Received February 25, 2018, accepted April 7, 2018, date of publication May 15, 2018, date of current version June 5, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2836334

Bi-Objective Optimization for Energy Aware Internet of Things Service Composition

OSAMA ALSARYRAH[1], IBRAHIM MASHAL[2], AND TEIN-YAW CHUNG[1], (Member, IEEE)
[1]Department of Computer Science and Engineering, Yuan Ze University, Taoyuan 32003, Taiwan
[2]Computer Science Department, Aqaba University of Technology, Aqaba 77110, Jordan

Corresponding author: Tein-Yaw Chung (csdchung@saturn.yzu.edu.tw)

This work was supported by the Ministry of Science and Technology of China, Taiwan, under Contract MOST 106-2221-E-155-014.

ABSTRACT In recent years, service-oriented-based Internet of Things (IoT) has received massive attention from research and industry. Integrating and composing smart objects functionalities or their services is required to create and promote more complex IoT applications with advanced features. When many smart objects are deployed, selecting the most appropriate set of smart objects to compose a service by considering both energy and quality of service (QoS) is an essential and challenging task. In this paper, we reduced the problem of finding an optimal balance between QoS level and the consumed energy of the IoT service composition to a bi-objective shortest path optimization (BSPO) problem and used an exact algorithm named pulse to solve the problem. The BSPO has two objectives, minimizing the QoS including execution time, network latency, and service price, and minimize the energy consumption of the composite service. Experimental evaluations show that the proposed approach has short execution time in various complex service profiles. Meanwhile, it can obtain good performance in energy consumption and thus network lifetime while maintaining a reasonable QoS level.

INDEX TERMS IoT services, energy efficiency, service composition.

I. INTRODUCTION

Internet of Things (IoT) is a new technology paradigm that allow smart objects to be connected together to collaborate, cooperate, and communicate with each other to provide and support smart applications [1], [2]. Currently, there are more than 8 billion connected smart objects, and the number will continue to increase dramatically year after year [3]. Smart objects are heterogeneous in their functionalities, communication capabilities, and resources. In general, smart objects are resource constrained with very limited computation and storage capacities when it is a battery-powered device, e.g., wireless sensors and mobile phones.

With the advent and rapid development of service-defined everything, smart objects are represented as services corresponding to their co-hosted functions [4]–[6]. In other words, each IoT smart object provides its function through standard services that can be directly accessed. To this end, Service-Oriented Computing (SOC) [6] is seen as the key enabler for IoT.

Integrating and composing smart objects functions or their services is required to create and promote more complex IoT applications with advanced features [4]. Composing those services is done by aggregating atomic services to provide new functions that none of the services could provide individually [7]. This integration must consider the Quality of Service (QoS) and energy efficiency of the composed objects. Take a large-scale complex IoT environment as an example, when a smart object has low energy, it should be replaced with another smart object, if any, that has more energy and can provide the same functions and a good QoS level. However, this is a challenging task since IoT QoS values are dynamic and can substantially vary during the lifetime of the application when network states change. Moreover, smart objects can join, leave, fail, or new services with better quality can appear at any time. Therefore, finding a good balance between the energy consumption of all objects and its QoS to prolong the network lifetime is not a simple task.

In the past, several studies have discussed service discovery and composition and many techniques have been developed for Web services and Representational State Transfer (REST) [8]. However, those techniques only considered the functional and non-functional properties of services. As mentioned above, due to the nature of IoT and the smart objects, IoT services composition must consider not only QoS, but also power consumption and residual energy level [9].

There are a few studies on IoT service composition that address energy consumption [10]. However, none of them considered energy consumption as a separated objective; they considered both QoS and energy consumption as a single objective. Unlike previous studies, we deal with energy consumption of IoT services separately from other QoS attributes. In this way, both QoS level and energy consumption are treated equally with the same priority.

This study aims to model and develop a Bi-objective Shortest Path Optimization (BSPO) for IoT service composition and to find an optimal balance between QoS level and the consumed energy of the IoT service composition. The BSPO energy-aware IoT service composition has two objectives: minimize the QoS including execution time, network latency, and service price and minimize the energy consumption of the composite service. BSPO derives the optimal solution by finding a Pareto-optimal solution for QoS and energy-aware IoT service composition based on users or operators preferences.

The main contribution of this paper can be summarized as follows:

1) Investigate IoT service composition by considering the energy consumption along with QoS of selected services.
2) Formulate a novel optimization problem to maximize the QoS level and to minimize the energy consumption of composite service.
3) Propose a BSPO model and use the pulse algorithm to solve the formulated problem.
4) Simulate and evaluate the proposed service composition scheme, and compare its performance with other greedy algorithms.

The rest of the paper is organized as follows. Section II reviews the related work. Section III presents the service composition model. Section IV describes problem formulation. Section V depicts the optimization problem. Section VI explains the pulse algorithm. Section VII shows results and performance evaluation. Finally, Section VIII concludes the paper.

II. RELATED WORK

A. SERVICE-ORIENTED IOT

IoT adopts Service-Oriented Architecture (SOA) paradigm because it can provide cooperation between heterogeneous smart objects and is very flexibile for system integration. Accordingly, smart objects can be connected and composed via service composition approaches. Recently, many research works have abstracted IoT devices as a service to provide an efficient and unified way of accessing and operating IoT services [1], [11], [12].

The Web of Things (WoT) [13] has been proposed to seamlessly integrate heterogeneous objects. Through existing Web technologies such as Web services and RESTful interfaces, WoT enables objects to communicate and interact. Sun *et al.* [14] proposed a microservice IoT framework to provide a generic IoT architecture based on a module or multi-component application instead of the monolithic applications. The framework abstracts smart objects and IoT application modules as services.

Cheng *et al.* [15] proposed a platform for event-driven service-oriented IoT coordination, where SOA is adopted to solve interoperability issues among large numbers of heterogeneous services and physical entities in IoT. Another service-oriented IoT architecture is presented in [16], where the authors proposed a user-centric IoT-based service-oriented architecture to integrates services that utilize IoT resources in an urban computing environment. In [16], user goals are represented as an explicit task definition that is coordination of activities. Activities consist of configurations of abstract services that can be instantiated by orchestrating available service instances, including services that can be actuated through the IoT devices or composed of more than one smart object.

Other studies also focused on Wireless Sensor Networks (WSNs) to provide composite services by integrating its functions. Zhou *et al.* [4] proposed a three tiers service-oriented framework. The function of each sensor is abstracted as a service within a service class. Service classes are chained to fulfill the functional requirements and energy-efficiency. Another work adopted service-oriented WSN presented in [17].

