View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by Elsevier - Publisher Connector

HOSTED BY
ELSEVIER
Available online at www.sciencedirect.com
ScienceDirect
journal homepage: www.elsevier.com/locate/jides

Co-locating services in IoT systems to minimize the communication energy cost

Zhenqiu Huang[a,*], *Kwei-Jay Lin*[a], *Shih-Yuan Yu*[b], *Jane Yung-jen Hsu*[b]

[a] *Department of Electrical Engineering and Computer Science, University of California, Irvine, United State*
[b] *Department of Computer Science and Information Engineering, National Taiwan University, Taiwan*

ARTICLE INFO

Article history:
Received 23 October 2014
Received in revised form
19 January 2015
Accepted 15 February 2015
Published online 7 March 2015

Keywords:
Service mapping
Energy optimization
Sensor selection

ABSTRACT

Ubiquitous sensing and actuating devices are now everywhere in our living environment as part of the global cyber–physical ecosystem. Sensing and actuating capabilities can be modeled as services to compose intelligent Internet of Things (IoT) applications. An issue for perpetually running and managing these IoT devices is the energy cost. One energy saving strategy is to co-locate several services on one device in order to reduce the computing and communication energy. In this paper, we propose a service merging strategy for mapping and co-locating multiple services on devices. In a multi-hop network, the service co-location problem is formulated as a quadratic programming problem. We show a reduction method that reduces it to the integer programming problem. In a single hop network, the service co-location problem can be modeled as the Maximum Weighted Independent Set (MWIS) problem. We show the algorithm to transform a service flow to a co-location graph, then use known heuristic algorithms to find the maximum independent set which is the basis for making service co-location decisions. The performance of different co-location algorithms are evaluated by simulation in this paper.

© 2015 Qassim University. Production and Hosting by Elsevier B.V.
This is an open access article under the CC BY-NC-ND license
(http://creativecommons.org/licenses/by-nc-nd/4.0/).

1. Introduction

The vision of *Internet of Things* (IoT) has recently brought many new and game-changing products to the market. Sensors and communication capabilities have been added into many traditional devices, controllers, and infrastructures so that systems can make informed and smart decisions. Market research has predicted that by 2020, more than 50 billion smart devices will be deployed and connected to Internet, serving people more timely and properly. New applications have been developed using various IoT platforms, for sensing and collecting information to identify our needs, then composing and deploying smart services to make our lives simple and safe.

The WuKong project [1,2] is building the support for self-configuring and self-evolving physical and digital ecosystems. The WuKong middleware is designed to automatically discover physically connected IoT devices and use them to compose high level IoT applications. Flow-based programs (FBP) (Fig. 1) are defined to specify *abstract* sensors to be

Peer review under responsibility of Qassim University.
* *Corresponding author*.
E-mail address: zhenqiuh@uci.edu (Z. Huang).

http://dx.doi.org/10.1016/j.jides.2015.02.005
2352-6645/© 2015 Qassim University. Production and Hosting by Elsevier B.V. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

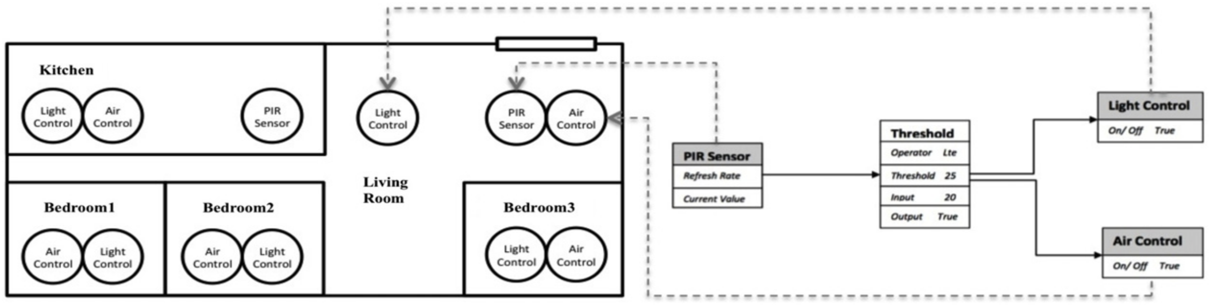


Fig. 1 – Flow based program for a smart home.

utilized and the workflow among them. WuKong then uses a polling protocol to discover and identify physical devices, and finds the optimal mapping from the FBP definition to services on devices for execution. As shown in Fig. 1, an FBP can be deployed in the living room for automatically controlling light and room temperature according to the needs of the people there. To adapt to specific user needs under various context, WuKong accepts policies specified by users for making personalized mapping decisions. It also tries to continuously improve the system performance by monitoring executions and making reconfigurations. In this way, WuKong provides the support for intelligent IoT ecosystems with the goals of agile deployment and optimized performance.

One issue for perpetually running IoT services on distributedly located devices is the energy cost. Running 50 billion devices and communicating among them will use a lot of energy. Researchers have proposed various device sleep scheduling algorithms [3] to keep some devices power off or running at a low-power mode. Another approach is to reduce network communication traffic to conserve energy. In this research, we investigate how to minimize the communication among devices. We use a service mapping scheme [4] to co-locate as many FBP services on the same device as possible to reduce distributed communication, in order to minimize the total energy cost for an application.

In our previous study [4], we use a simple greedy algorithm that co-locates two neighboring services with the largest communication cost first. In this paper, we present a comprehensive study on the service co-location problem for both multi-hop and single-hop networks. For devices in a multi-hop network, the service co-location problem is formulated as quadratic programming problem. We show a reduction method that reduces it to an integer programming problem. For single-hop networks, we present methods to find better solutions by transforming the service co-location problem to the Maximum Weighted Independent Set (MWIS) problem [5], which is a well-known data clustering problem. Using the MWIS model, we can find solutions that reduce about 10% communication energy from our previous solution in [4]. This paper is an extension of [6] by including theoretical complexity analysis as well as the study on multi-hop networks.

Our contribution in this paper includes:

1. We define the general service co-location problem on multi-hop networks. We model it as a quadratic programming problem, and show how to reduce it to integer programming.
2. We also model the single-hop co-location problem as an optimization problem, and show its NP-completeness.
3. To reduce the computation complexity, we transform the optimization problem to the MWIS problem by defining the co-location graph, and design the transformation algorithm to construct the co-location graph from an FBP.
4. We use efficient algorithms to find service co-locations from co-location graphs, and then select the devices to efficiently run the new co-located services.
5. We show simulation results to compare the performance and effectiveness of different heuristic algorithms.

The rest of this paper is organized as follows. Section 2 presents related work on service selection in service computing and energy saving in sensor networks. Section 3 introduces the concept of flow based program, the system architecture of WuKong systems, and the communication energy model. Section 4 investigates the co-location problem on multi-hop networks. Section 5 proves the NP-completeness of the optimization problem in single hop network. Section 6 shows how we transform a general flow based program to its corresponding co-location graph. We also present a greedy selection framework, which can be used to adopt various greedy algorithms for the MWIS problem, to select the best co-location decision to save energy for an FBP. A simulation study on the performance using different algorithms for single-hop networks is shown in Section 7. The paper is concluded in Section 8.

2. Related work

In service-oriented computing (SOA) research, QoS aware service composition [7,8] and service selection [9] are two important topics. However, most SOA research concentrates on performance and QoS issues, rather than energy cost. Moreover, the communication overhead usually is not considered. Our research adopts the composition ideas from SOA to build IoT applications, but also takes energy consumption into consideration since IoT systems must be energy efficient in order to run constantly.

