Abstract—Systems built for future Internet of Things (IoT) may have a large number of intelligent objects with sensing, actuating, and computing capabilities. In a specific context, a smart application deployed in a system needs to meet the QoS requirement in order to provide a good experience for users. We are building the WuKong middleware which adopts the service oriented paradigm to help application developers define the logical functionalities using Flow Based Programs (FBP). WuKong then maps an FBP onto physical smart devices and actuators, and deploys the codes for executions. In this paper, we study the QoS oriented mapping algorithm for applications with a set of QoS definitions. After developers specify how each attribute contributes to the overall QoS, WuKong will find the best mapping solution according to the user QoS specification. We formalize the problem as a maximum weighted bipartite problem, and present the algorithm based on the Integer Linear Programming. We study the algorithm performance by simulation.

Keywords—Internet of Things, Middleware, Service-oriented computing, Service Selection, Service Composition, Matchmaking;

I. INTRODUCTION

Internet of Things (IoT) envisions a future that a large number of real-world objects will be integrated through Internet. Due to the advances in sensor technology and the decreased sensor production cost, the growth of sensor deployments has increased over past five years. It was estimated that the number of sensors and actuators will grow to millions and even billions. As a result, interoperability, scalability and flexibility challenges must be addressed carefully [1]. Most smart objects with sensing and actuating capabilities are developed on different platforms and are connected to different networks. Wide heterogeneity of smart objects increases the difficulty of interaction between IoT systems. Application developers need to re-engineer and redeploy their applications, which is considered laborious and time-consuming task, whenever there is a change in environment.

To isolate the platform independent logic application functionality with the physical devices that provide sensing and actuating features, the WuKong project [2], [3] supports a flow-based programming (FBP) model, shown on the right side of Fig. 1, to define the data and control flow among virtual sensing devices. Given an FBP, users can specify policies to impact the system deployment decisions. The WuKong middleware will then select the physical devices to provide specific functionalities under constraints from policies.

Our previous study [4] presents a flexible mapping algorithm that achieves a low overall system energy consumption and long system life-time under location and energy policies on FBP components and devices respectively. However, the model only considers energy consumption as the optimization goal. In real world, the mapping decision for an application should also achieve other runtime QoS objectives such as accuracy, response time, etc.

In Fig. 1, we show an example IoT system deployed in an indoor environment on the left and an FBP application on the right. The FBP specifies that a PIR component will trigger the AC control and turn on the light automatically. Depending on the user’s needs, different sensors may be used to execute the FBP. Since there are two physical PIR sensors capable of providing similar functionality, the system may select the one by the front door rather than the one in the kitchen if a faster response time is desired for QoS. On the other hand, if these two PIR sensors are heterogeneous, the one in kitchen has a high quality and the one by the front door is not working well, the FBP should use the one in the kitchen if accuracy is more important for QoS. Which device to use in a given context is a decision to be made by WuKong but under user-defined QoS preferences.

In this paper, we introduce a quality score system to hide the complexity and diversity of application QoS, and present a mapping methodology that takes multiple QoS attributes into consideration and finds the optimal deployment decision with the highest overall application level quality score. We model the problem as a maximum weighted bipartite problem, and solve it using Integer Linear Programming (ILP).

The rest of the paper is structured as follows. We review the literature on IoT middleware solutions and related work on service selection and composition in Section II. In Section III, we give the background on our WuKong project. The problem description for our research is given in
Section IV. Section V presents the proposed solution using the Maximum Weighted Bipartite Matching problem model. Performance study using the proposed algorithm is reported in Section VI. Finally we present a conclusion and prospects for future research in Section VII.

II. RELATED WORK

Since the vision of the explosive growth IoT industry, many companies design their own hardware, RF communication protocol, and smart things competing for the market. It brings the problem of interoperability between devices and also the programmability of connected things. Many Internet of Things projects present middleware solutions to improve the programmability of IoT systems. As a toolkit for IoT, middleware can help hide the complexity and heterogeneity of the underlying hardware and network platforms, ease the management of system resources, and increase the predictability of application execution [5], [6]–[8]. MagnetOS and LooCI [9] are two of these middleware designed for wireless sensor network. By adopting philosophy in IoT, WuKong applications are viewed as a virtual flow between smart objects [2], [3]. In WuKong, IoT programmers can interact with IoT systems by translating their requirements into high-level flow of information. Therefore, sensors and actuators can communicate with each other even if they are deployed in different platforms and different networks.