B. SERVICE COMPOSITION AND QOS-AWARE SERVICES COMPOSITION

Service composition techniques are designed for relatively complex services when the required functions cannot be satisfied by any single service. It combines more than one service to fulfill the request. Traditional service composition techniques compose services in a specific order to meet the required goals [9]. In the literature, many techniques have been proposed for service composition based on its function such as semantic-based matchmaking, Logic, Graph-Theory, Petri net, and AI-Planning based [18].

QoS-aware services composition is known to be an NP-hard problem. However, this problem has already been addressed by several methods. Ngoko *et al.* [19] proposed a Mixed-Integer Linear Programming (MILP) method to solve QoS service composition. Furthermore, they considered energy consumption as a QoS attribute. Yu *et al.* [20] formulated the problem as a Multi-dimensional Multi-choice Knapsack Problem (MMKP). Llinás and Nagi [21] proposed a graph-based model to solve the problem as a Multi-Constraint Shortest Path problem (MCSP). Wu and Zhu [22] used a Directed Acyclic Graph (DAG) to model services composition as a path search problem. Several studies proposed Pareto optimality techniques for solving the QoS-based services selection problem [23], [24]. For example, Chen *et al.* [23] proposed a services composition algorithm using a partial selection approach. Based on dominance relation, this approach allows us to reduce the search space by pruning unpromising candidate services in QoS. However, they adopted service selection by local optimization, which only selects the best candidate service locally for each abstract service without considering the relation between tasks in the workflow. Unlike the work cited above, we considered the QoS for the entire workflow as an end to end service composition and used an exact method to solve the bi-objective optimal problem including QoS and energy consumption.

C. SERVICE COMPOSITION IN IOT CONTEXT

Many studies in QoS-aware service composition and selection problem are available in the literature. Bellido *et al.* [7] analyzed stateless compositions of RESTful services and its control-flow patterns. The researchers also presented a comparative evaluation of different QoS attributes. Dar *et al.* [25] addressed the problem of integrating IoT smart objects by adopting the concepts of centralized service composition (orchestration) and decentralized service composition (choreography).

Simple Additive Weighting (SAW) technique has been used to rank candidate IoT service in QoS. Kouicem *et al.* [26] and Yachir *et al.* [27] calculated a single utility value by aggregating the QoS values, where the service composition problem is transformed to a single objective optimization problem that finds the services with the best utility value. Jin *et al.* [24] proposed a three phases' service composition algorithm. The first phase pre-sorts services according to user's preferences. The second phase applies dominance-based filtering to eliminate sub-optimal solutions and the final phase sorts the rest of services to select the best service. Four QoS attributes are associated with each IoT. The candidate services are evaluated concerning user's requirements by aggregating each QoS rating (utility) function. However, studies above do not consider energy consumption.

In the literature, only a few studies considered both energy and QoS. For example, Khanouche *et al.* [10] proposed an IoT energy-centered and QoS-aware services selection and composition. The proposed selection approach preselects the services offering the QoS level required for user's satisfaction using a lexicographic optimization strategy and QoS constraints relaxation technique. The concept of relative dominance of services is also proposed. However, the preselecting phase aggressively prunes services that don't meet the local QoS requirements or affect the quality of end to end QoS level when considering execution time and network latency.

Unlike the aforementioned studies, our work deals with the energy efficiency of services separately from other QoS attributes and takes into account both QoS level and energy consumption of IoT services. The proposed selection approach transformed the problem into a bi-objective optimization problem that aims to minimize the amount of energy consumed and maximize the QoS of a service. We solved the optimization problem using bi-objective shortest path algorithm with four online pruning techniques to reduce the complexity of the algorithm. In addition, to ensure a good balance among the candidate services in energy consumption, the services consuming less energy are selected. Our model aims to provide high availability of services by minimizing the energy consumption, i.e., by maximizing the devices battery lifetime.

To the best of our knowledge, there exists no previous study that considers an end to end service composition scheme with both QoS and the energy efficiency as the primary metrics for IoT resources.

III. IOT SERVICE COMPOSITION MODEL

IoT applications consist of a set of decomposed services. In our model, we define two types of services: an Abstract Service (AS) represents function of a service provided by one or more IoT devices or software modules, while a Candidate Service (CS) represents a real Web service that may be invoked. All cs are distributed across available resources.

The candidate services are characterized by two types of properties: functional and non-functional properties. Functional properties indicate the actions and the functions provided by a cs, while non-functional properties are defined by the QoS attributes and the energy profile of the service running at a battery-powered object. Here, we considered three QoS attributes, namely, execution time, cost, and energy consumption profile.

From a user's perspective, when a user's request arrives at a service composition broker, the service composition process is invoked. The process combines a series of atomic service components appropriately to form a composite service (path) that provides an optimal balance between QoS and consumed energy.

Fig.1 shows a model for composing IoT services. Suppose an IoT service consists of n tasks, $\{T_i,\ 1 \leq i \leq n\}$, which are needed by a given IoT service requested by an end user. For each task T_i a corresponding abstract service, AS_i, is used to represent its functional requirement. For each AS_i, $1 \leq i \leq n$, , there exists l_i candidate services, $\{CS_{i,j},\ 1 \leq j \leq l_i,\}$ that can meet the functional requirements of AS_i and thus can be selected for realizing this service component. Note that a service component may be either an IoT service or a software component. To meet the requirements of a specific service request, the service composition path is constructed as a path from a service entrance portal (CS_0) to a service exit portal (CSe) by traversing only one candidate service of each service component. For example, $CS_0 \rightarrow CS_{1,3} \rightarrow CS_{2,5} \rightarrow \cdots \rightarrow CS_{m,j} \rightarrow CS_e$ is a composite service path that may provide useful service to a user. Therefore, the main focus for IoT services composition is to select an optimal service sequence from a pool of available smart objects and software components while satisfying various QoS requirements, in addition to the amount of consumed energy.

IV. PROBLEM FORMULATION

Let $T = \{T_1, T_2, T_j, \ldots\ldots, T_\mathrm{n}\}$ denote the set of tasks that cover the composite IoT service, where n is the total number of decomposed tasks and T_j is the jth (j = 1, 2, 3, n) sub task

