in real-time. The service recommendation method based on Pearson formula focuses on these usage records. It includes following steps.

1) The usage frequency of service is in the form of vector. For user us_i, the historical record is $r_{ws1,usi}$ $r_{ws2,usi}$ .... $r_{ws_{n-1},usi}$ $r_{ws_n,usi}$ where $r_{wsj,usi}$ shows the usage frequency of service ws_i used by user us_j.

2) After calculating all users' historical records, a two-dimensional matrix for users and services is generated as follows,

$$\begin{array}{c} \\ us_1 \\ us_2 \\ \cdots \\ us_{m-1} \\ us_m \end{array}
\begin{array}{c} \begin{array}{ccccc} ws_1 & ws_2 & & ws_{n-1} & ws_n \end{array} \\
\begin{bmatrix} r_{ws1,us1} & r_{ws2,us1} & \cdot & r_{ws_{n-1},us1} & r_{ws_n,us1} \\ r_{ws1,us2} & r_{ws2,us2} & \cdot & r_{ws_{n-1},us2} & r_{ws_n,us2} \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ r_{ws1,us_{m-1}} & r_{ws2,us_{m-1}} & & r_{ws_{n-1},us_{m-1}} & r_{ws_n,us_{m-1}} \\ r_{ws1,us_m} & r_{ws2,us_m} & & r_{ws_{n-1},us_m} & r_{ws_n,us_m} \end{bmatrix} \end{array} \quad (3)$$

3) The user-service matrix is changed into the service-service matrix according to the Pearson formula (Benesty, 2009; Jiang, 2011) that,

$$v_{ws,ws'} = \frac{\sum_{i=1}^{n}(r_{ws,us_i} - \bar{R}_{ws})(r_{ws',us_i} - \bar{R}_{ws'})}{\sqrt{\sum_{i=1}^{n}(r_{ws,us_i} - \bar{R}_{ws})^2 \sum_{i=1}^{n}(r_{ws',us_i} - \bar{R}_{ws'})^2}} \quad (4)$$

The $v_{ws,ws'}$ is a correlation value for services ws and ws'. The r_{ws,us_i} is a count value for service ws used by user us_i. $\bar{R}_{ws}$ is the average count value of user ws. The r_{ws',us_i} is a count value for service ws' used by user us_i. $\bar{R}_{ws'}$ is the average count value of user ws'.

However, it should consider interface operations since the incompatibility will make service unavailable. Obviously, the correlation value $v_{ws,ws'}$ is not accurate to decide whether two users are similar or not, especially they are incompatible with each other. To this point, the $v_{ws,ws'}$ should be improved to enhance the accuracy of user similarity based on the interface operation, cost, and reliability, which aims to enable Pearson formula to be suitable to recommend related services.

$$v'_{ws,ws'} = \left(\lambda \times \frac{|ws'.I| + |ws'.O|}{\underset{n\in N}{Max}\{|\sum I^w(n)| + |\sum O^w(n)|\}} + \varphi \times \frac{c(ws')}{\underset{n\in N}{Min}\{C^w(n)\}} + \phi \times \frac{R(ws')}{\underset{n\in N}{Max}\{R^w(n)\}}\right) \times v_{ws,ws'} \quad (5)$$

In formula (5), we consider the interface similarity $\frac{|ws'.I| + |ws'.O|}{\underset{n\in N}{Max}\{|\sum I^w(n)| + |\sum O^w(n)|\}}$, the cost similarity $\frac{c(ws')}{\underset{n\in N}{Max}\{C^w(n)\}}$, and the reliability similarity $\frac{R(ws')}{\underset{n\in N}{Max}\{R^w(n)\}}$. Parameters λ, φ and ϕ are used to control the weight value during computing the correlation value $v_{ws,ws'}$, where $\lambda+\varphi+\phi=1$. If cost is important, then φ can be greater than ϕ and λ. If reliability is important, then ϕ can be greater than φ and λ. If interface is important, then λ can be greater than φ and ϕ.