In service computing community, automatic QoS aware service composition [10] and service selection [11] are two active research topics. In the research of service selection, the functionality of distributed system is described by a business process which is composed by a set of abstract services in a directed acyclic graph. The problem is to find a best combination of real services to achieve overall QoS optimization by finding proper services instances from a large set of service candidates with different QoS. The automatic service composition problem is even harder for composition algorithm need to compose a flow of service to meet the both functionality requirement and QoS constraints. In this work, the QoS service is well defined, but the characteristic of every heterogeneous physical sensors is different. Our work define a extendable scoring system to convert them into QoS performance values in different application context, environment and user experience.

In the research of sensor network, lots of researchers devise MAC protocols and routing protocols to improve the QoS of homogeneous sensor network in the terms of energy consumption, reliability and real-time data transmission. Since IoT system is designed to improve quality of people’s life, thus mostly in-in-door environment. In such environment, the QoS of a system is no longer determined by network itself, but also impacted by the heterogeneous feature of things and also interoperation between things. Our research mainly take these into attributes consideration, and aim to improve the QoS of intelligent applications itself. All these work inspire us in the aspect of finding the optimal deployment solution for a FBP according to the QoS requirement of an intelligent application in the highly evolvable and reconfigurable WuKong middleware.

III. BACKGROUND ON WUKONG

A WuKong IoT system consists of a mixture of distributed sensing devices and gateway devices. Each embedded sensing device is capable of running multiple tasks such as sensing, computing, or actuating at the same time, depending on what sensors or actuators are installed. The gateway devices connect and support the communication of neighboring devices as well as remote connections via RF and wired communication. WuKong has a Master management node, which plays as a policymaker, to coordinate and manage all underlying sensor node devices. WuKong Master can run on some gateway devices and configure all sensing and gateway devices, making them cooperate with each other to run an IoT application.
In WuKong, the flow based programming model is used for composing IoT applications. In Fig. 2, users interact with FBP editor tool, and utilize a set of virtual service components, which is called WuClass in WuKong, to design an application by organizing the flow of information between them. The mapping compiler in WuKong accepts and translates the high-level language application into a number of machine-dependent codes, and then deploys them onto corresponding devices. Therefore, the node devices can cooperate with each other like the way defined in the FBP editor as shown in Fig. 1.

A. IoT services

Smart things with sensing, actuating and computing capabilities can be abstracted as IoT service components, and further utilized for designing IoT applications. In WuKong, we use WuClasses to model IoT services. In the WuKong design environment (including WuKong Master, FBP editor, node devices), we uses a universal WuClass library for references in order to model an IoT service. Each WuClass has a definition in the library XML file, which makes the WuClass manageable for WuKong master. The definition of a WuClass includes,

- The set of functional properties as exposed resources for the system.
- The set of non-functional properties that specify the characteristics of the WuClass. These properties may be updated by WuKong Master during run-time monitoring.
- The implementation of update() function used to specify WuClass’ behavior.

In addition, the definition also include WuClass hierarchy so as to model the relationship between WuClasses. For example, outdoor temperature sensor service and indoor temperature sensor service can both be classified as temperature sensor service. The functional properties are the resources that a WuClass provides for system, reading value is an example functional properties for temperature sensor service. Two functional properties can be connected if the type of the properties are compatible. Non-functional properties can describe the characteristics of a WuClass service. In IoT system, non-functional properties on each service are needed to be updated as the real-time condition or environment changes for more accurate service specification.

After defining WuClasses, WuObject is the instance to perform functionality of the WuClass. Many WuObjects, or services, are distributed in an IoT system as the left side of Fig. 1 shows. Each node device in WuKong system can have multiple WuObjects running on. When there is any change on the properties in a WuObject, WuKong software triggers the update() function of the WuObject and updates other functional properties. For example, when there is change for the input value of threshold component, the update function of threshold component will be triggered and new output value is stored after the computation. Then, the software on each node device will propagate functional property values between WuObjects if the property has been changed.