In wireless sensor network research, energy efficiency has been well studied. Earlier projects have focused on minimizing energy consumption on individual sensor nodes, whereas more recent studies have suggested that the energy efficiency for the whole system is actually more important to extend network lifetime [10]. A common technique [11–13] to achieve energy efficiency is to put as many sensors in the sleep mode as possible, and keep only enough sensors in the active mode for sensing, communicating and processing. Wang et al. [3] propose a cross-layer sleep scheduling design in a service-oriented WSN while meeting the system requirement on the number of active service nodes for each service at any time interval. Another approach to prolong the network lifetime is energy consumption balancing. [14] studies the uneven energy depletion phenomenon in sink-based wireless sensor networks. [15] considers an energy efficient layout with a good coverage by using a multi-objective particle swarm optimization algorithm. [16,17] propose two node deployment schemes, namely, distance-based and density-based, to balance each sensor node's energy consumption and to prolong network lifetime. In [18], we show how to use quadratic programming model to balance the energy usage. Nevertheless, since the quality of sensor data is an important factor on how intelligent a system can be, we also use the QoS oriented mapping [19].

To build energy efficient IoT systems, our earlier work [4] has proposed a simple greedy algorithm that iteratively co-locates, if possible, two connected components with the largest communication cost on the same device to reduce energy consumption. In this work, we model the problem as the Maximum Weighted Independent Set (MWIS) problem [20,5]. MWIS has been used to solve many large data clustering problems. The Maximum Independent Set (MIS) problem, which is a special case of MWIS in that the weight of each vertex is 1, has been studied in [20] using the GMAX and GMIN greedy algorithms. Extending the result, [5] proposes several greedy algorithms, including GWMAX, GWMIN, and GWMIN2, for MWIS. We use these algorithms to solve the service co-location problem in this paper.

3. System model

3.1. IoT service and application

An IoT application is defined by a network of service *components*, each of which belongs to a service class, called *WuClass* in the WuKong paradigm. Similar to the class definition in object-oriented programming, a WuClass defines the abstraction of sensing and actuating functionalities. For execution, a *WuObject* residing on a device can be used to provide the capability. WuKong supports the flow based structure so that application developers only need to design the flow structure between virtual service components. Each FBP is defined by a directed acyclic graph (DAG) $G(C, L)$ where C is a set of components C_i and L is the set of links $L_{ij} = (C_i, C_j)$ between components C_i and C_j.

Each component has a set of properties defined, such as sensor reading and sensor refresh rate. An FBP user can define the initial property value for each component. After an FBP is deployed, WuKong allows system developers to plug in progression modules in order to learn what property values work best for the application and to automatically refine the component settings. In this paper, however, we consider only how to deploy these components initially and leave the self-evolving reconfiguration study to future work.

3.2. Devices and system

In our study, IoT systems are modeled as distributed systems consisting of a set of sensing, actuating and computing devices deployed in different locations of a target environment, connected by wireless or fixed line communication networks. An IoT system M has a set of physical devices D. On each device D_k, there may be several services available for sensing, actuating, and/or computing. *WuObjects* S_{ik} is an instance of WuClass C_i and is hosted on device D_k. Device D_k may host multiple sensing or computing WuObjects to execute the components in an FBP. For example, in Fig. 2, S_{11} and S_{21} on device D_1 can be used for C_1 and C_2 respectively.

The WuKong middleware is responsible for mapping FBP components to different devices capable of providing corresponding services. Fig. 2 shows an FBP defined with 7 components (C_i) in a system of 5 devices (D_k). In the FBP, the weight on each link shows the units of energy communication cost. The data volume of each communication link could be inferred from WuClass definition and refined by real-time monitoring and prediction. Later in Section 5, we will define the overall energy cost of an FBP. Sometimes, due to the environmental and context changes, the data volume on each link may fluctuate during FBP executions. In this paper, however, we assume the data size on each FBP link is fixed.

3.3. Communication energy

In [21], energy costs for transmitting and receiving t bits of data over a communication distance of d meters have been formulated as follows:

$$E_T(t, d) = E_{elec} \times t + \epsilon_{amp} \times t \times d^2 \quad (1)$$

$$E_R(t, d) = E_{elec} \times t \quad (2)$$

where radio electronics parameter E_{elec} is about 50 nJ/bit and transmit amplifier parameter ϵ_{amp} is about 10 pJ/bit/m^2.

For each FBP link between service components (C_i, C_j), if it is mapped to the communication between devices (D_n, D_m), the energy consumption $E(t_{ij}, d_{nm})$ of the link will be decided by the transmission energy $E_T(t_{ij}, d_{nm})$ on D_n, the receiving energy $E_R(t_{ij}, d_{nm})$ on D_m, and, for multi-hop routings, the total communication energy used by all devices on the route from D_n to D_m. But if we use a device that can host both end components of a link, then the energy cost U_{ij} becomes zero. This is the motivation for our energy sentient co-location strategy.

4. General sensor selection problem in multi-hop network

In large IoT application scenarios like smart factory or smart building, an FBP will be mapped to devices whose communication to other devices may go through multiple hops. In this section, we present the energy model for multi-hop networks. We show how to formulate the problem as a quadratic programming problem, and how to solve it by integer programming.

4.1. Energy model

In this paper, we consider the multi-hop network with a static routing table, in that each device always routes messages to

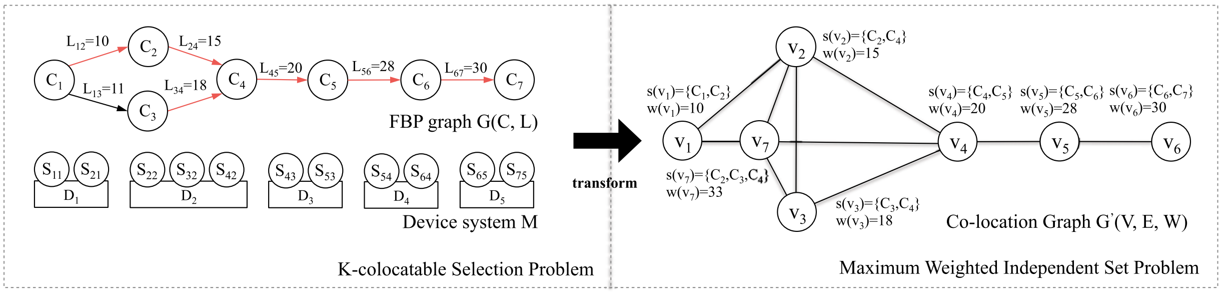


Fig. 2 – Co-location mapping example.

the same destination using the same path. Usually, devices use a static path A_{nm} between any two devices D_n and D_m by the shortest path between them.

Based on the energy formula Eqs. (1) and (2), we define the cost of transmitting t bits from D_n to D_m along the path A_{nm} with H hops as follows:

$$E_T(t, D_n, D_m) = H \times E_{elec} \times t + \epsilon_{amp} \times \sum_h d_h^2 \tag{3}$$

$$E_R(t, D_n, D_m) = H \times E_{elec} \times t. \tag{4}$$

In Eq. (3), d_h is the distance between two devices in the hth hop along the path A_{nm} from D_n to D_m. Since sparse and diverse network density is a common attribute of multi-hop networks, we take distances into consideration in Eq. (3).

