IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received February 26, 2020, accepted March 8, 2020, date of publication March 11, 2020, date of current version March 20, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2979970

A Simulation-Based Optimization Approach for Reliability-Aware Service Composition in Edge Computing

JIWEI HUANG, (Member, IEEE), JINGYU LIANG, AND SIKANDAR ALI
Department of Computer Science and Technology, China University of Petroleum, Beijing 102249, China
Beijing Key Laboratory of Petroleum Data Mining, China University of Petroleum, Beijing 102249, China

Corresponding author: Jiwei Huang (huangjw@cup.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61972414, in part by the Beijing Natural Science Foundation under Grant 4202066, and in part by the Fundamental Research Funds for Central Universities under Grant 2462018YJRC040 and Grant 2462020YJRC001.

ABSTRACT With the prevalence of Internet of Things (IoT), edge computing has emerged as a novel computing model for optimizing traditional cloud computing systems by moving part of the computational tasks to the edge of the network for better performance and security. With the technique of services computing, edge computing systems can accommodate the application requirements with more agility and flexibility. In large-scale edge computing systems, service composition as one of the most important problems in services computing suffers from several new challenges, i.e., complex layered architecture, failures and recoveries always in the lifecycle, and search space explosion. In this paper, we make an attempt at addressing these challenges by designing a simulation-based optimization approach for reliability-aware service composition. Composite stochastic Petri net models are proposed for formulating the dynamics of multi-layered edge computing systems, and their corresponding quantitative analysis is conducted. To solve the state explosion problem in large-scale systems or complex service processes, time scale decomposition technique is applied to improving the efficiency of model solving. Additionally, simulation schemes are designed for performance evaluation and optimization, and ordinal optimization technique is introduced to significantly reduce the size of the search space. Finally, we conduct experiments based on real-life data, and the empirical results validate the efficacy of the approach.

INDEX TERMS Edge computing, reliability, service composition, stochastic Petri net, simulation-based optimization.

I. INTRODUCTION

With the rapid development of information technology, services computing has emerged as a new cross discipline that covers the science and technology of bridging the gap between business services and IT services [1]. It provides a well-defined architecture and interface to create, operate, manage and optimize the service processes with high flexibility facing future business dynamics [2]. With the services computing technology, the services can collaborate to provide users with much more powerful functionalities that atomic services commonly cannot fulfill. Such technology, namely service composition, has been a hot topic in both academia and industry for decades.

The associate editor coordinating the review of this manuscript and approving it for publication was Shouguang Wang.

With the growing prevalence of the Internet of Things (IoT) devices and technology, edge computing (or sometimes called Fog computing as a more specific solution) is delicately proposed and designed to accommodate the application requirements of IoT scenarios [3]–[5]. It is a novel networked architecture that enables cloud computing capabilities and an IT service environment at the edge of the cellular network [6]. With the combination of edge computing and services computing, pervasively distributed things (e.g. devices, sensors, actuators, smartphones and appliances) in an IoT environment can offer heterogeneous capabilities which are abstracted as services. Therefore, services computing techniques have been increasingly popular in a large variety of areas, especially in mobile applications [7].

In edge computing environments, service composition achieves significant strength in meeting the increasingly

complex and diverse service requirements [8]. It makes it possible for the edge nodes and cloud clusters to cooperate with each other in order to deal with the delays and resource limitations for customized composite services [9], [10]. Service composition has been successfully applied in multimedia and IoT services provisioning [11], [12].

In most of the modern IT systems, services are usually deployed on virtual machines (VMs) for improving the reliability and fault-tolerance, especially for the applications in some critical areas such as power supply, traffic control, medical healthcare, etc. VM migration and recovery techniques have been applied for restarting a virtual machine while keeping the services from interrupt when errors or failures occur. Nevertheless, meanwhile, the quality of service (QoS) from the user perspective has been affected by such techniques; since the error probing, fault repairing and system restarts have to take time to process.

In the edge computing paradigm, service composition meets difficulties and challenges, which are mainly manifested in the following aspects. (1) The introduction of edge layer to traditional cloud service systems makes the service provisioning system running with a complex architecture with multiple layers, each of which is able to process all or part of the required tasks. Besides traditional service selection which has been studied in the community for decades, the scheduling between the layers should also be carefully designed. (2) The failures and recoveries in the system bring several challenges to performance evaluation and then performance optimization. Their dynamics should be comprehensively considered in performance modeling, and the reliability constraints as well as performance requirements should be satisfied simultaneously in service composition. (3) With the growing popularity of services computing and IoT, the scale of services computing systems may become extremely large. Solving the system model for performance evaluation and optimization sometimes appears an extremely difficult task because of the state space explosion problem [13]. An approach with high efficiency need to be studied accordingly.

In order to address the challenges, this work presents a simulation-based optimization approach for REliability-Aware Service compositiON (REASON) for edge computing environments. The main thrust of the contribution is in the following four aspects. (1) We apply Stochastic Petri Net (SPN) to construct the atomic service model for reliability-aware performance evaluation, which is able to capture the dynamics of task arrivals, service procedures, failures, and recoveries. (2) We present a model composition scheme to formulate the complex process of service composition, where scheduling between multiple layers and service collaboration inside or beyond layers can be dynamically modeled. (3) Quantitative analyses of the SPN models are provided, and then time scale decomposition (TSD) technique is applied for solving the state explosion problem in large-scale systems or complex service processes. (4) By introducing Ordinal Optimization (OO) technology, we design a simulation-based approach of performance and reliability optimization for service composition in edge computing environments. The OO can further reduce the time consumption in performance evaluation and optimization by leveraging crude models and reducing the search space. Both theoretical and experimental results are presented, and the approach is expected to establish composition process promptly especially for large-scale edge computing systems.

The remainder of this paper is organized as follows. Section II surveys the related work on the pertinent topic of this paper. In Section III, we present SPN models of edge computing systems. In Section IV, mathematical analyses of the SPN models are conducted, and a practical solution of solving large-scale SPN models is provided. In Section V, we design a simulation-based optimization scheme of service composition which introduces the OO technique to improve the efficiency of simulations by goal softening. In Section VI, we conduct experiments based on real-life data for validating the efficacy of our approach. Finally, we conclude the paper in Section VII.

II. RELATED WORK

A. MEASUREMENT-BASED EXPERIMENTAL APPROACHES

Measurement-based scheme is the most straightforward solution to evaluate and optimize the system in most of the scenarios. In such type of approaches, hardware equipments or computer programs are designed and deployed into the real-life systems or simulators. They collect data inside the systems and calculate various quantities of our interests, based on which optimal strategies can be selected.

Some researchers have designed measurement modules or programs in the edge computing systems that have been constructed and deployed. Stusek *et al.* [14] conducted testing of selected OSGi (Open Service Gateway Initiative) frameworks for IoT systems, and provided practical performance analytical results for selecting the optimal solution. Truong and Karan [15] designed a mobile edge cloud cornering assistance (MECCA) application for examining various performance and data quality impact for mobile edge cloud applications. Morabito *et al.* [16] constructed a real testbed to evaluate the container-based solutions in IoT environment, and analyzed the power and resources consumption for performance evaluation.

Some other researchers dedicated to developed simulator or emulators to simulate or emulate a real system and its environment, conduct measurement in the system and obtain the performance indicators using statistics. D'Angelo *et al.* [17] proposed a methodology of simulating large-scale IoT environments. A multi-level simulator was employed, and performance evaluation was conducted. Bouloukakis *et al.* [18] designed queueing network models of IoT applications, and simulated the models for performance evaluation and optimization. Sonmez *et al.* [19] built a simulator tool upon CloudSim for performance evaluation of edge computing systems. Chen and Kunz [20] evaluated the performance of

IoT protocols using a network emulator, with the combination of measurement and emulation, the optimal protocol was selected.