B. IoT applications

Most IoT applications are event-driven by nature; thus we can use the flow of information between virtual service components, which is called service graph, to describe the logics in an application. A service graph is defined as a network of virtual service components, each of which belongs to a service class. The link between service components stands for the flow of information. In WuKong, the FBP model implements the concept of service graph using WuClass component blocks as Fig. 2 shows.

A WuKong user can define the initial functional property value for each WuClass, such as activation threshold value and refresh rate. After a FBP is deployed, WuKong is designed to monitor and identify what property values are best for system performance and application preference and then automatically refine the settings and update functional properties.

A WuKong user can also specify the user preference for each WuClass as Fig. 2 shows. For each component, the sliders in the property editor can be adjusted and used to specify how each non-function property contributes to the overall quality of service performance. For example, Property “energy efficiency” is weighted heavier than “precision”, when the developer prefers an energy-efficient service even when the precision is not good enough.

IV. Problem Description

WuKong is responsible for deploying all WuClasses in an FBP to their corresponding WuObjects, each of which is hosted by a physical device in IoT systems. Among all
correct mappings, how to select a good one is the problem we are interested in. In Fig. 1, the application’s best mapping is to map the PIR WuClass to the PIR sensor near the door instead of the one in the kitchen.

### A. IoT Service Matchmaking

Each IoT application implemented in FBP can be defined by a directed acyclic graph (DAG) \( F = (C, L) \) with user preference \( P \) where,

- \( C \) is the set of WuClasses.
- \( L \) is the set of link that stands for the flow of information. WuClasses can be connected via the functional properties on each WuClass.
- \( P \) is user preference set for WuClasses. For each WuClass \( C_i \), user can decide the values in weight set \( P(C_i) \). For each component \( C_i \), we let \( \sum_k P_k(C_i) = 1 \).

Each element \( P_k(C_i) \) represents the proportion of emphasis that a user puts for a specific QoS attribute.

An IoT systems \( M \) consists of a set of available services \( S \), and a set of property set \( Q \) to describe the characteristics of each service. The services are instances of virtual service components, which we called WuObject in WuKong. Through automatic service discovery and capability identification, WuKong can discover all WuObjects which are provided by some physical devices in \( M \). Because of different real-time and hardware conditions for each WuObject, a vector \( Q(S_j) \) is used to describe it. The value of each element in \( Q(S_j) \) represents a specific QoS attribute and is normalized in the range from 0 to 1 in our problem.

In WuKong, a WuObject \( S_j \) can be matched to a WuClass \( C_i \) if \( S_j \) is capable of providing the needed functionalities for \( C_i \). The class hierarchy defined in WuKong Master helps determine if a WuClass can be mapped to a WuObject. For example, a WuClass "Temperature" can be mapped not only to "Temperature" WuObject, but also to "Wall Temperature Sensor" or others. Based on the hierarchical definition of WuClasses, WuClass \( C_i \) and WuObject \( S_j \) can be matched if and only if they are related in the WuClass hierarchy definition.

We define the matching score \( y(C_i, S_j) \) between \( C_i \) and \( S_j \) as Eq. 1 with class hierarchical relation.

\[
y(C_i, S_j) = \begin{cases} 
    1 - d(C_i, S_j), & \text{if } C_i \text{ is same class with } S_j \\
    \alpha(1 - d(C_i, S_j)), & \text{if } C_i \text{ is super class of } S_j \\
    \beta(1 - d(C_i, S_j)), & \text{if } C_i \text{ is sub class of } S_j \\
    0, & \text{otherwise}
\end{cases}
\]  

(1)

In this study, we set \( \alpha = 0.6 \) and \( \beta = 0.4 \) but can adjust it some other settings. We evaluate the QoS performance of \( S_j \) when mapping it with service component \( C_i \) under user preference \( P(C_i) \) using the weighted distance function defined in Eq. 2. Each element in user preference of Wu-Class \( C_i \), \( P_k(C_i) \) represents how a QoS parameter \( Q_k(S_j) \) contributes to the overall QoS performance \( d(C_i, S_j) \).