4.2. *Quadratic programming formulation*

Given an FBP and an IoT system, we denote the data volume of an FBP link L_{ij} to be t_{ij} bits, the routing path between two devices D_n and D_m to be A_{nm} in the static routing table. During the mapping stage, the FBP link L_{ij} is mapped to a device pair (D_n, D_m), which are the start and end devices of the physical routing path A_{nm}. We define $H(L_{ij})$ to be the set of paths between all devices D_n and D_m where D_n can host C_i and D_m can host C_j. Let $x_{ik} = 1$ denote C_i has selected to use service S_{ik} on device D_k. Then, we can formulate the energy cost of $\mu(L_{ij})$ of link L_{ij} as:

$$\mu(L_{ij}) = \mu_T(L_{ij}) + \mu_R(L_{ij}) \tag{5}$$

$$\mu_T(L_{ij}) = \sum_{A_{nm} \in H(L_{ij})} x_{in} * x_{jm} * E_T(t_{ij}, D_n, D_m) \tag{6}$$

$$\mu_R(L_{ij}) = \sum_{A_{nm} \in H(L_{ij})} x_{in} * x_{jm} * E_R(t_{ij}, D_n, D_m) \tag{7}$$

i.e. the energy cost of L_{ij} is the cost to transmit and receive t_{ij} bits between devices D_n and D_m.

The energy cost $\mu(C_i)$ of a component C_i can be defined by:

$$\mu(C_i) = \sum_p \mu_T(L_{ip}) + \sum_q \mu_R(L_{qi}) \tag{8}$$

i.e. the energy cost of C_i is the sum of transmitting energy cost $\mu_T(L_{ip})$ of all its remote out-links L_{ip} in the FBP and receiving energy cost $\mu_R(L_{qi})$ of all its remote in-links L_{qi} in the FBP. The total energy consumption on device D_k can be defined by:

$$\mu(D_k) = \sum_i x_{ik} * \mu(C_i) \tag{9}$$

i.e. the summation parameter i represents the ith WuClasses on the device D_k.

The optimization objective function is to minimize the overall energy consumption among all nodes:

$$\min \left(\sum_k \mu(D_k) \right) \tag{10}$$

subject to:

$$\sum_k x_{ik} = 1, \quad \forall 1 \le i \le N, 1 \le k \le M \tag{11}$$

where N is the number of components in an FBP, and M is the number of devices in an IoT system.

To show the problem is a quadratic programming problem, let us define the energy cost of component C_i on device D_k as $\mu^k(C_i) = x_{ik} * \mu(C_i)$. From Eq. (8), we have:

$$\mu^k(C_i) = x_{ik} * \left(\sum_p \mu_T(L_{ip}) + \sum_q \mu_R(L_{qi}) \right). \tag{12}$$

Let us first expand the transmission unit using Eq. (6):

$$\mu_T^k(C_i) = x_{ik} * \sum_p \sum_{A_{nm} \in H(L_{ip})} x_{in} x_{pm} E_T(t_{ip}, D_n, D_m). \tag{13}$$

Since the constraints in Eq. (11) ensure that two devices D_k and D_n cannot be selected for deploying a particular component C_i at the same time, we can see that:

$$x_{ik} * x_{in} = 0, \quad k \ne n. \tag{14}$$

Therefore, we can further simplify Eq. (13) as:

$$\mu_T^k(C_i) = \sum_p x_{ik}^2 \sum_{A_{km} \in H(L_{ip})} x_{pm} E_T(t_{ip}, D_k, D_m) \tag{15}$$

$$= \sum_p \sum_{A_{km} \in H(L_{ip})} x_{ik} x_{pm} E_T(t_{ip}, D_k, D_m). \tag{16}$$

In Eq. (15), since x_{ik} is a 0–1 integer variable, x_{ik} and x_{ik}^2 have the same value. Similarly, the receiving unit of Eq. (12) is:

$$\mu_R^k(C_i) = \sum_q \sum_{A_{nk} \in H(L_{qi})} x_{qn} x_{ik} E_R(t_{qi}, D_n, D_k). \tag{17}$$

Given the data volume of links and distances between devices are fixed in a problem instance, we can also calculate $E_T(t_{ip}, D_k, D_m)$ and $E_R(t_{qi}, D_n, D_k)$ as constants. In this way, we can define the energy consumption equation for each device. Thus, the final optimization problem is indeed a quadratic programming problem.

4.3. *Integer programming reduction*

Optimizing the quadratic programming problem is an NP-hard problem for which no polynomial algorithm is known. However, we can transform the problem to integer linear programming by rewriting the problem using new variable y_{ikpm} and constraints to take the value $x_{ik} * x_{pm}$ for every combination of x_{ik} and x_{pm}, and y_{qnik} for value $x_{qn} * x_{ik}$ also. The equivalence of two formulations has been proved in [22]. We thus obtain the following equivalent 0–1 linear programming definition:

$$\min\left(\sum_k \sum_i (\mu_T^k(C_i) + \mu_R^k(C_i))\right). \tag{18}$$

In Eq. (18), the transmission unit $\mu_T^k(C_i)$ and receiving unit $\mu_R^k(C_i)$ are thus transformed as follows:

$$\mu_T^k(C_i) = \sum_p \sum_{A_{km}\in H(L_{ip})} y_{ikpm} E_T(t_{ip}, D_k, D_m) \tag{19}$$

$$\mu_R^k(C_i) = \sum_q \sum_{A_{nk}\in H(L_{qi})} y_{qnik} E_R(t_{qi}, D_n, D_k). \tag{20}$$

In addition to the formal constraints defined, we need constraints for every y_{ikpm} and y_{qnik}:

$$y_{ikpm} \geq 0 \tag{21}$$

$$x_{ik} - y_{ikpm} \geq 0 \tag{22}$$

$$x_{pm} - y_{ikpm} \geq 0 \tag{23}$$

$$1 - x_{ik} - x_{pm} + y_{ikpm} \geq 0 \tag{24}$$

$$y_{qnik} \geq 0 \tag{25}$$

$$x_{qn} - y_{qnik} \geq 0 \tag{26}$$

$$x_{ik} - y_{qnik} \geq 0 \tag{27}$$

$$1 - x_{qn} - x_{ik} + y_{qnik} \geq 0. \tag{28}$$

These eight equations ensure the value of y_{ikpm} to be exactly the same as $x_{ik} * x_{pm}$, and value of y_{qnik} to be the same as $x_{qn} * x_{ik}$ in all cases. Therefore we can replace $x_{ik} * x_{pm}$ with y_{ikpm}, and replace $x_{qn} * x_{ik}$ with y_{qnik}. After we replace each $x_{ik} * x_{pm}$ and $x_{qn} * x_{ik}$ by its corresponding y_{ikpm} and y_{qnik}, the objective function is guaranteed to be optimal under the same setting of x_{ik} in the original objective function.

5. Communication minimization problem for single-hop network

In smaller scale IoT systems, devices are installed close to each other and communicate with each other in a single hop network. In this section, we study the energy parameters for such systems. We present the problem definition and the analysis on the computation complexity of the problem.

5.1. *Problem definition*

To study the problem complexity, we first formulate the mapping problem in a general problem $\mathbf{P_A}$. We then study a special class $\mathbf{P_K}$ of $\mathbf{P_A}$.