B. MODEL-BASED ANALYTICAL APPROACHES

Most of the measurement-based approaches discussed above need to be deployed to the real-life systems or be conducted based on system log data, which makes them impractical when designing a system in the first place. To mitigate the barrier, some researchers used model-based analytical schemes for performance evaluation and optimization in edge computing systems. Although some assumptions or simplifications have to be made, the model-based approaches are quite computationally efficient and are able to reveal the interrelationships between the performance attributes and system parameters, making them valuable in system design and optimization.

Queueing theory is one of the most popular models of performance evaluation and optimization. Xia *et al.* [21] applied queueing network model for providing an analytic solution of performance attributes, which was expected to optimize the energy efficiency of cloud data centers. Chen *et al.* [22] constructed the queueing model for mobile edge computing systems, and formulated a stochastic optimization problem for energy-efficient dynamic task offloading. Li *et al.* [23] used the M/M/k queueing model to estimate the response time of IoT nodes, based on which explored optimal QoS-aware services composition for service-oriented IoT. Bouloukakis *et al.* [24] proposed a queueing network model of the middleware overlay infrastructure of mobile things. Wang *et al.* [25] formulated the edge nodes using the queueing model, and designed a near-optimal offloading scheme for the Internet of Vehicles.

Petri net is another powerful model for describing the dynamics of discrete event systems. In recent years, there have been many works applying Petri net models to handle hard problems in computing systems [26]–[28]. Stochastic Petri net (SPN) is a specific Petri net model which introduces stochastic processes for performance analysis. He *et al.* [29] defined a stochastic colored Petri net (SCPN) model for the IoT-based smart environment and proposed a game model for security situational awareness. Zhang *et al.* [30] proposed a hierarchical-timed-colored Petri net (HTCPN) model to analyze the sensor performance data for IoT-enabled real-time environment. Ni *et al.* [31] designed priced timed Petri nets of fog computing systems, and obtained optimal strategies of resource allocation. Ding *et al.* [32] applied the Petri net model to interactive control for obtaining an optimal systematic strategy.

C. SUMMARY

Although there have been several cutting-edge research works dedicating to the optimal service provisioning in edge computing, the combination of quantitative modeling and performance optimization remains largely unexplored. Also, both the measurement-based and model-based approaches still may face state space and search space explosion problems, making several schemes impractical in large-scale edge computing systems. Last but not least, there lacks a comprehensive study on performance and reliability in the system evaluation and optimization. Therefore, a systematic and efficient methodology of reliability-aware QoS optimization needs to be studied.

Previously, we have conducted some research works on the topic of performance evaluation and optimization in edge computing service systems. In [33], we put forward queueing network models for performance evaluation of IoT service systems. In [34], we introduced simulation techniques to queueing models and designed an optimization scheme of service selection for mobile edge computing. In [35], based on the queueing model, we formulated a stochastic optimization problem of energy efficient task scheduling for sensor hubs in IoT, and designed an efficient scheme to solve the problem. In [36], we further studied the task scheduling and resource management problem in mobile edge computing by designing a more efficient optimization algorithm. In [37], we applied the generalized SPN model and made a preliminary attempt to evaluate the performance and reliability simultaneously of IoT services. In this paper, we combine the SPN modeling and computer simulation techniques, and jointly study the performance evaluation and optimization problems for reliability-aware service composition in the edge computing environment.

III. SPN MODEL OF SERVICE COMPOSITION

A computer system constructed with edge computing architecture consists of two main layers, i.e., edge layer and cloud layer. In each layer, there are multiple virtualized servers fulfilling different requirements from users. In this section, we apply Stochastic Petri Net (SPN) model to formulate both the layers, and present a model aggregation approach for service composition.

A. SPN MODEL OF EDGE SERVERS

Commonly, in the edge layer, several IoT devices or users are submitting their requests to the system. At the access network end, edge servers are deployed to process part of the requests for guaranteeing the quick response and privacy. Meanwhile, some of the requests especially the computing-intensive and data-intensive ones will be submitted to the cloud for high performance computing. To capture the architecture and dynamics of the service procedures, we construct an SPN model for performance evaluation. Formally, the definition of an SPN is given as follows.

Definition 1 (Stochastic Petri Net): A Stochastic Petri Net (SPN) Σ is defined as a 6-tuple:

$$\Sigma = (P, T, A, w, m_0, \lambda)$$

where

- $P = \{p_1, p_2, \ldots, p_{N_P}\}$ is the finite set of places.
- $T = \{t_1, t_2, \ldots, t_{N_T}\}$ is the finite set of transitions.
- $A \subseteq (P \times T) \cup (T \times P)$ is the set of arcs from places to transitions and from transitions to places.

- $w : A \rightarrow \mathbb{N}^+$ is the weight function of the arcs.
- $m_0 : P \rightarrow \mathbb{N}$ is the initial state of the SPN.
- $\lambda : T \rightarrow \mathbb{R}^+$ is the set of firing rates of the transitions.

In an SPN, we use *tokens* to represent the resources or tasks in the system, and assign them to the places. A *marking* denotes how the tokens are assigned to the places, whose definition is formally given in Definition 2.

Definition 2 (marking): A marking m of an SPN $\Sigma = (P, T, A, w, m_0)$ is a function $m : P \rightarrow \mathbb{N}$.

Commonly, a marking is expressed by a vector as $\mathbf{m} = [m(p_1), m(p_2), \ldots, m(p_{N_P})]$, where $m(p_i)$ denotes the number of tokens in place p_i. At a certain time point, we defined the *state* of an SPN by a marking vector.

Considering the dynamics of the service processes in the edge layer, with the definition of SPN, we propose the SPN model of edge servers. We use places to indicate the state of an edge computing system which are conventionally represented by circles, and timed transitions to denote the change of state illustrated by bars. We draw directed arrows for representing the arcs, and write their weights on them (but usually omitted if equal to 1). Tokens under marking m are indicated by dark dots in the appropriate places where $m(p_i) \neq 0$. Afterwards, the SPN model of edge servers is provided as Figure 1.

Basically, the dynamics of the service procedures in the edge layer is illustrated as follows. Firstly, the users or IoT devices submit their requests to the system at certain rates conforming to certain distributions. It is possible for them to select one or multiple edge servers deployed on the nearby base stations. The requests are formulated by the tokens in place p_I representing the initial state of the system, and the task arrivals at the edge server indexed by i are captured by a transition labeled as t_{Ai}. Such procedures are called *Task Arrival Model* in Figure 1. Secondly, upon the arrivals at each edge server i, the requests are queued in a buffer, which is represented by place p_{Qi}. Once the server is available for service, the request will be handled and then either completed or submitted to the cloud site for further processing. The service procedure is formulated by transition t_{Si}. The queueing behaviors are described by the *Queueing Model*. Meanwhile, failures may happen on each of the edge servers, which are captured by transition t_{Fi} resulting in the transition of the server into failed state under different failure rate. With virtualization techniques, the server can be recovered or restarted, which is represented by transition t_{Ri} with its particular repair rate. The above dynamic failure and recovery processes are formulated by the *Reliability Model* which is another sub-model in Figure 1.

B. SPN MODEL OF CLOUD SERVERS

Similarly, we present the SPN model of cloud servers. Figure 2 illuminates an example.