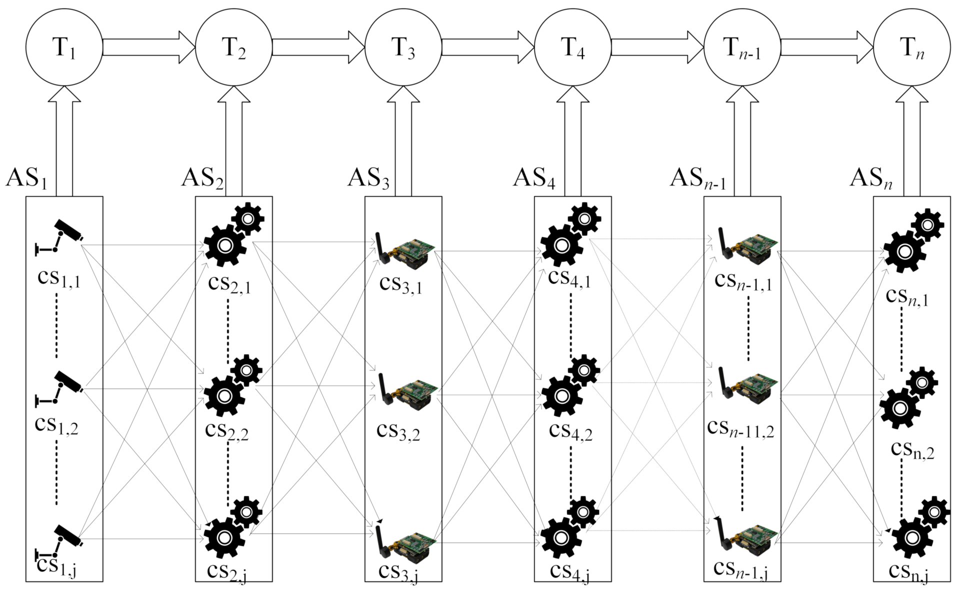

FIGURE 1. Service composition model.

of T. Let $AS_i = \{cs_{i1}, cs_{i2}, cs_{ij}, cs_{im_i}\}$ be the candidate services available for task T_j, where m_i represents the number of candidate services and cs_{ij} is the jth candidate service. Thus, the directed graph model represents all possible compositions for an IoT service as shown in Fig.1.

A distributed IoT service formed by interconnecting n different service components can be modeled as a directed graph $G = (N, A)$, where $N = \{v_1, \ldots, v_i, \ldots, v_n\}$ denotes a set of service components with n nodes and $A = \{e_{ij} | v_i, v_j \in N\}$ is the set of edges (links). Each node v_i is associated with a weight c_i representing its functional capability of abstract service AS_i, $1 \leq i \leq n$. Each edge $e_{ij} \in A$ are associated with two nonnegative weights $QoSU_{ij}$ and EP_{ij}, where $QoSU_{ij}$ and EP_{ij} denote the utility value of QoS attributes and the consumed energy when traversing e_{ij}. Henceforth, without loss of generality, $QoSU_{ij}$ refers to QoS attributes such as execution time, network latency and cost. The objective of service composition is to select one candidate service from each set AS_j and generate an optimal IoT Composition Service Path (CSP) $x = \{x^1, x^2, \ldots\ldots, x^i, \ldots\ldots\ldots, x^{\mathrm{n}}\}$ from the set of available compositions under multi-objective requirements, where x^i denotes the selected candidate service for sub task T_i.

A. ENERGY PROFILE MODEL (EP)

Since energy consumption is a significant factor for devices hosting the candidate service, it is considered necessary that each candidate service provides the composer its energy consumption variable so that the composer can select the most energy efficient one.

We defined the energy profile (EP) of cs_{ij} by two variables, the Residual Energy level $RE(cs_{ij})$ of the device hosting cs_{ij} and the Consumed Energy $CE(cs_{ij})$ which represents the consumed energy when running cs_{ij}. $RE(cs_{ij})$ is estimated as follows.

$$RE(cs_{ij}) = CDE(cs_{ij}) - E_{th}(cs_{ij}) \tag{1}$$

where $CDE(cs_{ij})$ represent current energy level of the battery-powered device hosting cs_{ij} and $E_{th}(cs_{ij})$ the energy threshold value under which the device cannot support cs_{ij} anymore.

Since our model is based on service-oriented computing, the consumed energy of running cs_{ij}, $CE(cs_{ij})$, is calculated as in Eq. (2). Here, we assumed that the energy consumption of cs_{ij} is constant since the service runs on the same platform, uses the same resources, and receives and sends the same amount of data.

$$CE(cs_{ij}) = ECR\left(cs_{ij}\right) * T(cs_{ij}) \tag{2}$$

where $ECR\left(cs_{ij}\right)$ represents the energy consumption rate, and $T(cs_{ij})$ represent the execution time of cs_{ij}. The energy profile of cs_{ij} is calculated as shown in Eq. (3) by taking the ratio between the energy consumed by invoking cs_{ij} and its residual energy.

$$EP(cs_{ij}) = CE(cs_{ij})/RE(cs_{ij}) \tag{3}$$

Thus, the smaller is $EP(cs_{ij})$ the better is the cs_{ij} as an candidate for T_i.

Finally, the EP of a service composition x can be calculated as shown in Eq. (4), the energy consumed by composition path x^i is:

$$EP\left(x\right) = \sum_{i=1}^{n} EP\left(x^i \middle| x^i \in S^h\right) \tag{4}$$

where S^h represents the set of battery-powered devices. In our model, we consider the differences in the underline infrastructure of the running services. In general, IoT software services are not executed on battery powered devices, in contrast to sensing and actuating services. To differentiate between these two types of required services we refer to the set of battery-powered devices as S^h.

B. QOS CRITERIA

In our study, we consider a set of quantitative non-functional properties of IoT services which can be used to describe the quality criteria of a Web service. For the sake of simplicity, in this paper, we consider only negative attributes as our non-functional properties. These values of negative attributes need to be minimized. We included two attributes: service execution time (T), which is collected from records of previous execution monitoring, and service cost (C), which is directly collected from service providers.

1) SERVICE EXECUTION TIME (T)

Service execution time represents the average time expected for executing a candidate service. The service provider updates it continuously because the load of the device hosting the service changes dynamically. Clients expect their jobs to be completed in a minimal time when they submit requests to the service composer. Let $L(cs_{ij})$ be the latency of transmitting data from cs_{ij} and $ET\left(cs_{ij}\right)$ be the execution time of service request for cs_{ij}. Thus, the service execution time $\mathrm{T}(cs_{ij})$ can be computed as in Eq. (5).

$$T(cs_{ij}) = L(cs_{ij}) + ET(cs_{ij}). \tag{5}$$

Therefore, the service execution time for composition path x can be given as follows.

$$T\,(x) = \sum_{i=1}^{n} T(x^i) \tag{6}$$

2) SERVICE COST (C)

When a user submits his/her request to a service composer, the composer manages and finds the fastest composition path for the request. Meanwhile, the user is also expected to pay the fairest price for running his/her tasks. Therefore, the service cost is considered a valuable QoS property. In this model, we set $C(cs_{ij})$ as the cost of executing cs_{ij}. The cost is usually fixed but may be changed according to the service provider's business policy. The execution cost is registered by the service provider.