Given an FBP of n components to be deployed in a physical system of m sensing devices, the data communication of link L_{ij} between components C_i and C_j is known to be t_{ij} bits. The problem $\mathbf{P_A}$ is to find a mapping decision that maps each component in FBP to run on one device, while minimizing the total communication energy cost on these devices. In home environments, sensor devices may have a relatively uniform layout so that the distances between them are similar and do not make much difference on energy consumption. If so, we can simplify the second term of $E_T(t, d)$ in Eq. (1) to be independent of the distance between devices, and instead use a layout parameter, δ, i.e.

$$E_T(t, \delta) = E_{elec} \times t \times (1 + \delta). \tag{29}$$

With this approximated transmission energy model, we can define the transmission energy and receiving energy cost of link L_{ij} as below:

$$\mu_T(L_{ij}) = \sum_{A_{nm}\in H(L_{ij})} x_{in} * x_{jm} * E_T(t_{ij}, \delta) \tag{30}$$

$$\mu_R(L_{ij}) = \sum_{A_{nm}\in H(L_{ij})} x_{in} * x_{jm} * E_R(t_{ij}). \tag{31}$$

Then, we can define the energy cost $\mu(C_i)$ of a component C_i by:

$$\mu(C_i) = \sum_p \mu_T(L_{ip}) + \sum_q \mu_R(L_{qi}) \tag{32}$$

i.e. the energy cost of C_i is the sum of transmitting cost $\mu_T(L_{ip})$ of all its out-links L_{ip} in FBP and receiving cost $\mu_R(L_{qi})$ of all its in-links L_{qi} in FBP. Again we use variable $x_{ik} = 1$ to denote C_i has selected to run on device D_k. Then, we can find the energy consumption on a device as:

$$\mu(D_k) = \sum_i x_{ik} * \mu(C_i). \tag{33}$$

The objective function to minimize the total energy consumption on all devices is defined by:

$$\min\left(\sum_k \mu(D_k)\right) \tag{34}$$

subject to:

$$\sum_k x_{ik} = 1, \quad \forall 1 \leq i \leq N, 1 \leq k \leq M. \tag{35}$$

5.2. *Co-location graph*

We define problem $\mathbf{P_K}$ to be the K-sized co-location selection problem, if the total number of components that can be co-located on a device is no more than K in $\mathbf{P_A}$. The parameter K is determined by how many components exist on each device.

If we use the exhaustive search algorithm to solve the $\mathbf{P_K}$ problem, the time complexity is $O(mK * K^n)$, where n is the number of components in an FBP, m is the number of devices. However, some of them cannot be selected at the same time. For example, in Fig. 2, service component C_4 has 4 co-location options: $\{C_2, C_4\}$ on device D_2, $\{C_3, C_4\}$ on D_2, $\{C_4, C_5\}$ on device D_3, and $\{C_2, C_3, C_4\}$ together on D_2. These options are mutual exclusive since we can select only one device to deploy C_4.

We define a co-location graph as a vertex-weighted undirected graph $G(V, E, W)$, where V is a set of vertices that give all co-location options, E is a set of edges that represents the mutual exclusive relationship among co-location options, and W is a set of weighted labels that represent the gain when selecting a co-location option. Each vertex $v_i \in V$ represents a valid co-location option and contains a set of mergeable links. $s(v_i)$ is the set of WuClasses that may be co-located, and the weight $w(v_i)$ is the energy saving. The edge $e_{ij} \in G$ represents a conflict between v_i and v_j. For example, in Fig. 2, co-location option $v_4 = \{C_4, C_5\}$ and $v_5 = \{C_5, C_6\}$ are in conflict because they both have service C_5 which can reside on only one device, i.e. D_3 or D_4. The co-location node v_7 is for the option of placing C_2, C_3, C_4 together. In fact, a co-location graph may include vertices with many components co-located.

5.3. *Problem complexity*

We study the complexity of $\mathbf{P_K}$ by a reduction from the Maximum Weighted Independent Set (MWIS) problem to the co-location graph. MWIS is a well-studied graph problem [20,5]. Let $G = (V, E, W)$ be a vertex-weighted undirected graph without loops and multiple edges, where V is the set of vertices, E is the set of edges, and W is the vertex weighting function. For any nonempty set $S \subseteq V$, $W(S)$ is defined by $\sum_{u \in V} W(u)$. A subset $I \subseteq V$ is an *independent subset* of G if for any two vertices $u, v \in I$, $(u, v) \notin E$. An independent subset I of G is the *maximum* if there is no other independent subset I' of G such that $W(I) < W(I')$. MWIS is to find the independent subset from G that has the maximum total weight among all independent subsets.

If we find a MWIS solution for a co-location graph, the co-locations selected in those vertices will have no conflict with each other since they are not connected in the co-location graph. Moreover, the total weight is the maximum so that the energy saving is the largest.

Hastad [23] has shown that MWIS for a general graph is NP-hard in the strong sense. It is hard to approximate within $n^{1-\epsilon}$, for any $\epsilon > 0$. We now show the problem $\mathbf{P_K}$ is NP-hard even for $K = 2$ by reducing it from the MWIS problem.

Theorem 5.1. *Problem $\mathbf{P_K}$ is NP-hard in the strong sense for $K = 2$.*

Proof. Given an instance of MWIS, we assume each vertex v_i contains two numbers *p, q* that represent component C_p and C_q, and no vertices has the same two numbers. Then, we can construct an instance of problem $\mathbf{P_K}$ with $K = 2$ as follows. We create a node v' for each pair of p and q, an edge e'_{pq} for node v_i, and put the weight $w(v_i)$ on e'_{pq} as the communication cost between v'_p and v'_q. After that, we create a device D_{pq} with two components C_p and C_q for edge e'_{pq}. Then, we use the constructed graph $G'(V', E', W)$ as an FBP of $|V'|$ components and a system with $|E'|$ devices. Essentially, if we do not consider the weight, $G(V, E)$ is the line graph of $G'(V', E')$. Since every device only has 2 components, the problem constructed is a 2-colocatable problem.

Next, we show the two problem's optimal solutions are equivalent. Suppose that there is a maximum weighted independent set for the MWIS problem. We can use it to find the optimal co-location solution with $K = 2$. Using the optimal set for MWIS, if v_i is chosen, we co-locate two components C_p and C_q and place them on device D_{pq}. This decision saves the most communication energy consumption, implying the optimal solution for the objective function Eq. (34).

Similarly, if we have the optimal solution for a 2-colocatable problem, for the corresponding MWIS problem we can choose node v' of number p, q, if C_p and C_q of the FBP are co-located on device D_{pq} in the optimal solution for the 2-colocatable problem. The optimality of the solution for the 2-colocatable problem also ensures the optimality of the solution in the MWIS problem. □

For P_K where $k \geq 2$, it can be transformed to an MWIS problem where some node contains more than 2 numbers. The above reasoning between an MWIS instance and the optimal solution for a K-colocatable problem still applies. Therefore P_K is NP-Hard for any K.

For each instance of P_A, we can find its upper bound on K by the maximum number of components in FBP that can be co-located together. In this way, we can conclude that the P_A problem is NP-Hard.

6. Co-location graph and selection

We are interested in finding efficient algorithms to solve the co-location problem. From the previous section, we can see that every mapping problem instance has a corresponding graph from which a MWIS could be used to solve the original co-location problem.

In this section, we first show how to construct a co-location graph $G_c = (V_c, E_c, W_c)$ from an FBP to be deployed in a system of IoT devices to depict all co-location options. In G_c, each $v_c \in V_c$ represents a co-location decision for a set of service components. An edge $(u_c, v_c) \in E_c$ represents a conflict between two neighboring co-location decisions.

After the co-location graph construction, we can solve the selection problem by using the MWIS algorithm on the co-location graph. We show three different greedy strategies of MWIS, define a greedy selection framework that could adopt these MWIS algorithms, and then show how we select devices for the remaining service components that have not been mapped yet.