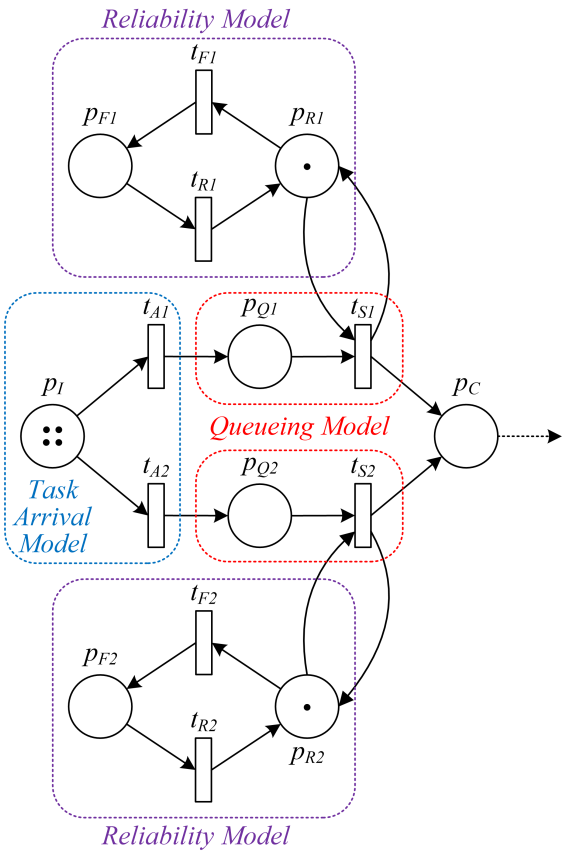

FIGURE 1. SPN model of edge servers.

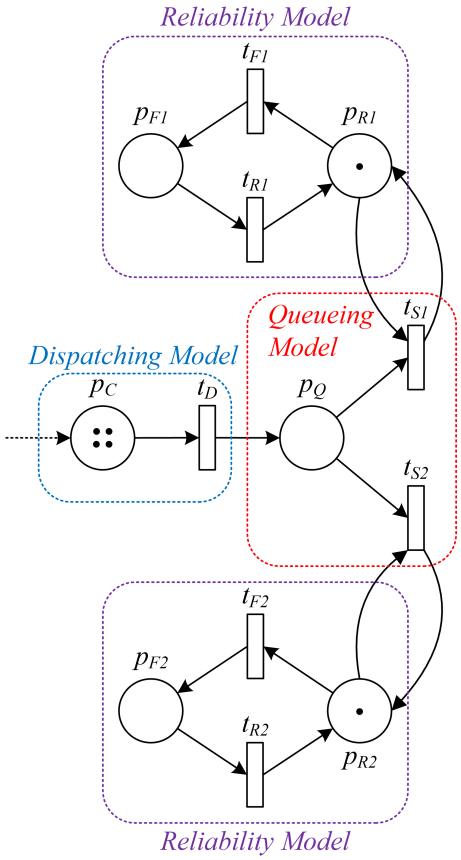

FIGURE 2. SPN model of cloud servers.

Initially, the requests from the edge side are dispatched to one or multiple cloud servers for processing. The dispatching, also called task scheduling, may take some time for calculating the optimal (or near-optimal) solution, which is represented by the transition t_D in the sub-model namely *Dispatching Model*.

The requests arrived at each of the servers will also be buffered, and have to wait until the server is available. Different from the edge layer, most of the cloud systems consist of multiple servers constituting a cluster. A cluster shares the same buffer but the requests can be processed in a parallel way. Specifically, the characteristic of multiple servers with a shared buffer is formulated by the *Queueing Model* in Figure 2.

C. MODEL COMPOSITION

An edge computing system consists of both edge and cloud layers. In order to formulate the whole system with the complex layered architecture, we present a model composition scheme in this subsection.

The basic idea of model composition is to "splice" the edge model onto the cloud model according to the task offloading pattern. The "output" place p_C in the SPN model of edge servers is spliced to the "input" place of the cloud model. We should note that, the services can dynamically composite, and the service selection may happen when choosing proper services deployed on certain cloud servers. Therefore, the requests come from the edge side can be scheduled to certain cloud servers, demanding for an optimal policy.

The main thrust of model composition is to clearly analyze the relationships between the edge side and cloud side in service composition in an edge computing environment, and construct the SPN models accordingly. Figure 3 shows an example of the composite model. In this sample edge computing system, there are traffic sources from two services. Service selection may take place for the first one, since two edge servers are able to process the requests submitted from the services. For the other one, the requests will be routed directly to the corresponding edge server. At the cloud side, two clusters are available for handling the request. The former one is equipped with two cloud servers, while the latter consists of three HPC nodes.

IV. QUANTITATIVE ANALYSIS OF SPN MODEL

With the SPN models, quantitative analysis of performance and reliability attributes can be carried out, which is the foundation of QoS-aware service composition. In this section, we present theoretical and technical approaches to performance and reliability evaluation by solving the SPN models for edge computing service systems.

A. SOLVING ATOMIC MODEL

The basic idea of the methodology for solving the SPN model is to construct the underlying continuous-time Markov chain (CTMC). To facilitate the construction process, some notations are defined as follows.

- *Root node*: is defined as the first state of the CTMC, obtained from the initial marking m_0.
- *Terminal node*: is defined as a node from which no transition of the SPN can fire.
- *Duplicate node*: is a node that is identical to a node already in the CTMC.
- *Node dominance*: we define that marking m_1 dominates marking m_2, denoted by $m_1 >_d m_2$, if the following two conditions hold:
  1) $m_1(p_i) \geq m_2(p_i)$, for all $i = 1, 2, \ldots, n_{N_P}$;
  2) $m_1(p_i) > m_2(p_i)$, for at least some $i = 1, 2, \ldots, n_{N_P}$.
- *Symbol* ω: means "infinity" in representing some places with unbounded tokens. For $\forall n \in \mathbb{N}$, we specify $n < \omega$ and $\omega + n = \omega - n = \omega$.

We borrow the idea of constructing the reachability tree from traditional Petri nets and present the algorithm of generating the underlying CTMC of an SPN Σ as Algorithm 1.

Algorithm 1 Algorithm of Constructing CTMC

Input: SPN $\Sigma = (P, T, A, w, m_0, \lambda)$
Output: transition rate matrix Q

Initialize m_0 as the first state of the CTMC;
Let $\Psi \leftarrow \{m_0\}$, $Q \leftarrow 0$;
for $\psi \in \Psi$ **do**
  if *no transition can fire at state* ψ **then**
    ψ is a terminal node;
  else
    Find ψ' s.t. $\psi \xrightarrow{t_j} \psi'$ for some $t_j \in T$;
    if ψ' *is a duplicate node* **then**
      Set the transition rate $Q(\psi, \psi') \leftarrow \lambda_j$;
    else
      if $\psi(p_i) = \omega$ *for some* $p_i \in P$ **then**
        Let $\psi'(p_i) \leftarrow \omega$;
      if $\exists \theta$ *s.t.* $\psi' >_d \theta$ **then**
        Let $\psi'(p_i) \leftarrow \omega$ for all $p_i \in P$ s.t. $\psi'(p_i) > \theta(p_i)$;
      Add a new node ψ' to the CTMC, and set the transition rate $Q(\psi, \psi') \leftarrow \lambda_j$;
      Let $\Psi \leftarrow \Psi \cup \{\psi'\}$
  Let $\Psi \leftarrow \Psi - \{\psi\}$;
Let $Q(\psi, \psi) \leftarrow -\sum_k \lambda_k$ for all $\psi \in CTMC$;

Suppose there are n states of the constructed CTMC, and let $X = [x_1, x_2, \ldots, x_n]$ denote the steady-state probability of the CTMC. For the steady-state probability analysis, we have