4) Finally, the new correlation matrix is generated as follow,

$$\begin{array}{c c} & \begin{array}{ccccc} ws_1 & ws_2 & & ws_{n-1} & ws_n \end{array} \\ \begin{array}{c} ws_1 \\ ws_2 \\ \dots \\ ws_{n-1} \\ ws_n \end{array} & \begin{bmatrix} v_{1,1} & v_{1,2} & \cdot & v_{1,n-1} & v_{1,n} \\ v_{2,1} & v_{2,2} & \cdot & v_{2,n-1} & v_{2,n} \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ v_{n-1,1} & v_{n-1,2} & & v_{n-1,n-1} & v_{n-1,n} \\ v_{n,1} & v_{n,2} & & v_{n,n-1} & v_{n,n} \end{bmatrix} \end{array} \quad (6)$$

For each service, the correlation value can be ranked from low to high ratings for service recommendation. However, according to improved Pearson formula, there will be huge number of candidate services. Note that the IPS(ws) function will output related top-k services as a recommendation set RS.

$$RS = \bigcup_{i=1}^{k} IPS(ws) \setminus \{ws \in CS \mid R(ws) \geq R^w(n) \wedge C(ws) \leq C^w(n)\} \quad (7)$$

Thus, the formula (7) is used to eliminate services whose cannot satisfy the non-functional requirements, especially reliability and cost.

Definition 9 (The Minimal Recommendation Service Set). In order to give the minimal set, it is reduced to following formula.

$$RS^{Min} = \{RS' \subseteq 2^{RS} \mid Min(|\sum_{rs_i\in RS'} rs_i.I| - |\sum_{n_j\in N} I^w(n_j)|, |\sum_{rs_i\in RS'} rs_i.O| - |\sum_{n_j\in N} O^w(n_j)|)\} \quad (8)$$

In the worst condition, candidate services from the recommendation service set will lead to the state space exploration problem when all logic nodes of workflow have at most k component services. The minimal recommendation service set returns a subset RS^{min} which covers inputs and outputs of interface requirements. In this process, we use the greedy algorithm to get the minimal recommendation service set.

3. APPLYING PROBABILISTIC MODEL CHECKING TO SERVICES COMPOSITION

The challenge of services composition for workflow is the solution evaluation. To this problem, we apply probabilistic model checking to services composition in order to evaluate each plan of service configurations.

3.1 Verification Process Overview

According to service plan, candidate service will be mapped to each task to obtain the target service workflow. The probabilistic model checking employed to verify service workflow includes three steps, mainly formal modeling, verification property generation, and verification execution. The verification result contains not only the satisfiability assertion between model and property but also the quantitative information about cost and reliability.

Step 1) The behavioral model is generated in the form of formal model considering temporal relations of service invocation.

Step 2) The requirements of workflow is changed into verification property in the form of temporal logic formulae.

Step 3) Model checker tool PRISM is used to perform automatic verification when formal model and verification property are coded into PRISM input language.

Step 4) Verification results are analyzed to confirm the optimal plan.

3.2 Formal Verification Model

Definition 9 (The Formal Model of Services Composition) The verification model is a labeled transition system. It is defined as tuple FWS::=(S, s0, [illegible], P, C, AP, L), that is,

- S is a finite set of states, where each state corresponds to a Web service.

- s0 is an initial state.
- $\delta(S) \subseteq S$ is a finite set of edge relations, which shows the service invocation relation.
- $P(\cdot,\cdot) \rightarrow [0,1]$ is the transition probability matrix, where $\sum_{s' \in S} P(s,s') = 1$ for all $s \in S$.
- $C(S) \rightarrow N$ is the cost function, which shows service cost.
- AP is a finite set of atomic propositions.
- L: $S \rightarrow 2^{AP}$ is labeling function that assigns a set of atomic propositions to each state $s \in S$.

For state s, $s' \in S$, the transition probability matrix gives the probability P(s,s') of making a transition from s to s' in one discrete step. The service cost is specified by the function C(S), which gives each state cost in S. Each transition is annotated with a probability value indicating the likelihood of its occurrence during possible service invocations. The transition probability matrix denotes a set of all probabilistic distributions over the state space.