6.1. *Layer based graph construction*

We propose a general construction algorithm to include all co-locations in Algorithm 1. Since not all components connected by links in FBP are co-locatable, we first remove those non-candidate links and keep the list of co-locatable links in L. We also need the devices in the system as input M. M will be used by the algorithm to determine if there is a feasible co-location option by finding a device that can host those service components.

The algorithm checks the feasibility to deploy all service components of the union of $s(v_i)$ and $s(v_j)$ on a single IoT device. If it is feasible to select two co-location vertices at the same time, it creates vertex v_k as a new option to select all co-location options in its generator set at the same time. v_i and v_j are added to the generator set of v_k in order to keep track of how v_k is being created. The newly created vertex v_k is pushed to G and its corresponding layer according to its size of service components. Before the nested loop starts, for each vertex in each layer, it will create edges between it and all neighboring vertices of each generator vertex. In this way, the algorithm builds up a complex relationship of vertices between different layers. Finally, the algorithm finishes the transformation when all layers are settled, leaving the graph G as the co-location graph.

Algorithm 1 Co-Location Graph Construction

Input: A list of co-locatable links L as a connected graph and a system M

Output: Co-location graph $G(V, E, W)$

```
1: $X_k = \emptyset, 1 \leq k \leq |FBP|$.
2: Set layer k = 1
3: Generate a vertex for each $L_{ij}$ in $L$ and add it to $G$ and $X_1$.
4: while $X_k \neq \emptyset$ do
5:   for all co-location vertex $v_i \in X_k$ do
6:     for all $v_i$'s generator $v_j \in g(v_i)$ do
7:       add edge $e_{ik} = (v_i, v_k)$ to G, for every $v_k$ that is $v_j$'s neighbor
8:     end for
9:   end for
10:  for all pairs of co-location vertices $(v_i, v_j)$ in $X_k$ do
11:    if $s(v_i) \cap s(v_j) \neq \emptyset$ then
12:      add distinct edge $e_{ij} = (v_i, v_j)$ to G
13:      if $s(v_i) \cup s(v_j)$ can run on the same device in $M$ then
14:        create new vertex $v_k$ from $v_i$ and $v_j$
15:        if $v_k$ exists in G then
16:          retrieve existing vertex $v_k$ from G
17:        else
18:          add $v_k$ to set $X_{|s(v_k)|-1}$ and G
19:        end if
20:        add $v_i$ and $v_j$ to generator set $g(v_k)$
21:      end if
22:    end if
23:  end for
24:  find the smallest $i > k$ where $X_i \neq \emptyset$
25:  if no such $i$ exists, stop, else set $k = i$
26: end while
```

6.2. *Selection strategies*

In our earlier study [4], we use a simple algorithm that treats every problem as a 2 co-locatable problem, which means every time the algorithm only selects an edge to co-locate. In this work, we take all possible co-location combinations into consideration, and use the solution strategies for the MWIS problem in our selection framework. In [5], researchers have studied three type of strategies and given their corresponding lower bounds. We briefly review them below.

1. GWMAX: the strategy selects each v_i that minimizes the function $W(v_i)/d_{G_i}(v_i)(d_{G_i}(v_i) + 1)$. Once a node v_i is selected, it and its corresponding edges will be eliminated. When there is no edge left in G, the remaining nodes will form a maximum independent set.
2. GWMIN: this strategy selects each v_i that maximizes the function $W(v_i)/(d_{G_i}(v_i) + 1)$. A node v_i will be selected in every iteration, and it will then be eliminated with its neighbors. The selected nodes during this process will return an independent set.
3. GWMIN2 is an extension of GWMIN by using different vertex-selecting rule. It selects each v_i that maximizes the function $W(v_i)/\sum_{w \in N^+_{G_i}(v_i)} W(w)$.

In all strategies described above, v_i represents the i_{th} node chosen from G. G_i is the G after $i - 1$ round of node selection and update. The function $d_{G_i}(v_i)$ determine the degree of v_i in G_i. In GWMIN2, $N_{G_i}(v_i)$ denotes the neighborhood of v_i, and $N^+_{G_i}(v_i)$, $v_i \cup N_{G_i}(v_i)$.

6.3. *Co-location selection framework*

We now present the general selection algorithms. It takes an FBP as input, splits the FBP into several subgraphs, in which every edge is co-locatable, and then builds corresponding co-location graphs. After that, for each graph, it uses the co-location selection strategy to select the maximum weighted independent co-location node set. Then, it selects a device to host all WuClasses in every co-location node in the maximum weighted independent set.

Algorithm 2 Selection Framework

Input: FBP $G(C, L)$ and device system M

Output: A pairing list P of C_i and its deployed device D_k

```
1: $P = \emptyset$
2: split $G$ into a list of sub-graphs $H$
3: add all devices $D$ of $S$ to queue $Q_d$ in descending order of current energy cost
4: generate co-location graph $G'$ for every sub-graphs in $H$ and add them to list $L(G)$
5: for all co-location graph $G_i \in L(G)$ do
6:   select MWIS $I_i$ from $G_i$ with a specific strategy
7:   for all vertex $v \in I_i$ do
8:     if $D = \{D_k | D_k$ can host every $C_i \in s(v)\} \neq \emptyset$ then
9:       select $D_k \in D$ that has the smallest energy
10:      for all $C_j \in s(v)$ do
11:        add pair $(C_j, D_k)$ to $P$
12:      end for
13:      update the current energy cost on $D_k$
14:    end if
15:  end for
16: end for
17: for all component $C_j$ without a deployment target do
18:   if $D = \{D_k | D_k$ can host $C_j\}$ then
19:     select $D_k \in D$ that has the smallest energy
20:     add pair $(C_j, D_k)$ to $P$
21:   end if
22: end for
```

Since the graph construction algorithm has checked the feasibility of creating a vertex v for a co-location graph and the order of selecting co-location nodes to deploy will not impact the total energy saving, we just randomly pick a node to deploy at each round. (However, the order of selecting nodes would affect the maximum energy consumption among all devices; the problem will be investigated in our future work.) On Line 6 of Algorithm 2, we could use any selection strategy that is good for a particular FBP structure or system setting. The flexibility provided by the framework allows a system to dynamically replace selection strategy at run time, which is desirable during the reconfiguration of an intelligent IoT system.

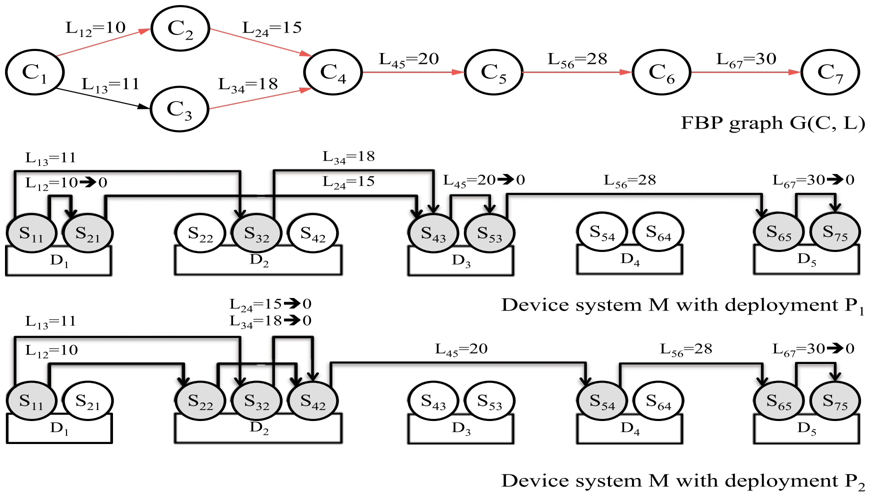


Fig. 3 – Co-location solution comparison.