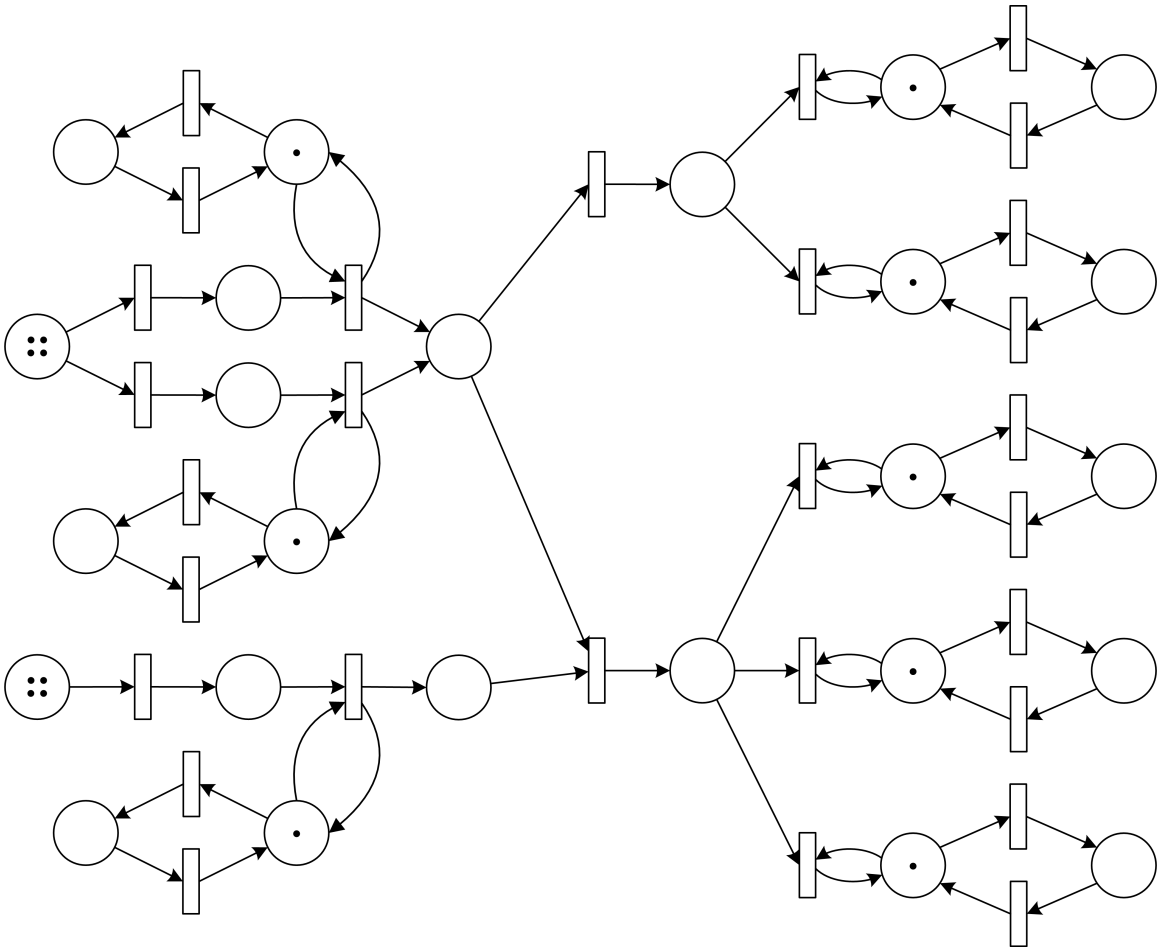

FIGURE 3. SPN model of an edge computing system.

the following equations as Eq. (1).

$$\begin{cases} X \cdot Q = 0; \\ \sum_{i=1}^{n} x_i = 1. \end{cases} \tag{1}$$

Hereafter, one can obtain the steady-state probability of the state of a place $p \in P$ containing $i \in \mathbb{N}$ tokens using Eq. (2).

$$P\{m(p) = i\} = \sum_{m_j(p)=i} X_j. \tag{2}$$

Therefore, we obtain the average number of tokens for each place $p_i \in P$ as Eq. (3). This expression can help when calculating the average queue length, utilization of servers, reliability of services, etc.

$$u(p_i) = \sum_j j \cdot P\{m(p_i) = j\}. \tag{3}$$

With Little's Law, the average response time of the tasks being processed by the server (or cluster) indexed by i can be calculated by Eq. (4), where λ is the task arrival rate at the server (or cluster).

$$T = u(p_i)/\lambda. \tag{4}$$

Moreover, the reliability of the server (or cluster) is calculated by Eq. (5).

$$R = \frac{P\{m(p_R) = 1\}}{P\{m(p_F) = 1\} + P\{m(p_R) = 1\}}. \tag{5}$$

B. ANALYSIS OF SYSTEM MODEL

An edge computing system may consist of several edge servers and cloud clusters, which makes its SPN model quite complicated, embedded with a number of places and transitions. Although one can solve the SPN model with the same methodology presented in the above sub-section, we provide a more efficient analytical approach of the system model in this part.

It is well-known that the failure rates and repair rates are commonly orders of magnitude smaller than the arrival and service rates. Also, some existing open-source data sets have validated this fact [38]. It has been shown that the inter-arrival times and service times are usually in the magnitude of seconds or microseconds, while the times between failures are commonly several hours or even days. Therefore, we classify the transitions into two sub-sets. The ones representing the task arrivals, service processes, and task scheduling are

defined as *fast transitions* denoted by the set of T_f, while the transitions of failures and recoveries constitute the set of *slow transitions* expressed by T_s.

Then we apply time scale decomposition (TSD) [39] technique to the SPN analysis. At the first step, we remove all the slow transitions in T_s, and then obtain the simplified SPN model at fast time scale, expressed by Σ_f. In Σ_f some places such as *Failed* states have no initial tokens and will never enter, and thus we are able to remove them without affecting the analysis of the SPN model. Afterwards, the simplified SPN model at fast time scale of edge servers is shown in Figure 4.

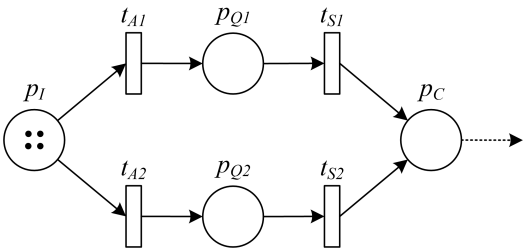

FIGURE 4. SPN model of edge servers at fast time scale.

At the second step, we aggregate each SPN model at fast time scale into a single place, and generate an SPN model with the new places and slow transitions, denoted by Σ_s. The initial state of Σ_s is calculated from the initial state of Σ and of Σ_f. Illustratively, Figure 5 shows the aggregated SPN model of Figure 1.

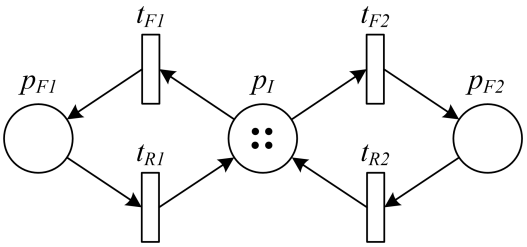

FIGURE 5. Aggregated SPN model of edge servers at slow time scale.

Consequently, we transfer the initial SPN solving problem into two problems on the aggregated SPN models. Since the state space of the underlying CTMC model increases exponentially when SPN model scales up, the TSD approach is able to significantly reduce the time consumption and space complexity of SPN analysis. Without loss of generality, the SPN model at fast time scale of the edge computing system shown in Figure 3 is illustrated by Figure 6, while the aggregated model at slow time scale is shown in Figure 7.

V. SIMULATION-BASED OPTIMIZATION OF SERVICE COMPOSITION

With the SPN models and their analysis, we are able to evaluate the performance and reliability of an edge computing

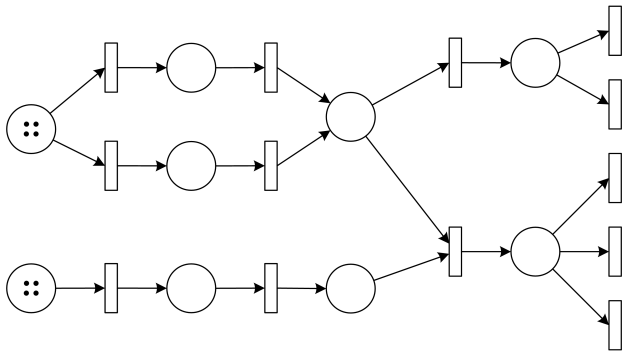

FIGURE 6. SPN model of an edge computing system at fast time scale.