Therefore, as given in Eq. (7), the service cost for composition path x is:

$$C\,(x) = \sum_{i=1}^{n} C\left(x^i\right) \tag{7}$$

3) UTILITY FUNCTION

Utility function is a helpful mechanism for evaluating the aggregated quality of a given composite service. In this research, we calculate the utility value of a service composition by aggregating normalized QoS attributes values. All QoS values are mapped to a single real value between 0 and 1 by comparing the QoS value with the minimum and maximum available QoS value. This enables uniform evaluation of the QoS value.

We adopted a Multiple Attribute Decision-Making (MADM) approach, namely, the Simple Additive Weighting (SAW) technique [28] for the mapping process. For a composite service path x the aggregated QoS values are compared with the minimum and maximum possible aggregated values. The minimum (or maximum) possible aggregated values can be easily estimated by aggregating the minimum (or maximum) value of each service class. The utility function of QoS is computed as in Eq. (8).

$$QoSU\,(x) = w_t * \acute{T}\,(x) + w_c * \acute{C}\,(x) \tag{8}$$

where w_t and w_c represent the weighting factors of execution time and cost, respectively. The sum of weights are equal to one, i.e., $w_t + w_c = 1$. $\acute{T}\,(x)$ is the normalized service execution time for x that is calculated using Eq. (6). $\acute{C}\,(x)$ is the normalized service cost for x that is calculated using Eq. (7).

With the above QoS utility and power profile formulas, we can formulate the optimization problem of the service composition as a BSPO problem. We described the BSPO problem in the next section.

V. BI-OBJECTIVE OPTIMIZATION PROBLEM

Multi-objective service composition selects one candidate service from each set AS_j and generates an optimal IoT CSP from the set of available compositions under multi-objective and constraints. In this study, we formulated the energy aware IoT service composition as BSPO for finding an optimal IoT CSP from the start node $cs_s \in N$ to the end node $cs_e \in N$ that minimizes two different (often conflicting) objective functions. The bi-objective shortest path problem can be formally defined as in Eq. (9).

$$\begin{aligned} &\min CSP(x) = (QoSU(x), EP(x)) \\ &\text{s.t.,}\ \ x \in X \end{aligned} \tag{9}$$

where $\boldsymbol{x}$ represents a candidate service path from the service entrance portal from cs_s to the service exit portal cs_e. $QoSU(x)$ represents the aggregated value of the QoS values along all edges of a path x. $EP\,(x)$ represents the aggregated value of the EP over all edges in x. X is the set of all paths from cs_s to cs_e. The objective of Eq. (9) is to minimize the QoS utility value and the energy profile of CS $P(x)$. Since the existence of a path that simultaneously minimizes both objectives in Eq. (9) cannot be guaranteed, we seek for a set of paths with an acceptable tradeoff between the two objectives.

IoT service composition is a NP-complete problem with $\prod_{j=1}^{n} M_j$ possible CSP for task T, where n represents the number of tasks and M_i represents the number of candidate services for task T_i. To solve the above problem, in this paper, we use the pulse algorithm detailed in the next section.

VI. THE PULSE ALGORITHM: OVERVIEW

We used the pulse algorithm [29] to solve the bi-objective optimization problem. The pulse algorithm optimizes a bi-objective function CSP(x) that is composed of a quality of service function $QoSU$ and energy profile function EP. A path x optimizes CSP or is Pareto-optimal if there is no other path x' that has lower $QoSU$ and lower EP than x. The goal of the pulse algorithm is then to find a series of such Pareto-optimal paths that together form an efficient set X_E by recursively examining the entire search space of the graph.

The efficiency of the algorithm is achieved by aggressively pruning partial paths. When a pulse reaches to a newly added node, it will check if adding the node to the existing partial path satisfies one of the adopted pruning conditions or not. The partial path will be eliminated if it meets one of the following conditions:

1) It includes cycles.
2) It exceeds either one or both upper bounds obtained at the initialization phase, represented by a nadir point, before reaching the end node.
3) It is dominated by any other path in the current efficient set before reaching the end node.
4) It is dominated by any objective value stored in the label of the newly added node. A node is labeled by accumulated objective values when it is traversed by a feasible path. Thus, if the partial path is dominated by the existing label of the newly added node, it will not be part of a Pareto optimal path.

Suppose that we have a given network with start node v_s and end node v_e. The pulse algorithm sends a pulse from v_s to v_e. This pulse travels through the entire network while storing the partial path p (an ordered sequence of visited nodes) and its cumulative objective functions, $QoSU\ (p)$ and $EP(p)$. Every pulse that reaches the end node v_e is a feasible solution that might be efficient. Once a pulse reaches the end node, it recursively backtracks to continue its propagation through the rest of the nodes in the search for more efficient paths from v_s to v_e. If the pulse is let free, this recursive algorithm identifies all possible paths, and guarantees that an efficient set is always found.

However, the pulse algorithm does not continue exploring any partial path that will not produce an efficient solution by using a look-ahead mechanism that prunes aggressively vast regions of the solution space. For the initialization procedure, the algorithm starts running a mono-objective shortest path algorithm to get the upper bound for each objective. The pulse algorithm is shown in Algorithm 1.

The pulse algorithm follows a depth-first search truncated by several pruning strategies to control the pulse propagation and prunes pulses without cutting off any efficient solution. The algorithm defines four pruning strategies namely, pruning by cycles, nadir point, efficient set, and label. The pulse recursive function is shown in Algorithm 2 . It takes four input parameters, current node v_i, the cumulative QoS utility value $QoSU(p)$, the cumulative energy profile $EP(p)$, and the partial path p. The pruning strategies applied to the pulse are

Algorithm 1 Pulse Algorithm

```
Input: $G$ directed graph; $v_s$: start node; $v_e$ end node
Output: $X_E$: true efficient set
1: $p \leftarrow \{\}$
2: $QoSU(p) \leftarrow 0$
3: $EP(p) \leftarrow 0$
4: $initialization\ (G)$
5: $pulse(v_s, QoSU\ (p)\ , EP\ (p)\ , p)$
6: return $X_E$
```

Algorithm 2 Pulse Function:
$pulse\ (v_s, QoSU\ (p)\ , EP\ (p)\ , p)$

```
Input: $v_i$, $current\ node$; $QoSU\ (p)$,
$cumalative\ QoS\ utility$; $EP\ (p)$,
$cumulative\ power$; $p$, $current\ path$.
Output: void
1: if isAsyclic($v_i$, $p$) then
2:   if checkNadirPoint($v_i$, $QoSU\ (p)\ , EP\ (p)$) then
3:     ifcheckEfficientSet($v_i$, $QoSU\ (p)\ , EP\ (p)$) then
4:       ifcheckLabels($v_i$, $QoSU\ (p)\ , EP\ (p)$) then
5:         store ($QoSU\ (p)\ , EP\ (p)$)
6:         $p \leftarrow \grave{p} \cup \{v_i\}$
7:         for $v_j \in \Gamma^+(v_i)$ do
8:           $qos\ (\grave{p}) \leftarrow QoSU\ (p) + QoSU(cs_{ij})$
9:           $EP\ (\grave{p}) \leftarrow EP\ (p) + EP(cs_{ij})$
10:          $pulse(v_j, QoSU\ (\grave{p})\ , EP\ (\grave{p})\ , \grave{p})$
11:        end for
12:      end if
13:    end if
14:  end if
15: end if
16: return void
```

shown in Lines 1–4; if the pulse is not pruned, line 5 stores the current $QoSU(p)$ and $EP(p)$ and line 6 adds the node v_i to the partial path. In lines 7–11, the pulse propagates over all nodes $v_j \in \Gamma^+(v_i)$, where $\Gamma^+(v_i)$ is the set of outgoing neighbors of v_i, and adds current QoS utility to the cumulative one and current EP to the cumulative EP.