Fig. 3 shows two mapping decisions P_1 and P_2. P_1 is selected by the MWL algorithm [4]. It merges links L_{67}, L_{45} and L_{12} in order, and totally saves $30+20+10=60$ units of energy cost. P_2 is selected by GWMIN2. To reproduce the selection scenarios of GWMIN2, we use the function $f_k(v_i)$ to represent the current value of $W(v_i)/\sum_{w\in N^+_{G_i}(v_i)} W(w)$ for node v_i in the updated co-location graph after selecting $k-1$ nodes. For the co-location graph in Fig. 3, it is easy to see that v_6 has maximum value $f_1(v_6) = 30/(30+20) = 0.6$. After that, the node v_5 is removed due to the independence constraints. Then, the function value of v_4 is updated. After comparing the value all five nodes $f_2(v_1) = 10/(10+15+33) = 0.172$, $f_2(v_2) = 15/(10+15+33+18+20) = 0.156$, $f_2(v_3) = 18/(15+18+33+20) = 0.209$, $f_2(v_3) = 20/(20+33+15+18) = 0.232$, and $f_2(v_7) = 33/(33+10+15+18+20) = 0.343$, GWMIN2 strategy will find the optimal co-location decision which saves $30+33=63$ units of energy cost.

6.4. *Mapping remaining services*

Since not all components in a FBP may be selected for co-location, we need to select a device for those components that have not been mapped in the co-location decisions. On lines 17–22, the algorithm selects a device with the lowest current energy load for each single component C_i by using the same selection strategy for co-location nodes. In this way, we could achieve a better energy balance on all devices.

For the example of Fig. 2, the mapping decision is to co-locate (C_2, C_3, C_4) on devices D_2 and (C_6, C_7) on device D_5. After that, we still need to select devices for mapping components C_1 and C_5. In this case, since D_1 and D_4 do not have any load, we simply select them for C_1 and C_5 respectively.

7. Performance study

We have implemented the sensor selection framework in Algorithm 2, and used selection strategies including MWL, GWMAX, GWMIN and GWMIN2. As an extended study of our previous work [6], we compare six mapping algorithms shown in Fig. 4. The maximum weight scoring function algorithm (MAXIMUM) is to select the maximum weighted link from the co-location graph, and the one layer maximum weight scoring (ONE LAYER) is to select the maximum weighted link from the one layer co-location graph that only co-locates two neighboring nodes. In this section, we show how we set up the simulation environment, the consideration for determining system parameters and performance metrics, and present the performance comparison for all six algorithms.

7.1. *Simulation setup*

We generate a simulation system with n components and m devices as follows. On each device D_j, we randomly select K different WuObjects as available services on it. K is the upper bound of co-locating size and is viewed as memory constraint for each device. We then use JGraphT to generate different types of flow graphs, including 1000 instances of linear, star or random structures as FBPs with the size half of the total WuClasses in the system. Linear FBPs are common for data transmission applications. Star FBPs are often used for system and environment monitoring application. We also use random FBPs as the topology for general intelligent applications.

In a WuKong system supporting Z-wave communication, a normal information exchange message is about 10 bytes. Including the header of Z-wave protocol, the total size of Z-wave packet is about 40 bytes. For system management messages, the payload is bigger but cannot exceed the maximum size of Z-wave payload which is 64 bytes. Therefore, the size of a WuKong message is about 40–100 bytes in normal cases. Before deployment, we assume application developers are responsible for finding the data rate of each component. In the simulation, we assume data rate of each component is one message per second. Based on these considerations, we use uniform distribution $d_1 = U(40, 100)$ to generate the data volume of each link L_{ij} in an FBP. Even though there are only about 6 – 7 types of messages with different sizes, the normal distribution would randomize the filter rate for components with a threshold. After that, we calculate the corresponding transmitting cost t_{ij} and receiving cost r_{ij} using Eqs. (1) and (2).

WuKong ecosystem aims to provide flexible application deployment and runtime management for large scale IoTs

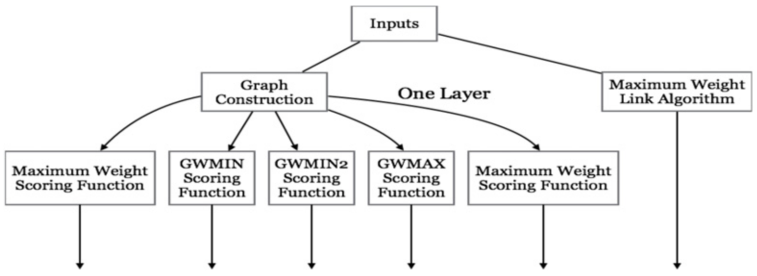


Fig. 4 – Mapping algorithms.

which involves hundreds or even thousands of devices connected. In this paper, the data for 50 components in FBP and systems scales from 100 to 1000 physical devices is reported. On each device, there are a set of K WuClasses with K from 4 to 6.

7.2. *Performance metrics*

In this paper, we compare the total saved energy cost ratio (T-Ratio) and the largest energy cost ratio (L-Ratio) defined as follows:

1. **Total Saved Energy Cost Ratio** (T-Ratio): is the percentage of saved energy cost of the whole FBP with service co-location compared to the FBP original cost without any co-location.
2. **Largest Energy Cost Ratio** (L-Ratio): is the percentage of energy cost in the device with the highest energy cost by applying different algorithms.

Using T-ratios and L-ratios, the higher a ratio value is, the more energy is saved. The energy saving reduces as the probability of service co-location reduces.

7.3. *Performance comparison*

Fig. 5 shows T-ratios in random structure of FBP. It can be seen that the three methods GWMIN, GWMIN2 and GWMAX perform better than the MWL algorithm and its variants MAXIMUM and ONE LAYER. When K is small, the difference of performance between the three methods and MWL algorithm is not so obvious. As K grows to 6, we find that the overall energy saving performance is growing since the likelihood of co-locating multiple WuClasses on a device increases. Moreover, as K grows, the difference of performance between the three methods and MWL algorithms becomes more obvious. The reason is that the MWL greedy algorithm considers the selection as a 2-co-locating problem. It is natural to expect that energy saving depends on the likelihood of service co-location in a system. If an application is a small FBP, or the system has a large number of unique sensors, the chance for sensor co-location, and thus energy saving, is small.

If we compare the result of using the same selection strategy of maximum weight link on two different co-location graphs (fully built co-location graph and one layer co-location graph), we may discover that a fully built co-location graph helps us find better solutions. It is because such a fully built graph includes co-location decision that co-locates multiple services. From Fig. 5, we can see that GWMIN, GWMIN2 and GWMAX algorithms always surpass MAXIMUM, MWL, and ONE LAYER. Therefore, we focus our performance study on GWMIN, GWMIN2, GWMAX and MWL in Fig. 6, which shows the T-ratio and L-ratios on linear structure of FBP's. A similar performance pattern for T-ratio can be seen.

We have also studied how the FBP structure affects T-Ratios. We compare linear, star and random FBP structures in Fig. 7. We see that the three new algorithms and MWL greedy algorithm perform worse in the star structure FBPs when the number of device grows from 100 to 500. The intuition for this fact is that there can be only one choice for the central component in star FBP to co-locate with. That means there only exists one co-location in star shape FBP. Moreover, MWL has a relatively bad performance in all shapes of FBP structure for all numbers of devices. This is because the MWL algorithm can only consider a single edge of FBP in each round.