\[
d(C_i, S_j) = \sqrt{\sum_k P_k(C_i)(1 - Q_k(S_j))^2},
\]  

(2)

### B. Problem Definition

Given an IoT application implemented in FBP \( F = (C, L) \) with \( P \) and an IoT system \( M \) consists of \( S \) and \( Q \), we want to find the best user-defined, QoS-aware matchmaking to deploy the FBP. We define the deployment matrix \( x \) to specify the relationship between WuClasses \( C \) and WuObjects \( S \), where variable \( x_{ij} = 1 \) denotes that \( C_i \) has been matched with \( S_j \), \( x_{ij} = 0 \) to denote \( C_i \) has not being matched to \( S_j \). Therefore, the total matching score for each service component \( C_i \) can be computed by:

\[
\text{Score}(C_i) = \sum_j x_{ij} \times y(C_i, S_j)
\]  

(3)

The objective function can be defined as Eq. 4. We want to find the matching matrix \( x \) so that the total matching score over FBP \( F \) is the maximum.

\[
\arg\max_x \sum_i \text{Score}(C_i)
\]  

(4)

### V. MATCHMAKING ALGORITHM

This section presents the user-defined QoS-aware matchmaking scheme. We show how we model the problem to be a maximum weighted bipartite matching problem. We then present the integer linear programming (ILP) formulation for the problem.

#### A. Maximum Weighted Bipartite Matching Problem

The Maximum Weighted Bipartite Matching problem is defined as follows. Given an edge-weighted bipartite graph \( G = (V, E) \) with partition \( A, B \) and a weight function \( W : E \rightarrow R \), we want to find a matching of the maximum weight where the weight of matching \( M \) is given by \( \sum_{e \in M} W(e) \). The matching \( M \) in \( G \) is a set of pairwise non-adjacent edges, which means no two edges share a common vertex.

We can model the service matchmaking problem as a maximum weighted bipartite matching problem, which is also known as the assignment problem. We consider an edge-weighted bipartite graph \( G(V, E) \) and a weighting function \( W \), with partition \( A, B \) as Fig. 3 shows, where

- Partition \( A \) is the set of WuClass in FBP \( F \).
- Partition \( B \) is the set of WuObject in IoT system \( M \).
- Edge set \( E \) is the pairs between partitions \( A \) and \( B \).
- Weighting function \( W \) is defined as the weighted distance calculated by Eq. 2.

Given FBP \( F = (C, L) \) with user preference \( P \) and IoT system \( M \) which includes \( S \) and \( Q \), we first construct
the corresponding edge-weighted bipartite graph \( G(V, E) \) and \( W \). We will then use well-known algorithms to find the maximum weighted bipartite matching on graph \( G \). Algorithm 1 shows how we construct the graph \( G \), which is a complete edge-weighted bipartite graph. We assign the matching score \( y(C_i, S_j) \) to be edge’s weight even when the matching score is zero.

**Algorithm 1 Edge-Weighted Bipartite Graph Generation**

**Input:** FBP \( F = (C, L, P) \) and IoT system \( M = (S, Q) \)

**Output:** An edge-weighted bipartite graph \( G = (V, E) \) and weighting function \( W \)

1. **for all** service component \( C_i \) in \( C \) do
   2. create new vertex \( v \) and add to partition \( A \) and \( V \)
   3. **for all** service \( S_j \) in \( S \) do
      4. create new vertex \( v' \) and add to partition \( B \) and \( V \)
      5. add edge \( e = (v, v') \) to \( E \), and assign the weight \( W(e) = y(C_i, S_j) \)
   6. end for
7. end for

**B. Problem Analysis**

Finding the best QoS-aware service matching matrix \( x \) between FBP \( F \) and IoT system \( M \) is equivalent to finding the maximum weighted bipartite matching \( M \) on the edge-weighted bipartite graph \( G \). In this paper, we address only one-to-one matchings between partitions \( A \) and \( B \). However, many-to-one and one-to-many mapping may also be possible in service selections, but will be left as our future work.