Whenever the pulse function is invoked at end node v_e, a partial path P becomes a complete solution x and we update the online efficient set $\widehat{X_E}$. Note that the information about X_E has a global scope and is not an attribute of the traveling pulse within the recursion. Algorithm 3 presents the pulse function when it is invoked on end node v_e. Since a new solution has been found, the algorithm verifies if the new solution is efficient and updates the online efficient set accordingly.

A. PRUNING TECHNIQUES

1) PRUNING BY CYCLES

Because all weights on the arcs are nonnegative, any efficient solution cannot contain cycles. To avoid cycles in a path, every time we invoke the pulse function at v_i, the algorithm

Algorithm 3 Pulse Function for the End Node: $pulse\left(v_e, QoSU\left(p\right), EP\left(p\right), p\right)$

```
Input: $v_e$, current node; $QoSU(p)$,
cumalative QoS utility; $EP(p)$,
cumulative power; $p$, current path.
Output: void
1: if CheckEfficientSet($v_e, QoSU(p), EP(p)$) then
2:   $p \leftarrow p \cup \{v_e\}$
3:   $X \leftarrow$ mapPathToSolution($p$)
4:   UpdateEfficentSet(X)
5: end if
6: return void
```

checks whether a node has been visited or not. If v_i has already lain on the partial path, it is pruned from P.

2) PRUNING BY NADIR POINT

Based on the idea of the nadir point which seen as the anti-ideal point in the objective space, the algorithm aims to prune as early as possible any pulse exceeding either smallest values (best path) of both objectives solutions. To do so, we calculate the minimum *QoSU(x)* (regardless of EP) and the minimum energy profile *EP(x)* (regardless of QoS) from entrance node to the end node.

Assume that we have an optimal solutions x^*_{QoSU} and x^*_{EP} where x^*_{QoSU} and x^*_{EP} represent Energy profile and QoS utility objectives of the mono-objective shortest path problem, respectively. The images for the optimal solutions in the objective space are $Z\left(x^*_{QoSU}\right) = (\overline{EP}, \underline{QoSU})$ and $Z\left(x^*_{EP}\right) = (\underline{EP}, \overline{QoSU})$. The nadir point, denoted by $Z^N = (\overline{EP}, \overline{QoSU})$, which represents a vector composed of the worst objective values in the objective space. In other words, it represents an upper bound for each objective in the objective space.

After applying the mono objective shortest path problem for each objective, a set of an alternative optimal solution for each objective can be found. $\overline{EP}$ and $\overline{QoSU}$ represent the smallest values among all alternative solutions for both objectives x^*_{QoSU} and x^*_{EP}, respectively. As shown in Fig.2, $Z\left(x^*_{QoSU}\right)$, $Z\left(x^*_{EP}\right)$, and Z^N are shows the minimizer victors in the objective space and Z* represents the ideal point.

Based on this, for any solution x with $QoSU\left(x\right) > \overline{QoSU}$ or $\mathrm{EP}\left(x\right) > \overline{EP}$, its image z($x$) is dominated and x is not efficient. Any point falls in the dark region in Fig.2, is eather dominated by $Z\left(x^*_{QoSU}\right)$ or $Z\left(x^*_{EP}\right)$, and thus any pulse exceeding either $\overline{EP}$ and $\overline{QoSU}$ will be pruned.

3) PRUNING BY EFFICIENT SET

Consider the online efficient set $\widehat{X_E}$ at a given intermediate stage of the algorithm. Using the lower bound found in the initiation phase of the algorithm namely, $\underline{QoSU}(v_i)$ for QoS utility and $\underline{EP}\left(v_i\right)$ for energy profile, we can determine

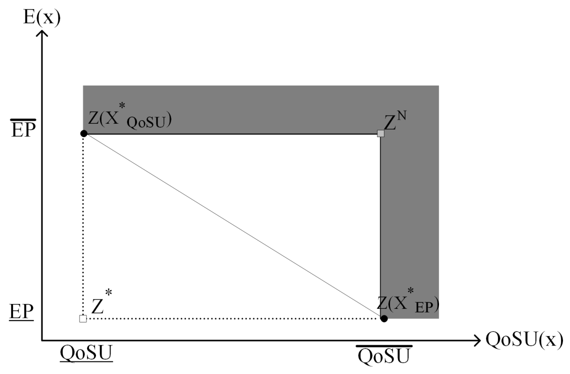

FIGURE 2. Nadir point and lower and upper bounds.

whether a partial path will become an efficient solution or not. Given a partial path p to node v_i, if there is a solution $x \in \widehat{X_E}$ such that $QoSU\left(p\right) + \underline{QoSU}(v_i) \geq QoSU(x)$ and $EP\left(p\right) + \underline{EP}(v_i) \geq EP(x)$, we can safely prune partial path p, because even if it spends both the minimum cost and the minimum time to reach the end node, it will still be dominated by path x.

4) PRUNING BY LABEL

For each node v_i a fixed number of labels saves a tuple of *QoSU* and *EP* values. The labels at node v_i are denoted by $\mathrm{L}\left(\mathrm{v}_i\right) = \left\{(QoSU_{il}, EP_{il}) \middle| l = 1, \ldots, \mathrm{Q}\right\}$ where $QoSU_{il}$ and EP_{il} are the cumulative QoS utility and energy profile for a partial path to v_i and Q denotes the number of labels at v_i. For an incoming pulse, the algorithm checks if the incoming partial path p is dominated or not; that is, if any label dominates $CSP(p)$, the pulse is discarded by label pruning.

VII. PERFORMANCE STUDY

In this section, we present the setup of our simulation. Then, we analyz the performance of the exact bi-objective algorithm for IoT service composition that optimizes QoS utility, and consumed energy.