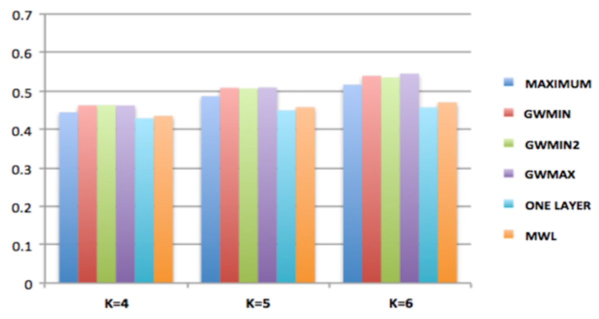
Beside the metrics for energy saving, we have also studied the execution times for different algorithms. It is an important factor if we want to deploy an application with many services. In Table 1, we show the performance of algorithms in six size settings. Each row of (n, m) shows the system with n components and m devices. For each case, we show the average execution time for each algorithm. We can see that the execution time grows linearly for the first three cases. But it grows to more than 192 ms in the last case, which is because the combination of selections grows exponentially. In Table 2, we study how K affects the execution time. The greedy algorithms take longer to consider more co-location decisions as K increases while the MWL uses about the same time.

Table 1 – Scalability with $K = 4$ and different (n, m).

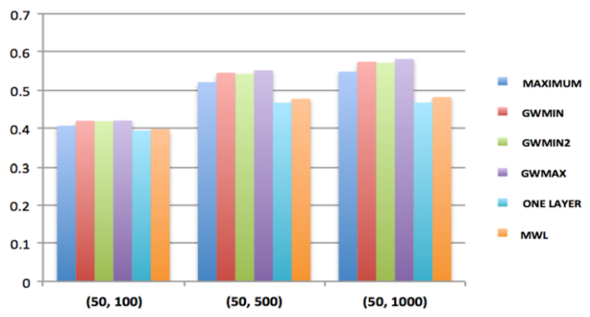
(n, m)	GWMIN	GWMIN2	GWMAX	MWL
(50, 100)	3.182 ms	3.223 ms	3.085 ms	0.07 ms
(50, 200)	9.126 ms	9.172 ms	8.852 ms	0.104 ms
(50, 300)	17.454 ms	18.554 ms	15.855 ms	0.154 ms
(50, 400)	32.254 ms	27.347 ms	22.643 ms	0.136 ms
(50, 500)	49.856 ms	34.809 ms	31.794 ms	0.185 ms
(50, 1000)	192.554 ms	127.458 ms	123.264 ms	0.346 ms

Table 2 – Scalability with $(n, m) = (50, 100)$ and different K.

K	GWMIN	GWMIN2	GWMAX	MWL
4	21.73 ms	18.525 ms	13.923 ms	0.303 ms
5	31.631 ms	24.571 ms	18.776 ms	0.213 ms
6	44.669 ms	32.212 ms	28.955 ms	0.196 ms

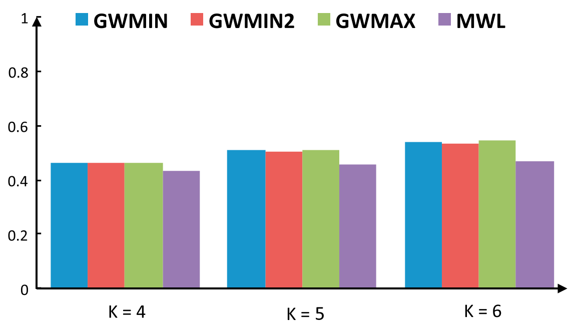


(a) T Ratio Comparison for random FBP $(n, m) = (50, 100)$.

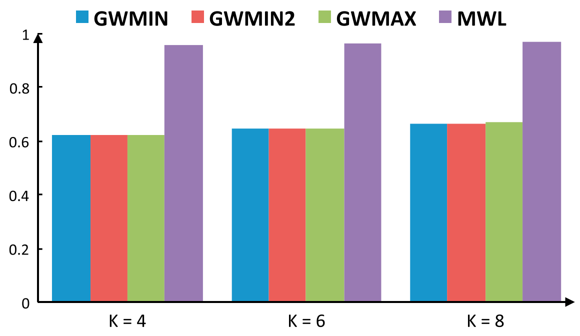


(b) T Ratio Comparison for random FBP $K = 4$.

Fig. 5 – T-Ratios for different selection algorithms.

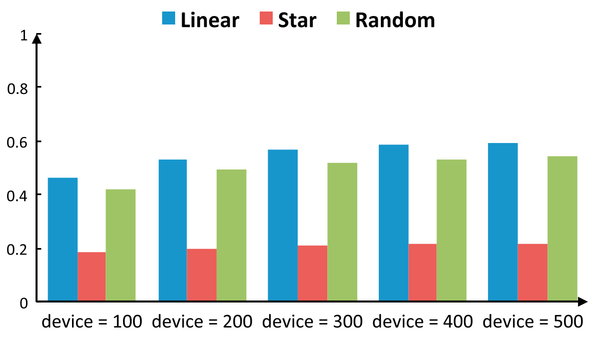


(a) T-Ratio.

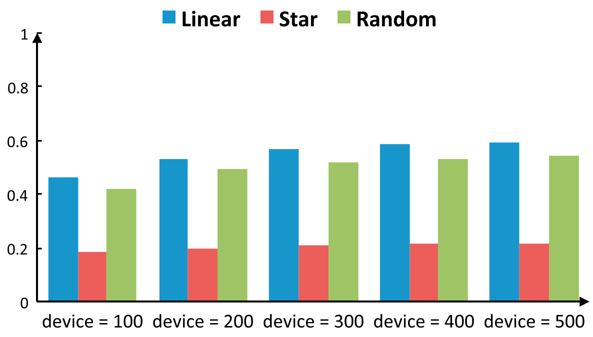


(b) L-Ratio.

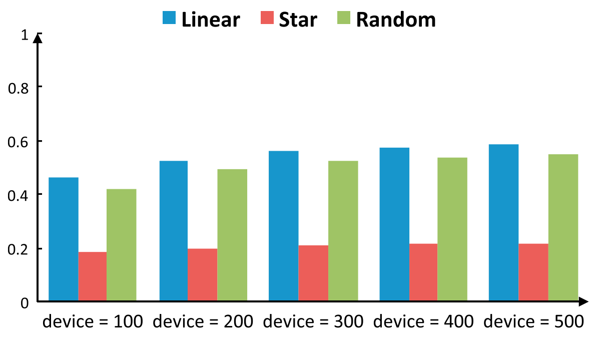
Fig. 6 – Performance of different K sizes for linear FBP's.



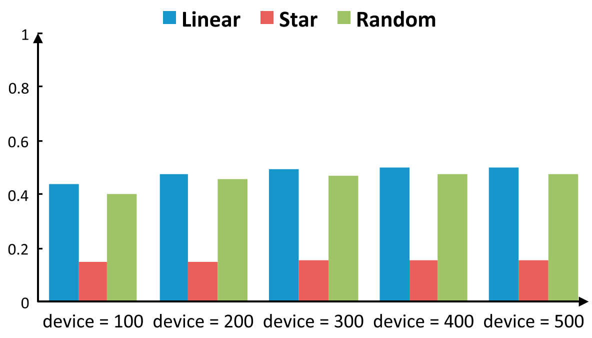
(a) GWMIN algorithm.



(b) GWMIN2 algorithm.



(c) GMAX algorithm.