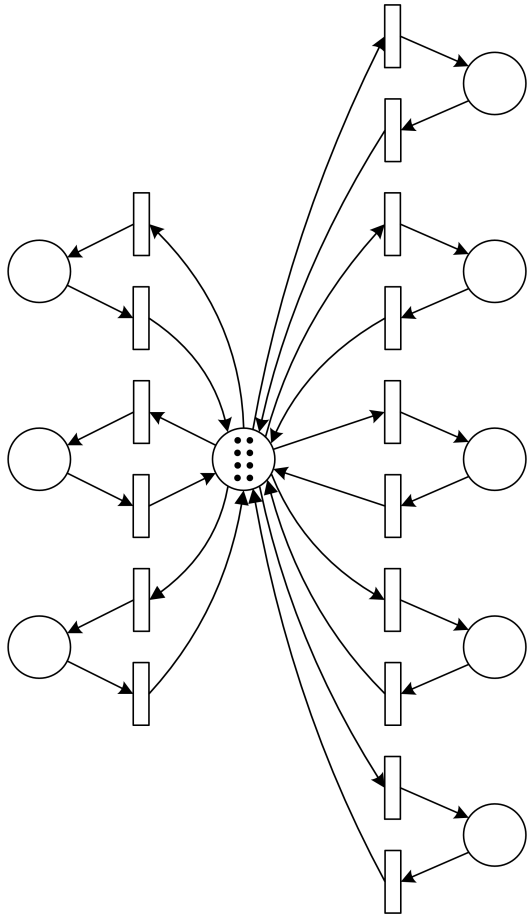

FIGURE 7. Aggregated SPN model of an edge computing system at slow time scale.

system with different service composition policies. According to the evaluation results, optimal policy can be selected, and the QoS requirements can be satisfied. In this section, we present a practical solution for optimal REliability-Aware Service compositiON (REASON) in the edge computing environment.

A. SIMULATION-BASED OPTIMIZATION BASED ON SPN MODEL

In the previous section, we present the methodology of solving the SPN models. Basically, the SPN models are transformed into their corresponding CTMC models, and then

CTMC models can be mathematically solved using a linear structural equation system. With the analytical results, the performance and reliability attributes can be obtained.

In reality, however, the scale of an edge computing system may be significantly large, and hence solving SPN models may suffer from the state explosion problem. Although we have applied the TSD technique to reduce the state space of the SPN model, sometimes it is still difficult to solve the problem within an acceptable time using the pure-mathematical solutions. To attack this challenge, we introduce computer simulation techniques for solving the SPN models practically. We should note that, we can apply computer simulation before or after applying the TSD to the SPN model, which means that TSD can benefit both mathematical and experimental approaches.

The basic idea is to design and implement a series of analytical experiments based on the SPN models, and the data from the experimental processes are collected to estimate performance quantities. In each of the processes, the simulation is implemented with the event-driven methodology. Here in our SPN models, the "event" means the firing of the transitions. A series of events is generated according to the dynamics of the system, and is kept updated which drives the models continuously evolved. The feasible events are stored in the sorted *Event List* which is the key data structure of our simulation experiments. *Random Variate Generators* are implemented and invoked for event updating. A general framework of the event-driven simulation for performance evaluation is illuminated by Figure 8.

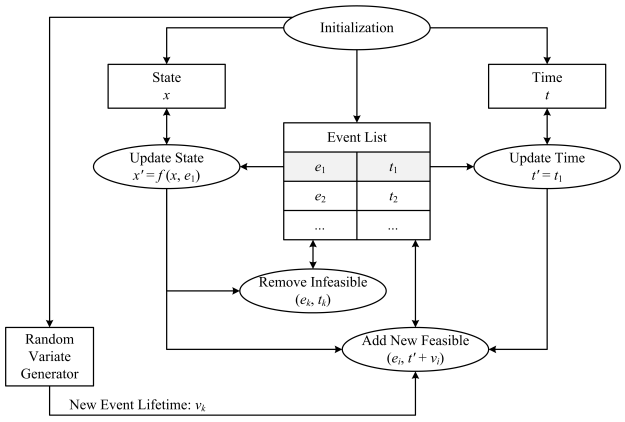

FIGURE 8. Framework of simulation-based performance evaluation with SPN model.

There are three ways to identify the parameters in the simulations. Firstly, the parameters can be obtained from real-life systems by analyzing system log data. This is the best way to generate accurate analytical results of performance evaluation. Secondly, the parameters can be set manually by the administrators in the system design phase. In reality, some parameters might be difficult to acquire especially before the deployment of the systems, and thus some assumptions may be made. Thirdly, the parameters can be tuned during the simulations. A series of simulations will be conducted, according to which the optimal parameter setting will be found.

Initially, the event list is generated according to the model parameter settings and input data. For example, some trace data from real-life systems can be used to generate the time points of task arrivals, and thus a number of arrival events are initialized and input to the event list. Afterward, the procedure of a simulation experiment is to continuously repeat the following five steps.

1) The first entry (e_1, t_1) from the *Event List* is selected and removed from the event list.
2) The simulation *Time* is updated to the new event time t_1.
3) We update the *State* of the system x according to the state transitions of the SPN model.
4) If some events are not possible to happen in the updated state, their corresponding entries from the event list will be removed.
5) According to the dynamics of the SPN model, we generate new feasible events under the updated state and add them to the event list, whose lifetimes are generated from the *Random Variate Generator*.

After the completion of the simulation processes, we collect all the raw data and calculate the performance attributes. The average response time can be obtained by analyzing the interval between the departure and arrival of each task, and the reliability attribute can be calculated by Eq. (5) shown in Section IV-A.

The simulation-based optimization of service composition consists of the following three steps. Initially, we find the feasible policies of service composition according to the requirements gathered from users, and construct SPN models of different policies. Next, we conduct simulation experiments for each of the policies and obtain the analytical results of performance and reliability quantities. Finally, we compare the data and find the optimal solution among the policies.

B. PRACTICAL SOLUTION FOR LARGE-SCALE EDGE COMPUTING SYSTEMS

For some extremely large-scale edge computing systems, the number of their feasible service composition policies may also be very large. It might be impractical in reality to experimentally simulate all the candidate policies and select the optimal one. Therefore, we have to further improve the efficiency of the optimization procedures to make our solution more practical in large-scale edge computing systems.

We introduce the Ordinal Optimization (OO) technique [40], the basic idea of which is to partially sacrifice the optimality for finding the acceptable near-optimal solutions in a reasonable time. Instead of finding the global optimal policy, our goal is softened into obtaining a good enough solution with high probability (the probability is called "*Alignment Probability*"). We assume that G denotes the good enough set consisting of the top-g ($g = |G|$) feasible solutions for the optimization problem, and S is the selected top-s ($s = |S|$) solutions obtained by our approach. Then the alignment probability is mathematically expressed as Eq. (6),

where $k \in \mathbb{N}^+$ is called the *Alignment Level*.

$$\Pr(|G \cap S| \geq k) \geq \alpha. \tag{6}$$

It can be proved that, even for the blind picking strategy, the alignment probability converges to 1 in an exponentially fast speed with respect to the size of the set G and S. Hence, the efficiency of our approach can be further significantly enhanced by goal softening using the OO fashion. Basically, there are three main steps in our approach as follows.