3.3 Model Checker PRISM and Its Language

PRSIM (Kwiatkowska, 2009) is a probabilistic model checker, which supports various model types including of Discrete-Time Markov Chain (DTMC), Markov Decision Process (MDP), Continuous-Time Markov Chain (CTMC) and Probabilistic Timed Automat (PTA). In this paper, we mainly use DTMC to code services composition model. The DTMC syntax of PRISM is in the form of action guard command which forces two or more modules to make transitions simultaneously with different probabilities.

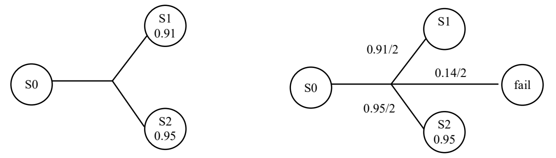
[] <location> → <prob>: <location'>+
…+<prob>: <location'>

where location is an label corresponding to states of FWS model, prob $\in$ [0,1] is the probability function after changing current location to new location'.

To extend the probability for each transition, the reliability of mapped service is used to generate the transition probability. The transition probability computing is worked as follows.

$$p(s,s') = \frac{r(s')}{|\{(s,s'') \mid (s,s'') \in \delta\}|} \quad (8)$$

For example, in Figure 1(a), suppose state s0 has two ongoing transitions (s0, s1) and (s0,s2). According to formula (8), the new transition probability for (s0,s1) is 0.91/2=0.405 and (s0,s2) is 0.95/2=0.425. The special state fail denotes the failure probability when states s1 and s2 are unavailable.



(a) (b)

Figure 1. The Transition Probability Example

PRISM supports the rewards structure (or, equivalently, costs), which assigns real values to states. The reward expression is as follows.

```
rewards "name"
   exp_1
   …..
   exp_n
endrewards
```

where each exp_i item describes an expression in the form of <guard>: <reward> that guard is a condition and reward is a cost.

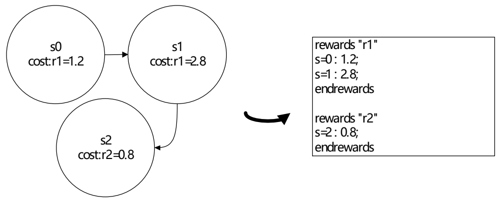


Figure 2. The Reward Example of Service Cost

For example, in Figure 2, states s=0, s=1, and s=2 have been denoted with cost. However, they are belonged to different cost definition. Thus, we define two reward expressions, mainly cost r1 and cost r2. Under cost r1, if state s=0 is visited, then the cost r1 is added with 1.2. If state s=1 is visited, then the cost r1 is added with 2.8. Under cost r2, if state s=2 is visited, then the cost r2 is added with 0.8.

4. RELATED WORKS

Web service has been changing traditional business models, which helps modern enterprises to seize business opportunities and accelerate inter-enterprise collaborations. A lot of researches on services composition have been published from different perspectives. We give a review on major techniques and methods that are most closely related to our works.

Many researches focus on service selection. Xianzhi Wang et.al., (Wang, 2016,) proposed algorithms to QoS-aware service selection based on the artificial bee colony algorithm (ABC). Yue Wang et.al., (Wang, 2016) proposed a skyline-based Web service selection method to deal with the efficiency problem as well as solved the frequent requests problem. To improve efficiency of service selection, the Kd-tree based search algorithm was designed to determine skyline and then reduce the search space. Puwei Wang et.al., (Wang, 2016) proposed a type of incentive contract that motivated the service providers to offer the QoS and prices. An incentive mechanism for effective service selection was proposed to fulfill the global QoS requirements. Dionisis Margaris et.al., (Margaris, 2015) considered a number of service selection issues related to WS-BPEL scenario adaptation, aiming to enhance adaptation quality and improve the QoS offered to end users. However, none of existing methods take into consideration service searching and service recommendation to improve the efficiency of service selection.