(d) MWL greedy algorithm.

Fig. 7 – T-Ratios for different FBP structures.

In summary, from the performance of energy saving and reasonably short computation time, we believe the co-location consideration is good for many IoT systems that need to support runtime application mapping, deployment and reconfiguration in a smart IoT environment.

8. Conclusion

This paper presents an energy sentient methodology for selecting and deploying flow-based IoT applications on sensor devices. Since energy is one of the most important resources for running IoT devices, we propose a mapping strategy that tries to minimize the total energy cost for communication by co-locating neighboring services on the same node. We have modeled the co-location problem on multi-hop networks as a quadratic programming problem, so that it can be solved by integer programming. For single-hop networks, we identify co-locatable components of an FBP to construct a co-location graph, and develop the selection framework using efficient MWIS algorithms to decide service co-locations. Our simulation study shows that the MWIS algorithms can save 10% more communication energy than our previous solution.

Acknowledgments

This work was partially supported by Taiwan Ministry of Science and Technology, National Taiwan University and Intel Corporation under Grants MOST102-2911-1-002-001 and NTU103R7501.

REFERENCES

[1] K.-J. Lin, N. Reijers, Y.-C. Wang, C.-S. Shih, J.Y. Hsu, Building smart M2M applications using the WuKong profile framework, in: 2013 IEEE International Conference on Green Computing and Communications and IEEE Internet of Things and IEEE Cyber, Physical and Social Computing, 2013, pp. 1175–1180.

[2] N. Reijers, K.-J. Lin, Y.-C. Wang, C.-S. Shih, J.Y. Hsu, Design of an intelligent middleware for flexible sensor configuration in M2M systems, in: SENSORNETS, 2013, pp. 41–46.

[3] J. Wang, D. Li, G. Xing, H. Du, Cross-layer sleep scheduling design in service-oriented wireless sensor networks, IEEE Trans. Mob. Comput. 9 (11) (2010) 1622–1633, http://doi.ieeecomputersociety.org/10.1109/TMC.2010.124.

[4] Z. Huang, K.-J. Lin, A. Han, An energy sentient methodology for sensor mapping and selection in IoT systems, IEEE International Symposium on Industrial Eletronics, 2014.

[5] S. Sakai, M. Togasaki, K. Yamazaki, A note on greedy algorithms for the maximum weighted independent set problem, Discrete Appl. Math. 126 (2–3) (2003) 313–322.

[6] Z. Huang, K.-J. Lin, S.-Y. Yu, J.Y.-j. Hsu, Building energy efficient internet of things by co-locating services to minimize communication, in: Proceedings of the 6th International Conference on Management of Emergent Digital EcoSystems, MEDES'14, ACM, New York, NY, USA, 2014, pp. 18:101–18:108, http://dx.doi.org/10.1145/2668260.2668270. URL: http://doi.acm.org/10.1145/2668260.2668270.

[7] Z. Huang, W. Jiang, S. Hu, Z. Liu, Effective pruning algorithm for QoS-aware service composition, in: Commerce and Enterprise Computing, 2009. CEC'09. IEEE Conference on, 2009, pp. 519–522, http://dx.doi.org/10.1109/CEC.2009.41.

[8] W. Jiang, C. Zhang, Z. Huang, M. Chen, S. Hu, Z. Liu, QSynth: A tool for QoS-aware automatic service composition, in: IEEE International Conference on Web Services (ICWS), 2010, pp. 42–49, http://dx.doi.org/10.1109/ICWS.2010.38.

[9] T. Yu, Y. Zhang, K.-J. Lin, Efficient algorithms for web services selection with end-to-end QoS constraints, ACM Trans. Web, 1 (1), http://dx.doi.org/10.1145/1232722.1232728.

[10] Q. Wang, M. Hempstead, W. Yang, A realistic power consumption model for wireless sensor network devices, in: 3rd IEEE Conference on Sensor and Ad Hoc Communications and Networks, SECON'06, vol. 1, 2006, pp. 286–295, http://dx.doi.org/10.1109/SAHCN.2006.288433.

[11] W. Ye, J. Heidemann, D. Estrin, An energy-efficient mac protocol for wireless sensor networks, in: Proceedings of IEEE INFOCOM, vol. 3, 2002, pp. 1567–1576, http://dx.doi.org/10.1109/INFCOM.2002.1019408.

[12] G. Lu, N. Sadagopan, B. Krishnamachari, A. Goel, Delay efficient sleep scheduling in wireless sensor networks, in: INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE, vol. 4, 2005, pp. 2470–2481, http://dx.doi.org/10.1109/INFCOM.2005.1498532.

[13] W. Ye, J. Heidemann, D. Estrin, Medium access control with coordinated adaptive sleeping for wireless sensor networks, IEEE/ACM Trans. Netw. 12 (3) (2004) 493–506, http://dx.doi.org/10.1109/TNET.2004.828953.

[14] S. Olariu, I. Stojmenovic, Design guidelines for maximizing lifetime and avoiding energy holes in sensor networks with uniform distribution and uniform reporting, in: INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings, 2006, pp. 1–12, http://dx.doi.org/10.1109/INFOCOM.2006.296.

[15] P. Pradhan, V. Baghel, G. Panda, M. Bernard, Energy efficient layout for a wireless sensor network using multi-objective particle swarm optimization, in: Advance Computing Conference, 2009. IACC 2009. IEEE International, 2009, pp. 65–70.

[16] Y.-C. Wang, W.-C. Peng, Y.-C. Tseng, Energy-balanced dispatch of mobile sensors in a hybrid wireless sensor network, IEEE Trans. Parallel Distrib. Syst. 21 (12) (2010) 1836–1850, http://dx.doi.org/10.1109/TPDS.2010.56.

[17] C.-Y. Chang, H.-R. Chang, Energy-aware node placement, topology control and mac scheduling for wireless sensor networks, Comput. Netw. 52 (11) (2008) 2189–2204, http://dx.doi.org/10.1016/j.comnet.2008.02.028.

[18] Z. Huang, K.-J. Lin, C. Li, S. Zhou, Communication energy aware sensor selection in iot systems, in: 2014 IEEE and Internet of Things (iThings/CPSCom), 2014.

[19] S.-Y. Yu, Z. Huang, C.-S. Shih, K.-J. Lin, J. Hsu, Qos oriented sensor selection in iot system, in: 2014 IEEE and Internet of Things (iThings/CPSCom), 2014.

[20] M. Halldórsson, J. Radhakrishnan, Greed is good: Approximating independent sets in sparse and bounded-degree graphs, in: Proceedings of the Twenty-sixth Annual ACM Symposium on Theory of Computing, STOC'94, ACM, New York, NY, USA, 1994, pp. 439–448.

[21] W. Heinzelman, A. Chandrakasan, H. Balakrishnan, An application-specific protocol architecture for wireless microsensor networks, IEEE Trans. Wirel. Commun. 1 (4) (2002) 660–670, http://dx.doi.org/10.1109/TWC.2002.804190.

[22] A. Billionnet, A. Faye, A lower bound for a constrained quadratic 01 minimization problem, Discrete Appl. Math. 74 (2) (1997) 135–146, http://dx.doi.org/10.1016/S0166-218X(96)00026-1. URL: http://www.sciencedirect.com/science/article/pii/S0166218X96000261.

[23] J. Hastad, Clique is hard to approximate within n1-epsiv, in: Foundations of Computer Science, 1996. Proceedings, 37th Annual Symposium on, 1996, pp. 627–636, http://dx.doi.org/10.1109/SFCS.1996.548522.