1) Construct a computationally fast crude model to estimate the quantitative objectives of all feasible policies.
2) Estimate the Ordered Performance Curve (OPC) class of the problem and the noise level of the crude model, and then calculate the size of set S.
3) Use the crude model to find the top-s policies which constitute the selected set S, and run simulations with a precise model for each of the top-s policies. The best one is selected as the output of our approach.

Constructing a crude model is the first step of our approach. In order to estimate the system very fast, we use M/M/c queueing model to evaluate the performance attributes and Reliability Block Diagram (RBD) to analyze the reliability quantity. Each edge server is modeled by an M/M/1 queue, while each cloud cluster is formulated as an M/M/c queueing system where c is the number of virtualized (or physical) servers in the cluster. With fundamental queueing theory, the expected response time of a task processed on an edge server represented by T_e can be obtained by Eq. (7), and on average it will cost T_c time for a cloud cluster to serve the requests expressed as Eq. (8).

$$T_e = \frac{1}{\mu_e - \lambda_e}; \tag{7}$$

$$T_c = \frac{1}{\mu_c} + \frac{1}{\mu_c} \cdot \frac{(c\rho_c)^c}{c!} \cdot \frac{P_0}{c(1-\rho_c)^2}; \tag{8}$$

where λ_e and λ_c are arrival rates of the edge server and the cloud cluster respectively, and μ_e and μ_c are the service rates. $\rho_c = \lambda_c/(c\mu_c)$ is the utilization of the cloud cluster, and P_0 is expressed as Eq. (9).

$$P_0 = \left[\sum_{k=0}^{c-1} \frac{(c\rho_c)^k}{k!} + \frac{(c\rho_c)^c}{c!} \cdot \frac{1}{1-\rho_c} \right]^{-1}. \tag{9}$$

The RBD model for analyzing the reliability of the system has been explored in our previously published paper [41], where the reliability attribute, as well as mean time to failure (MTTF) and mean time to repair (MTTR), have been fully discussed. We use the analytical expressions to estimate the reliability of the system for the crude model.

With the estimation results, the second step is to estimate the parameters of OO. The alignment level k and the size of good enough set g should be specified by administrators/users, and then we conduct a sampled simulation experiment to calculate the OPC class and noise level of the crude model. The appropriate s can be found from a pre-calculated table which have been provided in [42].

Finally, we conduct our simulation experiments with all the techniques we have presented in the above sections as the precise model to evaluate the top-s candidate policies. Afterward, with all the collected data from simulations, we are able to find the optimal policy of service composition.

VI. EXPERIMENTAL RESULTS

In this section, we conduct simulation experiments to validate our approach. Data sets from real-world systems are used for parameter settings, and experimental results are provided.

A. EXPERIMENTAL SETTINGS

We simulate an edge computing system with our SPN models. Experimental data is collected and analyzed, and empirical results are obtained to validate our approach.

In our experiments, we consider a popular IoT scenario where several cars are connected and upload their GPS and status data to the system for processing. A real-world data set namely "T-Drive" [43], [44] is applied to simulate the task arrivals of the edge computing system. The data was collected by Microsoft Research from 10,357 taxis in the city of Beijing, China. Nearly 15 million pieces of GPS trajectory data were recorded, covering over 9 million kilometers during a period of 1 week in 2008.

For the servers, we simulate their failures and recoveries according to the trace data provided by Los Alamos National Laboratory (LANL) [38]. The data was collected from 23 High Performance Computing (HPC) systems during the period of over 9 years from 1996 to 2005. There are totally 4,750 machines and 24,101 processors in the system, providing computing, Grid or web applications to the users. During the service procedures, 23,739 failure situations have been recorded, and the types and reasons have been analyzed. We calculate the failure rates and repair rates to define the parameters in our SPN models.

The service times are assumed to be exponentially distributed, and we apply a random scheduling algorithm to the system for task scheduling and load balancing in the following discussions. Different server deployment strategies, as well as parameter settings, are simulated and evaluated by our approach, among which optimal policies of service composition among edge sides and cloud sides can be selected.

B. EXPERIMENTAL RESULTS

We firstly tune the parameter settings of the systems to validate the effectiveness of our approach. Figure 9 illustrates the variance of service reliability with the scaling up of edge servers. With more edge servers processing the tasks arrived at the system, the reliability of edge side can be enhanced. Consequently, the end-to-end service reliability increases simultaneously. Also, we see from the results that the reliability values become more smooth when the number of servers goes large, which means that it become less useful for improving the reliability of a distributed system to add more servers when there exist a number of servers being deployed. It validates why IT companies commonly deploy

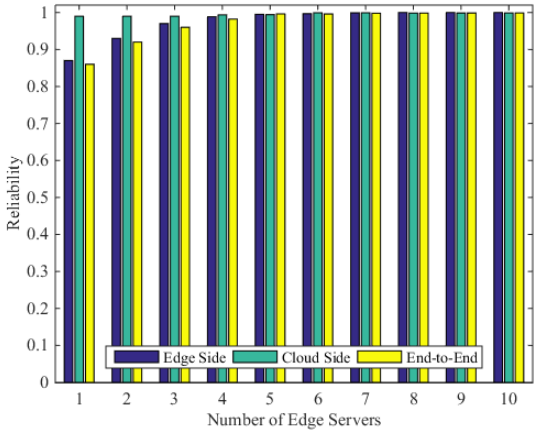

FIGURE 9. Service reliability with edge server scaling up.

only two backup servers for their online applications. At the cloud side, we have the similar experimental results shown in Figure 10. More servers or clusters are able to process the services in a parallel way, and thus providing more backup when one of the machine fails. Arranging backup machines is a popular way of enhancing system reliability in most of the computer systems, which accords with our experimental results.

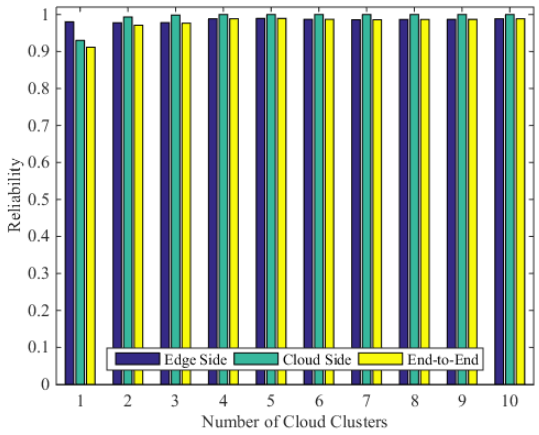

FIGURE 10. Service reliability with cloud cluster scaling up.

Figures 11 and 12 illuminate the average response time with the increase of edge servers and cloud clusters, respectively. The experimental results validate our analyses. With the increase of servers, the average response time decreases. The reason is that more servers are available for processing the user requests concurrently resulting in less delay. Both the mathematical analyses of queueing theory and SPN can draw the same conclusion.

Finally, we consider a service composition scenario where services deployed on the edge side and services deployed on cloud side need to cooperate for fulfilling a complex workflow process. The objective is to minimize the average response time while fully considering the failure and

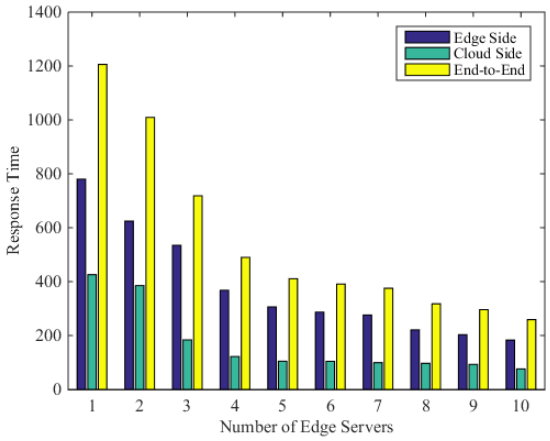

FIGURE 11. Average response time with edge server scaling up.