To show how IoT services composition can be instantiated and invoked while keeping an optimal balance between QoS level and the consumed energy of the composed service, we conducted an extensive simulation under various scenarios to evaluate the performance of the proposed IoT service composition. In these scenario we assumed that we have a smart environment consist of thousands of heterogeneous objects such as mobile devices, and wireless sensors. We also assumed that these devices are heterogeneous in communication protocols. For example, wireless sensors can be based on ZigBee, 6lowpan or Bluetooth [2]. To provide interoperability between these heterogeneous objects, their functions and their software components are abstracted as services to become accessible through SOP, Constrained Application Protocol (CoAP) or REST protocols [2]. Finally, we assumed that all candidate services are registered in IoT orchestration system and categorized based on its services classes.

A. SIMULATION ENVIRONMENT AND METHODOLOGY

We implemented our simulation using java under 64-bit Windows seven operating system, running on Intel Core i5-2500, 3.3 GHz, and 8 GB RAM. Each scenario generated a different number of abstract services AS and candidate services *CS*. Each candidate service has two QoS attributes: execution time including network latency, and service cost.

Due to the absence of datasets in QoS and energy profile values of IoT services, we chose to evaluate the proposed algorithm by using synthetically generated data. For instance, each service is produced with a random QoS value according to the values reported in the literature [10]. Based on the model presented in [29], we specified the energy profile of IoT services. The values of each quality parameter are generated according to a normal distribution. The importance or weight of each QoS attribute in the utility function is set to be fixed 1/2. The service execution time and network latency are generated assuming a uniform distribution over the interval [20, 1200] and [20, 800]. The cost of services is generated according to a uniform distribution over the interval [10, 20].

In order to overcome the problem of QoS values fluctuation during service runtime in dynamic IoT environments, QoS values are randomly changed after every service iteration by multiplying every QoS value with a random number in the interval [0.9, 1.1].

In order to study the energy consumption in battery operated objects, we referred to the energy model presented in [29]. We assumed that every device has an initial amount of charge and maximum battery charge, *Cinitial* and *Cmax*, where the values of *Cinitial* is chosen randomly in the interval [0.7 *Cmax*, 1.0 *Cmax*] and the maximum battery charge *Cmax* of an object is set as 1500 mA · h. Furthermore, any object has energy lower than *CThreshold* becomes unable to provide its services and will not be considered in the composition process. In this study, we set *CThreshold* to be 30% of *Cmax*. After every run of the selected candidate service, a specific amount of power is consumed and this amount is subtracted from current battery level of the device hosting the service. The amount of consumed power is chosen randomly in the interval [100 mA.s, 10000 mA.s] after every service invocation.

In our study, a service composer is used to find an optimal balance between QoS and consumed energy and prolong the network life time. In the following sections we will refer to the proposed algorithm by Bi-Objective Service Composition (BOSC). In order to show the added value of the proposed selection approach in a large-scale IoT services environment, we compared our results with two variants of our algorithm: QoSC, where only the QoS is taken into account in the selection process, and EPC, where only the power profile is taken into account in the selection process. The results presented here are derived based on the average of 100 simulations.

B. SIMULATION RESULTS

To evaluate the performance of the proposed model and algorithm we considered the following metrics: (1) Selection time which represents the computational time of the selection algorithm; (2) Energy consumption of the composite service which is equal to the total energy consumed by its components; (3) Composition lifetime which is the number of compositions that can be executed before the first candidate service failure. A service is considered failed when its autonomy is no longer sufficient to be invoked; (4) Optimality which is the ratio between the QoS value of the composite service obtained by BOSC and the optimal QoS value of the composite service, obtained by that of QoSC and EPC.

1) SELECTION TIME VERSUS NUMBER OF SERVICES

To validate the scalability of BOSC, we tested the execution time of the selection algorithm under various numbers of tasks involved in the composition process and various numbers of available candidate services for each task.

In the first experiment, we set the number of tasks involved in the composition to be 10, 15 and 20 tasks. We also set the number of candidate services for each task to be between 100 and 1000. Fig. 3 compares the average execution time (in millisecond or *ms*) of the composition algorithm with various numbers of tasks and candidate services for each task. As shown in Fig. 3, the average execution time increases as the number of tasks of the composite service increases. As shown in the figure, the average execution time is short and suitable for a large scale IoT environment. For example, the selection time does not exceed 10 *ms* when running 10 tasks each with 1000 candidate services. When increasing the number of tasks to 20 each with 1000 candidate services, the average execution time increases slightly to reach less than 70 *ms*. However, this increase is still reasonable and acceptable.

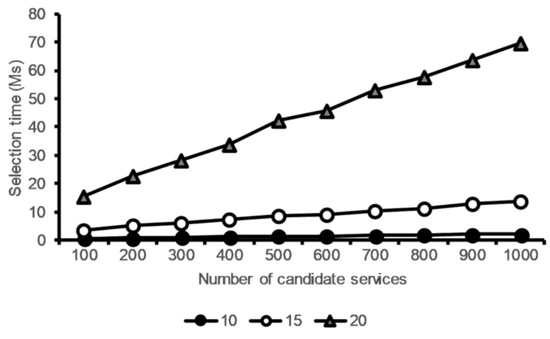

FIGURE 3. Selection time versus number of candidate services (10, 15, 20 tasks).

2) ENERGY CONSUMPTION VERSUS NUMBER OF SERVICES

In the second simulation, we compared the performance of the three algorithms in consumed energy (in mA.s). In the simulation, we considered a composition path consisting of 10 tasks where each task has 100 to 1000 candidate services. As shown in Fig. 4, the amount of consumed energy in

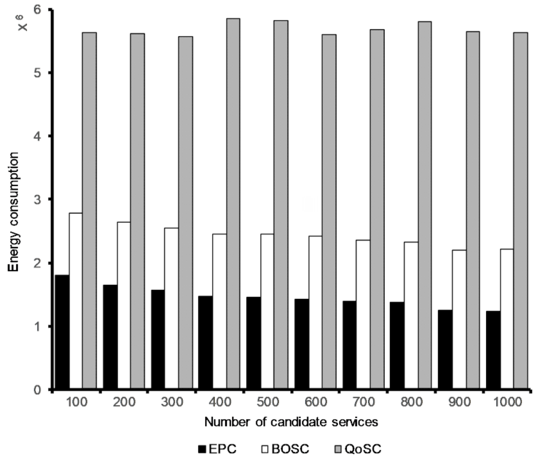

FIGURE 4. Energy consumption versus number of candidate services.

BOSC and EPC gets close to each other. Note that in EPC the candidate services with the lowest energy profile are always selected.

From Fig. 4, we can see that when candidate services are between 100 and 1000, the amount of consumed energy by BOSC is about 35% more than that by EPC in average. Certainly, EPC provides the best composite service in energy consumption. An interesting observation is that the amount of consumed power decreases when the number of the available services increases. This can be explained by the fact that when increasing the number of candidate services the probability of selecting more services with less power consumption increases, and hence, the energy consumed decreases. Another advantage of BOSC is that it saves more power than that by QoSC. BOSC consumed energy 70% less than that by QoSC.