Every WuClasses has its best-fit WuObject by considering user preferences. To compare all possible combinations to find the best solution, we have to compute \( m^n \) combinations, where \( n \) is the number of WuClass and \( m \) is the number of WuObject. Therefore, we may use some heuristic methods to efficiently find service matchmaking. If we traverse all the WuClasses in partition \( A \) and choose the most heavy edge for its matching, we can find a solution in \( O(n \times m) \) time complexity. However, this may not produce an optimal solution since in our problem setting, multiple WuClasses can have the same WuObject to best match. Therefore, we are interested in solving the service matching problem by using the bipartite matching algorithm to find the best solution efficiently since the number of available WuObject can be huge in IoT systems.

**C. ILP Formulation**

After the edge-weighted bipartite graph \( G = (V, E) \) and weighting function \( W \) are generated, we apply integer linear programming (ILP) to formulate the maximum weighted bipartite matching. Given bipartite graph \( G \), we want to find the matching matrix \( x \) where,

\[
\text{max } \sum_{i,j} x_{ij} \times W(A_i, B_j) \quad (5)
\]

subject to:

\[
\sum_{i} x_{ij} = 1, \forall 1 \leq i \leq |A| \quad (6)
\]

\[
\sum_{j} x_{ij} \leq 1, \forall 1 \leq j \leq |B| \quad (7)
\]

Eq. 6 ensures that each WuClass can find its matching WuObject in \( S \) since a matchmaking is not successful unless all service components on a FBP can be mapped to a service on a device. Eq. 7 ensures that each WuObject can only be matched by one WuClass in an IoT system since we already assume that services and WuObjects are not sharable in our problem setting.

**VI. Performance Study**

In this section, we show experiment results to evaluate the proposed ILP solution for service matchmaking. We compare ILP with the baseline method, which adopts a greedy matching algorithm to find the best matching for each component. We show how we set up the simulation testbed, the consideration for determining system parameters and performance merit, and the simulation result.

For an IoT system with a set of available services, we generate simulated systems for our testbeds as follows. For the WuClass hierarchy, we randomly generate a tree-like structure to model the relationship between WuClasses. Then, we randomly generate WuObjects of different class. For each WuObject \( S_i \), we generate a QoS attribute set \( Q(S_i) \) with 10 QoS parameters for describing the service, where every element in the set has a value between 0 and 1. We then use the JGraphT library to generate a random flow based program as target IoT applications, including 1000...
instances of random topology. For each WuClass component $C_i$ on FBP, we randomly generate a weight set $P(C_i)$ with the same size as the QoS attribute set, 10, as user preferences. After the weight set is generated for each FBP component, we normalize each element in the weight set so that the total sum of all elements in the weight set will be 1.

In this paper, the data for 100 WuClasses in FBP and a system with 1000 WuObjects is reported. We compare the performance with ILP and Greedy Matching, and we evaluate the performance by the total matching score defined in Eq. 4.

As Fig. 4 shows, after 1000 runs, we find that ILP can find a better QoS matching for the application than the greedy matching algorithm. However, ILP takes more time to compute the result, which may not be scalable for the setting of large IoT systems. We are currently investigating heuristic algorithms for the matching problem so that the system is more scalable. On the other hand, for most smart home environment, the ILP solution may be acceptable.

VII. CONCLUSION AND FUTURE WORK

As the number of IoT devices grows, it becomes challenging to deploy applications through the network. To make programming in IoT simple is important since we should engage everyone with the IoT network in the future. Automatically finding a good mapping between an abstract application and IoT systems is important for intelligently helping users deploy their applications. By using WuKong, we raise the level of programming abstraction so that users only need to focus on their logical design of an application. We formulate the service matchmaking problem as a maximum weighted bipartite matching problem, and compare the performance between greedy matching and ILP solution. We find that ILP solution is optimal but consumes more time and may not be scalable in large-scale IoT systems. Improving the greedy service matchmaking is a future direction of this research.

ACKNOWLEDGMENT

This work was also supported by National Science Council, National Taiwan University and Intel Corporation under Grants NSC102-2911-I-002-001 and NTU103R7501.

REFERENCES