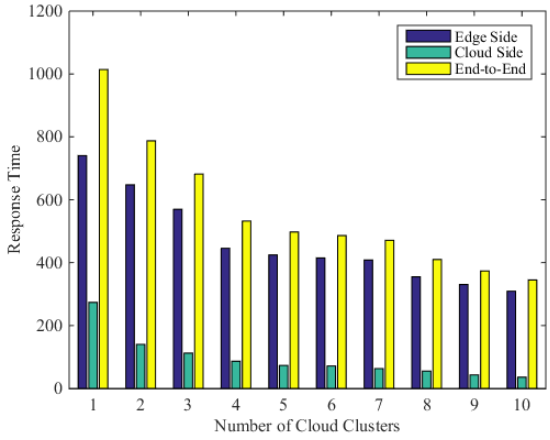

FIGURE 12. Average response time with cloud cluster scaling up.

recovery processes during the service procedures. Figure 13 illuminates the experimental results of the randomly selected 100 feasible solutions. We plot the response time and reliability of the feasible policies and the selected top-8 ones with our approach. Pareto optimality, which is for the optimization

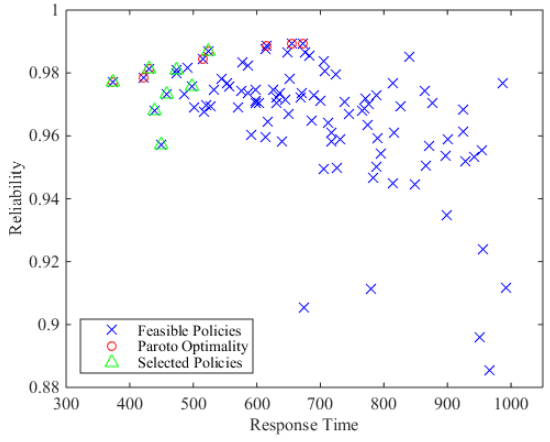

FIGURE 13. Average response time with cloud cluster scaling up.

of a vector of multiple criteria among optimal combinations, is manually analyzed for each of the service composition policy [45]. In our experiments, the blind picking algorithm is applied to select the good enough solutions by crude model, which is able to provide a worst-case analysis of our optimization approach [40]. It is shown from our experimental results that 37.5% of our selected set is Pareto optimal, indicating that it is highly possible for our approach to find the optimal solutions. We should also note that nearly 75% of the policies obtained by our approach are in the top-10 optimal ones. Therefore, Figure 13 validates the efficacy of our approach.

VII. CONCLUSION

In this paper, we study the reliability-aware service composition problem in edge computing from a systematic view by integrating both performance evaluation and QoS optimization. Stochastic Petri net models of edge servers and cloud clusters are proposed, and time scale decomposition technique is applied for their efficient quantitative analysis. Based on the SPN models, an event-driven simulation scheme is designed to evaluate the service composition policies, according to which the optimal strategy can be selected. For some large-scale edge computing systems, we apply the OO technique in order to further reduce the search space and hence decrease the time consumption of the optimization procedures. The efficacy of our approach is validated by experimental results. This work is expected to provide both theoretical and practical reference to the design and optimization of services computing systems in the edge computing environment.

REFERENCES

[1] L.-J. Zhang, J. Zhang, and H. Cai, *Services Computing*. Berlin, Germany: Springer, 2007.

[2] L.-J. Zhang, "Services computing: A new discipline," *Int. J. Web Services Res.*, vol. 2, pp. 1–4, May 2005.

[3] P. G. Lopez, A. Montresor, D. Epema, A. Datta, T. Higashino, A. Iamnitchi, M. Barcellos, P. Felber, and E. Riviere, "Edge-centric computing: Vision and challenges," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 45, no. 5, pp. 37–42, 2015.

[4] *Fog Computing and the Internet of Things: Extend the Cloud to Where the Things are*, Cisco, San Jose, CA, USA, White Paper, Apr. 2015. [Online]. Available: https://www.cisco.com/c/dam/en_us/solutions/trends/iot/docs/computing-overview.pdf

[5] M. Iorga, L. Feldman, R. Barton, M. J. Martin, N. Goren, and C. Mahmoudi, "Fog computing conceptual model," Nat. Inst. Standards Technol., Gaithersburg, MD, USA, Tech. Rep. 500-325, Mar. 2018.

[6] A. Ahmed and E. Ahmed, "A survey on mobile edge computing," in *Proc. 10th Int. Conf. Intell. Syst. Control (ISCO)*, Jan. 2016, pp. 1–8.

[7] Y. Jararweh, A. Doulat, O. AlQudah, E. Ahmed, M. Al-Ayyoub, and E. Benkhelifa, "The future of mobile cloud computing: Integrating cloudlets and mobile edge computing," in *Proc. 23rd Int. Conf. Telecommun. (ICT)*, May 2016, pp. 1–5.

[8] H. Wu, S. Deng, W. Li, M. Fu, J. Yin, and A. Y. Zomaya, "Service selection for composition in mobile edge computing systems," in *Proc. IEEE Int. Conf. Web Services (ICWS)*, Jul. 2018, pp. 355–358.

[9] I. Al Ridhawi, Y. Kotb, and Y. Al Ridhawi, "Workflow-net based service composition using mobile edge nodes," *IEEE Access*, vol. 5, pp. 23719–23735, 2017.

[10] Y. Chen, N. Zhang, Y. Zhang, X. Chen, W. Wu, and X. S. Shen, "Energy efficient dynamic offloading in mobile edge computing for Internet of Things," *IEEE Trans. Cloud Comput.*, to be published.

[11] S. Deng, Z. Xiang, J. Yin, J. Taheri, and A. Y. Zomaya, "Composition-driven IoT service provisioning in distributed edges," *IEEE Access*, vol. 6, pp. 54258–54269, 2018.

[12] Y. Duan, X. Sun, H. Che, C. Cao, Z. Li, and X. Yang, "Modeling data, information and knowledge for security protection of hybrid IoT and edge resources," *IEEE Access*, vol. 7, pp. 99161–99176, 2019.

[13] H. Dou, K. Barkaoui, H. Boucheneb, X. Jiang, and S. Wang, "Maximal good step graph methods for reducing the generation of the state space," *IEEE Access*, vol. 7, pp. 155805–155817, 2019.

[14] M. Stusek, J. Hosek, D. Kovac, P. Masek, P. Cika, J. Masek, and F. Kropfl, "Performance analysis of the OSGi-based IoT frameworks on restricted devices as enablers for connected-home," in *Proc. 7th Int. Congr. Ultra Modern Telecommun. Control Syst. Workshops (ICUMT)*, Oct. 2015, pp. 178–183.

[15] H.-L. Truong and M. Karan, "Analytics of performance and data quality for mobile edge cloud applications," in *Proc. IEEE 11th Int. Conf. Cloud Comput. (CLOUD)*, Jul. 2018, pp. 660–667.

[16] R. Morabito, I. Farris, A. Iera, and T. Taleb, "Evaluating performance of containerized IoT services for clustered devices at the network edge," *IEEE Internet Things J.*, vol. 4, no. 4, pp. 1019–1030, Aug. 2017.

[17] G. D'Angelo, S. Ferretti, and V. Ghini, "Multi-level simulation of Internet of Things on smart territories," *Simul. Model. Pract. Theory*, vol. 73, pp. 3–21, Apr. 2017.

[18] G. Bouloukakis, I. Moscholios, N. Georgantas, and V. Issarny, "Simulation-based queueing models for performance analysis of IoT applications," in *Proc. 11th Int. Symp. Commun. Syst., Netw. Digit. Signal Process. (CSNDSP)*, Jul. 2018, pp. 1–5.