In the third experiment, we intended to show the consumed energy under various numbers of tasks between 10 and 50 tasks when the number of candidate services for each task are set to be 100. Fig. 5 shows that the energy consumption of the composite path increases when the number of service classes increases. In fact, increasing the size of composite path will increase the number of selected services which causes more power consumption.

3) COMPOSITION LIFETIME VERSUS NUMBER OF CONCRETE SERVICES

Studying the performance of the proposed algorithm in terms of service composition life time is the aim of the fourth experiment. We evaluated and compared the composition lifetime by BOSC with that by EPC and QoSC. In the simulation, we considered a composition path consisting of 10 tasks and each task has 100 to 1000 candidate service. As shown in Fig. 6, the composition life time with BOSC is slightly less than that by EPC when only the lowest energy profile is selected. On the other hand, EPC guarantees the lowest energy consumption while reducing the *QoSU* of the composite path. Indeed, BOSC can achieve a good balance between

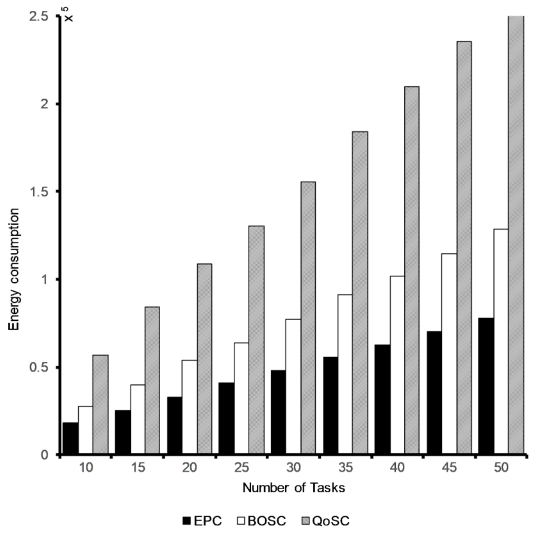

FIGURE 5. Energy consumption versus number of tasks.

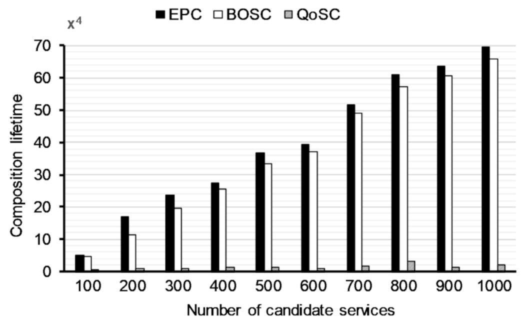

FIGURE 6. Composition lifetime versus number of candidate services.

the amount of consumed energy and *QoSU* of the composed path. Thus, BOSC ensures a long composition life time, which provides a high availability of candidate services.

4) OPTIMALITY OF THE SOLUTION

Studying the performance of BOSC in terms of optimality of the obtained *QoSU* is the purpose of this experiment. In the simulation we considered a composed path consisting of 10 tasks and each task has 100 to 1000 candidate services. As shown in Fig. 7, the optimality of the proposed method

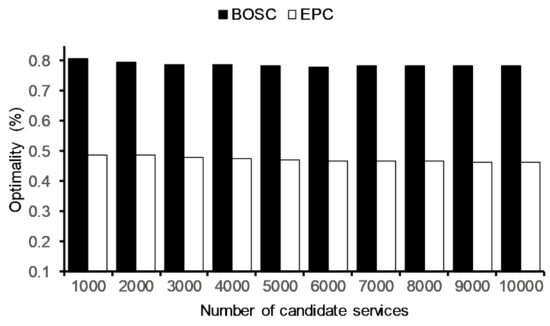

FIGURE 7. Optimality QoS versus number of candidate services.

guaranteed a QoS level about 80% close to that acquired by QoSC. It is worth noting that BOSC does not apply any constraint on QoS attributes in the simulation. The results of BOSC can be further improved if we apply some constraints on QoS attributes, such as execution time and cost thresholds. This is because it helps to reduce the solution choices with lower *QoSU* and hence better QoS can be achieved. As we can observe from Fig. 7, the QoS optimality level of EPC dramatically decreases, while BOSC can provide a solution close to optimal solutions.

VIII. CONCLUSION

Service-oriented IoT has received considerable attention over the past few years. A crucial factor for the success of IoT and its applications is creating more complex IoT applications with advanced features by composing smart objects functions and services.

In this paper, a bi-objective shortest path optimization model is presented to model IoT service composition where energy consumption and QoS are considered. The pulse algorithm with four embedded pruning techniques, namely, pruning by cycle, nadir point, efficient set, and label, is developed to solve efficiently the presented problem. Results show that our proposed IoT service composition scheme overcomes and surpasses other schemes that only consider QoS or power consumption individually. Experiments also show that the proposed scheme works reasonably fast in selecting suitable smart objects; the average execution time needs less than 70 ms, which makes the proposed model scalable for large-scale IoT environments. The amount of consumed energy by BOSC is about 35% more than that consumed by EPC on average. The composition lifetime with BOSC is 90% more than that by the QoS only scheme. Also, the acquired optimality level of the BOSC guaranteed a QoS level about 80% close to that obtained by QoSC. Therefore, the proposed solution provides an optimal balance between QoS level and consumed energy in IoT service composition.

REFERENCES

[1] I. Mashal, O. Alsaryrah, T.-Y. Chung, C.-Z. Yang, W.-H. Kuo, and D. P. Agrawal, "Choices for interaction with things on Internet and underlying issues," *Ad Hoc Netw.*, vol. 28, pp. 68–90, May 2015.

[2] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of Things: A survey on enabling technologies, protocols, and applications," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 4, pp. 2347–2376, 4th Quart., 2015.

[3] I. Gartner. (Feb. 12, 2017). *Gartner Says 8.4 Billion Connected 'Things'*. [Online]. Available: https://www.gartner.com/newsroom/id/3598917

[4] Z. Zhou, D. Zhao, L. Liu, and P. C. K. Hung, "Energy-aware composition for wireless sensor networks as a service," *Future Gener. Comput. Syst.*, vol. 80, pp. 299–310, Mar. 2018.

[5] X. Xu, Q. Z. Sheng, L. J. Zhang, Y. Fan, and S. Dustdar, "From big data to big service," *Computer*, vol. 48, no. 7, pp. 80–83, 2015.

[6] Z. Wen, R. Yang, P. Garraghan, T. Lin, J. Xu, and M. Rovatsos, "Fog orchestration for Internet of Things services," *IEEE Internet Comput.*, vol. 21, no. 2, pp. 16–24, Feb. 2017.