Many researches focus on service QoS. ZhiJun Ding et.al., (Ding, 2015) addressed the issue of selecting and composing services via genetic algorithm, and proposed a transaction and QoS-aware selection approach. Ikbel Guidara et.al., (Guidara, 2015) presented a heuristic based time-aware service selection approach to efficiently select a close-to-optimal combination of services. Liangzhao Zeng et.al., (Zeng, 2004) presented a middleware platform which addressed the services selection issue for the purpose of composition in a way. It aimed to maximize user satisfaction expressed as utility functions over QoS attributes. Tao Yu et.al., (Yu, 2007) designed a broker-based architecture to facilitate the selection of QoS-based services. The objective of service selection was to maximize an application-specific utility function under the end-to-end QoS constraints. However, these methods are hard to handle dynamic services composition because service selection requires human intervention to make decisions during QoS computing. Moreover, QoS is a collection of multidimensional indexes that how to evaluate service composition should translate QoS values into the unified evaluation index.

Different from above researches, we propose a two phase services composition for abstract workflow. In the first phase, the service selection is archived by service searching and service recommendation. In the second phase, probabilistic model checking is employed to check which plan of service configurations for workflow is optimal with low cost and high reliability.

5. CONCLUSION AND FUTURE WORKS

The service workflow as one of enterprise information integration methods has been widely used in E-commerce and scientific computing. However, in uncertain environment, workflows have risks of failure, which will make component service unavailable. In order to guarantee the service workflow satisfies the functional and non-functional requirements, it is necessary to research the cost-driven services composition. The aim of this paper is to select services for abstract workflow by using probabilistic model checking to verify service plan in a quantitative way.

First, the inverted index method is used to generate service index in order to improve the efficiency of service searching. Then, the service searching selects services according to interface operation of user functional requirements. Furthermore, the refined candidate service set will be returned to abstract workflow for service configuration. Third, the usage frequency is used to generate service-service correlation matrix by using improved Pearson formula, which considers interface, cost, and reliability factors to recommend correlated services. At last, model checker PRISM is employed to perform the formal verification by which the verification result helps to confirm which plan of service configuration is suitable to current workflow.

The paper only discusses the cost-driven services composition in uncertain environment. However, the service invocation has time limitations in practices (Xiang, 2015). Thus, we will consider the time constraint to extend the services composition verification in the future.

6. ACKNOWLEDGEMENT

This paper is supported by National Natural Science Foundation of China under Grant No.61502294, 61363007, Natural Science Foundation of Shanghai under Grant No.15ZR1415200, CERNET Innovation Project under Grant No.NGII20150609, and Foundation of Science and Technology Commission of Shanghai Municipality under Grant No.14590500500.

7. REFERENCES

Michael P. Papazoglou, Paolo Traverso, Schahram Dustdar, Frank Leymann(2008). Service-oriented computing, *International Journal of Cooperative Information Systems*, 17(2), pp.223-255.

MP Papazoglou, D Georgakopoulos (2003). Service-oriented computing, *Communication on ACM*, 46(10), pp.25-28.

Wei Tan, Yushun Fan, MengChu Zhou(2010). Data-driven service composition in enterprise SOA solutions: A Petri Net approach, *IEEE Transactions on Automation Science & Engineering*, 7(3), pp.686-694.

Wei Jiang, Songlin Hu, Dongwon Lee(2012). Continuous query for QoS-aware automatic service composition, *IEEE International Conference on Web Services*, Honolulu, Hawaii, USA, 24-29 June 2012, pp.50-57.

Honghao Gao, Huaikou Miao(2013). A Quantitative Model-Based Selection of Web Service Reconfiguration. *2013 14th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD)*, Honolulu, Hawaii, USA, 1-3 July 2013, pp.365-371.

G Changyuan , Qi Kai (2010). Enterprise Modeling Framework for ERP in Manufacturing Industry. *Chinese Journal of Management*, 7(9), pp.1380-1385.

W. Aalst(1999). Formalization and Verification of Event-driven Process Chains. *Information and Software Technology*, 41(10), pp.639-650.

Weimin Zhang, Cancan Liu, Zhigang Luo (2011). A Review on Scientific Workflows, *Journal of National University of Defense Technology*, 33(3), pp.56-65.

Marco Crasso. Juan Manuel Rodriguez. Alejandro Zunino. Marcelo Campo(2010). Revising WSDL Documents: Why and How, *IEEE Internet Computing*, 14(5), pp.48-56.

Zhaohui Wu , Shuiguang Deng, Ying Li, Jian Wu(2009). Computing compatibility in dynamic service composition, *Knowledge and Information Systems*, 19, pp.107-129.