[19] C. Sonmez, A. Ozgovde, and C. Ersoy, "EdgeCloudSim: An environment for performance evaluation of edge computing systems," *Trans. Emerg. Telecommun. Technol.*, vol. 29, no. 11, p. e3493, Nov. 2018.

[20] Y. Chen and T. Kunz, "Performance evaluation of IoT protocols under a constrained wireless access network," in *Proc. Int. Conf. Sel. Topics Mobile Wireless Netw. (MoWNeT)*, Apr. 2016, pp. 1–7.

[21] Y. Xia, M. Zhou, X. Luo, S. Pang, and Q. Zhu, "A stochastic approach to analysis of energy-aware DVS-enabled cloud datacenters," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 45, no. 1, pp. 73–83, Jan. 2015.

[22] Y. Chen, N. Zhang, Y. Zhang, X. Chen, W. Wu, and X. S. Shen, "TOFFEE: Task offloading and frequency scaling for energy efficiency of mobile devices in mobile edge computing," *IEEE Trans. Cloud Comput.*, to be published.

[23] L. Li, S. Li, and S. Zhao, "QoS-aware scheduling of services-oriented Internet of Things," *IEEE Trans Ind. Informat.*, vol. 10, no. 2, pp. 1497–1505, May 2014.

[24] G. Bouloukakis, I. Moscholios, N. Georgantas, and V. Issarny, "Performance modeling of the middleware overlay infrastructure of mobile things," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2017, pp. 1–6.

[25] X. Wang, Z. Ning, and L. Wang, "Offloading in Internet of vehicles: A fog-enabled real-time traffic management system," *IEEE Trans Ind. Informat.*, vol. 14, no. 10, pp. 4568–4578, Oct. 2018.

[26] X. Guo, S. Wang, D. You, Z. Li, and X. Jiang, "A siphon-based deadlock prevention strategy for S3PR," *IEEE Access*, vol. 7, pp. 86863–86873, 2019.

[27] W. Duo, X. Jiang, O. Karoui, X. Guo, D. You, S. Wang, and Y. Ruan, "A deadlock prevention policy for a class of multithreaded software," *IEEE Access*, vol. 8, pp. 16676–16688, 2020.

[28] S. Wang, D. You, and M. Zhou, "A necessary and sufficient condition for a resource subset to generate a strict minimal siphon in s 4PR," *IEEE Trans. Autom. Control*, vol. 62, no. 8, pp. 4173–4179, Aug. 2017.

[29] F. He, Y. Zhang, H. Liu, and W. Zhou, "SCPN-based game model for security situational awareness in the intenet of things," in *Proc. IEEE Conf. Commun. Netw. Secur. (CNS)*, May 2018, pp. 1–5.

[30] Y. Zhang, W. Wang, N. Wu, and C. Qian, "IoT-enabled real-time production performance analysis and exception diagnosis model," *IEEE Trans. Autom. Sci. Eng.*, vol. 13, no. 3, pp. 1318–1332, Jul. 2016.

[31] L. Ni, J. Zhang, C. Jiang, C. Yan, and K. Yu, "Resource allocation strategy in fog computing based on priced timed Petri nets," *IEEE Internet Things J.*, vol. 4, no. 5, pp. 1216–1228, Oct. 2017.

[32] Z. Ding, H. Qiu, R. Yang, C. Jiang, and M. Zhou, "Interactive-control-model for human–computer interactive system based on Petri nets," *IEEE Trans. Autom. Sci. Eng.*, vol. 16, no. 4, pp. 1800–1813, Oct. 2019.

[33] J. Huang, S. Li, Y. Chen, and J. Chen, "Performance modelling and analysis for IoT services," *Int. J. Web Grid Services*, vol. 14, no. 2, pp. 146–169, 2018.

[34] J. Huang, Y. Lan, and M. Xu, "A simulation-based approach of QoS-aware service selection in mobile edge computing," *Wireless Commun. Mobile Comput.*, vol. 2018, Nov. 2018, Art. no. 5485461.
[35] J. Huang, C. Zhang, and J. Zhang, "A multi-queue approach of energy efficient task scheduling for sensor hubs," *Chin. J. Electron.*, vol. 29, no. 2, pp. 242–247, Mar. 2020.
[36] J. Huang, S. Li, and Y. Chen, "Revenue-optimal task scheduling and resource management for IoT batch jobs in mobile edge computing," *Peer–Peer Netw. Appl.*, to be published.
[37] S. Li and J. Huang, "GSPN-based reliability-aware performance evaluation of IoT services," in *Proc. IEEE Int. Conf. Services Comput. (SCC)*, Jun. 2017, pp. 483–486.
[38] Los Alamos National Laboratory. (2011). *All Systems Failure/interrupt Data 1996–2005*. [Online]. Available: http://institute.lanl.gov/data/fdata/
[39] H. H. Ammar and S. M. R. Islam, "Time scale decomposition of a class of generalized stochastic Petri net models," *IEEE Trans. Softw. Eng.*, vol. 15, no. 6, pp. 809–820, Jun. 1989.
[40] Y. C. Ho, R. S. Sreenivas, and P. Vakili, "Ordinal optimization of DEDS," *Discrete Event Dyn. Syst.*, vol. 2, no. 1, pp. 61–88, Jul. 1992.
[41] J. Huang, C. Lin, X. Kong, B. Wei, and X. Shen, "Modeling and analysis of dependability attributes for services computing systems," *IEEE Trans. Services Comput.*, vol. 7, no. 4, pp. 599–613, Oct. 2014.
[42] T. W. E. Lau and Y. C. Ho, "Universal alignment probabilities and subset selection for ordinal optimization," *J. Optim. Theory Appl.*, vol. 93, no. 3, pp. 455–489, Jun. 1997.
[43] J. Yuan, Y. Zheng, C. Zhang, W. Xie, X. Xie, G. Sun, and Y. Huang, "T-drive: Driving directions based on taxi trajectories," in *Proc. 18th SIGSPATIAL Int. Conf. Adv. Geograph. Inf. Syst. (GIS)*, 2010, pp. 99–108.
[44] J. Yuan, Y. Zheng, X. Xie, and G. Sun, "Driving with knowledge from the physical world," in *Proc. 17th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2011, pp. 316–324.
[45] Y. Chen, J. Huang, C. Lin, and J. Hu, "A partial selection methodology for efficient QoS-aware service composition," *IEEE Trans. Services Comput.*, vol. 8, no. 3, pp. 384–397, May 2015.

JIWEI HUANG (Member, IEEE) received the B.Eng. and Ph.D. degrees in computer science and technology from Tsinghua University, in 2009 and 2014, respectively. He was a Visiting Scholar with the Georgia Institute of Technology. He is currently an Associate Professor and the Dean of the Department of Computer Science and Technology, China University of Petroleum, Beijing. He has published one book and more than 40 articles in international journals and conference proceedings, including the IEEE TRANSACTIONS ON SERVICES COMPUTING, the IEEE TRANSACTIONS ON CLOUD COMPUTING, ACM SIGMETRICS, IEEE ICWS, and IEEE SCC. His research interests include services computing, the Internet of Things, and performance evaluation.

JINGYU LIANG received the B.Eng. degree in digital media technology from Huaibei Normal University, in 2019. She is currently pursuing the master's degree with the Department of Computer Science and Technology, China University of Petroleum, Beijing. Her research interests include edge computing and services computing.

SIKANDAR ALI received the Ph.D. degree from the China University of Petroleum, Beijing. He is currently a Faculty Member and a Postdoctoral Fellow with the Department of Computer Science and Technology, College of Information Science and Engineering, China University of Petroleum. He has authored over 40 articles in highly-cited journals and conferences. His research interests include software outsourcing partnership, software testing and test automation, agile software development, and global software engineering. He received the Excellent Graduate Award in recognition of his excellent academic results and superior research for the Ph.D. degree. For the Ph.D. degree, he has received the Chinese Government Scholarship.