[7] J. Bellido, R. Alarcón, and C. Pautasso, "Control-flow patterns for decentralized RESTful service composition," *ACM Trans. Web*, vol. 8, no. 1, pp. 1–30, 2013.

[8] M. Garriga, C. Mateos, A. Flores, A. Cechich, and A. Zunino, "RESTful service composition at a glance: A survey," *J. Netw. Comput. Appl.*, vol. 60, pp. 32–53, Jan. 2016.

[9] T. Baker, M. Asim, H. Tawfik, B. Aldawsari, and R. Buyya, "An energy-aware service composition algorithm for multiple cloud-based IoT applications," *J. Netw. Comput. Appl.*, vol. 89, pp. 96–108, Jul. 2017.

[10] M. E. Khanouche, Y. Amirat, A. Chibani, M. Kerkar, and A. Yachir, "Energy-centered and QoS-aware services selection for Internet of Things," *IEEE Trans. Autom. Sci. Eng.*, vol. 13, no. 3, pp. 1256–1269, Jul. 2016.

[11] V. Issarny, G. Bouloukakis, N. Georgantas, and B. Billet, *Revisiting Service-Oriented Architecture for the IoT: A Middleware Perspective*. Cham, Switzerland: Springer, 2016, pp. 3–17.

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

[13] D. Guinard, V. Trifa, S. Karnouskos, P. Spiess, and D. Savio, "Interacting with the SOA-based Internet of Things: Discovery, query, selection, and on-demand provisioning of Web services," *IEEE Trans. Serv. Comput.*, vol. 3, no. 3, pp. 223–235, Jul./Sep. 2010.

[14] L. Sun, Y. Li, and R. A. Memon, "An open IoT framework based on microservices architecture," *China Commun.*, vol. 14, no. 2, pp. 154–162, 2017.

[15] B. Cheng, D. Zhu, S. Zhao, and J. Chen, "Situation-aware IoT service coordination using the event-driven SOA paradigm," *IEEE Trans. Netw. Serv. Manag.*, vol. 13, no. 2, pp. 349–361, Jun. 2016.

[16] I.-Y. Ko, H.-G. Ko, A. J. Molina, and J.-H. Kwon, "SoIoT: Toward a user-centric IoT-based service framework," *ACM Trans. Internet Technol.*, vol. 16, no. 2, pp. 1–21, 2016.

[17] S. Y. Shah, B. K. Szymanski, P. Zerfos, and C. Gibson, "Towards relevancy aware service oriented systems in WSNs," *IEEE Trans. Services Comput.*, vol. 9, no. 2, pp. 304–316, Mar. 2016.

[18] Q. Z. Sheng, X. Qiao, A. V. Vasilakos, C. Szabo, S. Bourne, and X. Xu, "Web services composition: A decade's overview," *Inf. Sci.*, vol. 280, pp. 218–238, Oct. 2014.

[19] Y. Ngoko, A. Goldman, and D. Milojicic, "Service selection in Web service compositions optimizing energy consumption and service response time," *J. Internet Services Appl., J. Art.*, vol. 4, no. 1, p. 19, Nov. 2013.

[20] T. Yu, Y. Zhang, and K.-J. Lin, "Efficient algorithms for Web services selection with end-to-end QoS constraints," *ACM Trans. Web*, vol. 1, no. 1, p. 6, 2007.

[21] G. A. G. Llinás and R. Nagi, "Network and QoS-based selection of complementary services," *IEEE Trans. Serv. Comput.*, vol. 8, no. 1, pp. 79–91, Jan. 2015.

[22] Q. Wu and Q. Zhu, "Transactional and QoS-aware dynamic service composition based on ant colony optimization," *Future Gener. Comput. Syst.*, vol. 29, no. 5, pp. 1112–1119, Jul. 2013.

[23] Y. Chen, J. Huang, C. Lin, and J. Hu, "A partial selection methodology for efficient QoS-aware service composition," *IEEE Trans. Serv. Comput.*, vol. 8, no. 3, pp. 384–397, May 2015.

[24] X. Jin, S. Chun, J. Jung, and K. H. Lee, "IoT service selection based on physical service model and absolute dominance relationship," in *Proc. IEEE 7th Int. Conf. Service-Oriented Comput. Appl.*, Nov. 2014, pp. 65–72.

[25] K. Dar, A. Taherkordi, H. Baraki, F. Eliassen, and K. Geihs, "A resource oriented integration architecture for the Internet of Things: A business process perspective," *Pervasive Mobile Comput.*, vol. 20, pp. 145–159, Jul. 2015.

[26] A. Kouicem, A. Chibani, A. Tari, Y. Amirat, and Z. Tari, "Dynamic services selection approach for the composition of complex services in the Web of objects," in *Proc. IEEE World Forum Internet of Things (WF-IoT)*, Mar. 2014, pp. 298–303.

[27] A. Yachir, Y. Amirat, A. Chibani, and N. Badache, "Event-aware framework for dynamic services discovery and selection in the context of ambient intelligence and Internet of Things," *IEEE Trans. Autom. Sci. Eng.*, vol. 13, no. 1, pp. 85–102, Jan. 2016.

[28] G.-H. Tzeng and J.-J. Huang, *Multiple Attribute Decision Making: Methods and Applications*. Boca Raton, FL, USA: CRC Press, 2011.

[29] D. Duque, L. Lozano, and A. L. Medaglia, "An exact method for the biobjective shortest path problem for large-scale road networks," *Eur. J. Oper. Res.*, vol. 242, no. 3, pp. 788–797, 2015.

OSAMA ALSARYRAH received the M.S. degree in computer science from the University of Jordan, Jordan, in 2007.He is currently pursuing the Ph.D. degree with the Department of Computer Science and Engineering, Yuan Ze University, Taiwan. His research interests include future communication systems, Internet of Things and smart applications, Web services, sensor networks, and distributed computing.

IBRAHIM MASHAL received the M.S. degree in computer science from the University of Jordan, Jordan, and the Ph.D. degree in computer science, specializing in Internet of Things, from Yuan Ze University, Taiwan, in 2016. He is currently an Assistant Professor with the Computer Science Department, Aqaba University of Technology. His research interests include future communication systems, Internet of Things and smart applications, wireless mobile networks, and sensor networks.

TEIN-YAW CHUNG (M'87) received the M.S. and Ph.D. degrees in electrical and computer engineering from North Carolina State University, Raleigh, NC, USA, in 1986 and 1990, respectively. From 1990 to 1992, he was with the Network Service Division, IBM, RTP, NC, USA. Since 1992, he has been with Yuan Ze University, Taoyuan, Taiwan, where he is currently a Full Professor. His current research interests include overlay network, multimedia communication, and mobile networking.