San-Yih Hwang, Haojun Wang, Jian Tang, Jaideep Srivastava (2007). A probabilistic approach to modeling and estimating the QoS of web-services-based workflows, *Information Sciences*, 177, pp.5484-5503.

J. Benesty, J. Chen, Y. Huang(2009). Pearson Correlation Coefficient, *Noise Reduction in Speech Processing*, 2, pp.1-4.

Yechun Jiang, Jianxun Liu, Mingdong Tang(2011). An Effective Web Service Recommendation Method Based on Personalized Collaborative Filtering, *2011 IEEE International Conference on Web Services*, San Francisco, USA, June 27 - July 2, 2016, pp.211-218.

Marta Kwiatkowska, Gethin Norman and David Parker(2009). PRISM: Probabilistic Model Checking for Performance and Reliability Analysis.*ACM SIGMETRICS Performance Evaluation Review*, 36(4), pp.40-45.

Xianzhi Wang, Xiaofei Xu, Quan Z. Sheng, Zhongjie Wang, Lina Yao(2016). Novel Artificial Bee Colony Algorithms for QoS-Aware Service Selection, *IEEE Transactions on Services Computing*, http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7574305&isnumber=4629387.

Yue Wang, You Song, Mingyang Liang(2016). A skyline-based efficient web service selection method supporting frequent requests, *2016 IEEE 20th International Conference on Computer Supported Cooperative Work in Design*, Nanchang, China, 04-06 May, 2016, pp.328 -333.

Puwei Wang, Xiaoyong Du (2016). QoS-aware Service Selection Using An Incentive Mechanism, *IEEE Transactions on Services Computing*,http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7552591&isnumber=4629387

Dionisis Margaris, Panagiotis Georgiadis, Costas Vassilakis(2015). On Replacement Service Selection in WS-BPEL Scenario Adaptation, *2015 IEEE 8th International Conference on Service-Oriented Computing and Applications*, Rome, Italy, 19 - 21 October, 2015, pp.10-17.

ZhiJun Ding, JunJun Liu, YouQing Sun, ChangJun Jiang, MengChu Zhou (2015). A Transaction and QoS-Aware Service Selection Approach Based on Genetic Algorithm, *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 45(7), pp.1035-1046.

Ikbel Guidara, Nawal Guermouche, Tarak Chaari, Said Tazi, Mohamed Jmaiel(2015). Heuristic Based Time-Aware Service Selection Approach, *2015 IEEE International Conference on Web Services*, New York, NY, USA, Jun 27- Jul 2, 2015, pp.65-72.

Liangzhao Zeng, Benatallah Boualem, Anne H.H. Ngu (2004). QoS-Aware Middleware for Web Services Composition, *IEEE Transactions on Software Engineering*, 30(5), pp.311-327.

Tao Yu, Yue Zhang, Kwei-Jay Lin(2007). Efficient Algorithms for Web Services Selection with End-to-End QoS Constraints, *ACM Transactions on the Web*, 1(1), pp.1-26.

Chaocan Xiang, Panlong Yang, Xuangou Wu, Hong He, Shucheng Xiao(2015). QoS-based service selection with lightweight description for large-scale service-oriented internet of things. *Tsinghua Science and Technology*, 20(4), pp.336-347.

Authors

Honghao Gao, received the Ph.D degree in computer application technology from the School of Computer Engineering and Science of Shanghai University, Shanghai, P. R. China, in 2012. His research interests include Web service and model checking.

Qiming Zou, received the Doctors degree in mechanical automation from Shanghai University. He is currently an assistant professor at Shanghai University. His research interests include System Reliability and Grid-based Manufacturing.

Yuyu Yin, received the Doctors degree in computer science from Zhejiang University, Hangzhou, China, in 2010. He is currently an associate professor at Hangzhou Dianzi University. His research interests include service computing, cloud computing and middleware techniques.

Yucong Duan, received the Ph.D degree in Software Engineering from Institute of Software, Chinese Academy of Sciences, P. R. China in 2006. He is currently a full professor and vice director of Computer Science department in Hainan University, P. R. China. His research interests include software engineering, service computing, cloud computing, and big data