Research Article

International Journal of Distributed Sensor Networks

International Journal of Distributed Sensor Networks
2018, Vol. 14(8)
© The Author(s) 2018
DOI: 10.1177/1550147718795355
journals.sagepub.com/home/dsn
SAGE

A task allocation strategy for complex applications in heterogeneous cluster–based wireless sensor networks

Xiang Yin[1], Kaiquan Zhang[1], Bin Li[1], Arun Kumar Sangaiah[2] and Jin Wang[1,3]

Abstract
To a wireless sensor network, cooperation among multiple sensors is necessary when it executes applications that consist of several computationally intensive tasks. Most previous works in this field concentrated on energy savings as well as load balancing. However, these schemes merely considered the situations where only one type of resource is required which drastically constrains their practical applications. To alleviate this limitation, in this article, we investigate the issue of complex application allocation, where various distinctive types of resources are demanded. We propose a heuristic-based algorithm for distributing complex applications in clustered wireless sensor networks. The algorithm is partitioned into two phases, in the inter-cluster allocation stage, tasks of the application are allocated to various clusters with the purpose of minimizing energy consumption, and in the intra-cluster allocation stage, the task is distributed to appropriate sensor nodes with the consideration of both energy cost and workload balancing. In so doing, the energy dissipation can be reduced and balanced, and the lifetime of the system is extended. Simulations are conducted to evaluate the performance of the proposed algorithm, and the results demonstrate that the proposed algorithm is superior in terms of energy consumption, load balancing, and efficiency of task allocation.

Keywords
Wireless sensor network, complex application allocation, hierarchical architecture, heterogeneity, load balancing

Date received: 22 April 2018; accepted: 18 July 2018

Handling Editor: Mohamed Elhoseny

Introduction

In the last decade, with the development in technology of micro-electronics, digital electronics design, as well as advances in low-power wireless communication and network technique, a new type of network, wireless sensor network (WSN), has emerged.[1] In general, a WSN is composed of a large number of sensor nodes which are endowed with detection, computation, and communication capabilities. The emergence of WSN has greatly extended the application areas over traditional sensors. In fact, by grouping numerous sensors into a connected network, the end-users can obtain much more accurate data of a target that is in a remote and/or hostile environment. Therefore, WSNs have been widely used in many applications such as object

[1]College of Information Engineering, Yangzhou University, Yangzhou, China
[2]School of Computing Science and Engineering, Vellore Institute of Technology, Vellore, India
[3]College of Computer & Communication Engineering, Changsha University of Science & Technology, Changsha, China

Corresponding author:
Jin Wang, College of Information Engineering, Yangzhou University, 196 West Hua Yang Road, Yangzhou 225127, Jiangsu Province, China.
Email: jinwang@yzu.edu.cn

detection and tracking,[2] privacy and security protection,[3] natural disaster rescue,[4] health care,[5] and so on.

In many real-world scenarios, a WSN will face applications which may require large amount of information collection, considerable computation operations, and intensive communication overhead. In contrast, as a tiny, low-cost and battery-powered device, a single sensor node usually has limited resource and can perform relatively simple action. Consequently, in order to successfully execute a complicated application, various sensors should work cooperatively. In particular, by task decomposition and allocation, a difficult task can be divided into several uncomplicated parts, and each sub-task is assigned to a sensor node which can handle it independently. A typical example is the Third Generation Surveillance System (3GSS) where a number of cameras are connected with a network.[6] To such a system, a common application is to detect, classify, localize, and recognize an object; it involves computationally intensive operations which can hardly be performed by a single sensor node. A promising way to resolve this problem is to decompose the complicated application into simple tasks and distribute them among specific sensors. In such a way, various cameras can collaboratively accomplish the complex application. Thus, task allocation has a vital influence on the overall performance of a WSN, and the development of efficient strategies for collaborative task distribution has attracted much attention in recent research.

Although the issue of task allocation has been widely studied and well addressed in conventional distributed systems, including multiprocessor systems,[7] cloud computing,[8] social networks,[9] and so on, the problem has distinctive features in WSNs. As sensor nodes in WSNs are usually powered by batteries and the power is irreplaceable, they may be prone to be invalid due to energy exhaustion. If the energy of some sensor nodes is depleted in short term, critical data cannot be collected in the sensing area which may seriously weaken the function of a WSN. A more disastrous effect is that if too many nodes are dead, the communication link will become collapsed and the whole network may be out of work. On the other side, the sensor nodes in a WSN should consume the energy uniformly, and a poor workload balancing may lead to some sensor nodes deplete their energy much earlier than others. Therefore, how to reduce and balance energy consumption becomes a desirable design goal to WSNs.[10,11] The issue should also be taken into account for task allocation in WSNs. In fact, numerous studies have been conducted in this field.[12–14] Most existing works focus on distributing computation and communication tasks rationally among sensor nodes with the purpose of extending network lifetime by load balancing and energy conservation. These approaches model task allocation as a multi-objective optimization problem which has been proved to be NP-hard; thus, heuristic-based algorithms are widely presented to solve this problem in polynomial time.

In general, a WSN is considered to be an application-oriented network, that is it is created and deployed to implement a specific application. In real practice, an application normally requires detecting and gathering various types of information. In this article, this kind of applications is referred to as *complex* applications. Formally, a *complex* application is composed of multiple tasks each of which can be executed if some specific information is collected and processed. Actually, most real-world applications to WSNs are *complex* applications, one typical example is the environment monitoring system. In the scenario of an environment monitoring system, sensor nodes are deployed in the target area to detect environmental conditions. Considering an application that starts running on such a WSN to determine whether the environmental status is normal in a particular region, different sorts of data including temperature, humidity, pressure, and so on should be detected and sent back to the sink node for further analysis. Consequently, to perform such a complex application, various types of information are required. For a WSN, it should have the capacity to collect, process, and transmit distinctive kinds of data. As a limited resource and low-cost unit, a sensor node is usually equipped with only one sensing device, thus the whole WSN comprises a great deal of sensor nodes which are integrated with multiple sensing elements of different types. This kind of WSN is referred to as a heterogeneous WSN, a typical example of a heterogeneous WSN is shown in Figure 1, where sensor nodes with distinctive sensing devices are marked by different notations. Compared with homogeneous WSNs in which sensor nodes have identical or similar functionalities, heterogeneous WSNs have their unique characteristics. On one hand, all the nodes in a heterogeneous WSN can share underlying protocols or standards, such as the structure of a frame, the implementation of modulation and demodulation, and data transmission protocol. On the other hand, to high-level applications, such as task allocation, heterogeneous WSNs are quite different from homogeneous WSNs. Previous studies mainly focus on distributing tasks in a homogeneous WSN. Nevertheless, these methods cannot be directly adopted to solve the problem of assigning *complex* applications in heterogeneous WSNs. The challenge stems from the fact that to distribute a *complex* application in a heterogeneous WSN, sensor nodes with particular resources must be determined. To this end, an allocation algorithm should consider not only the conventional optimization metrics, such as energy consumption and latency, but also the sensing capability of a node. The node that has the required sensing capability can be regarded as the candidate for the task, otherwise it cannot undertake corresponding task even if it has superior properties in energy conservation or computation speed. In summary, as a common issue in practice, *complex* application allocation in heterogeneous WSNs is different from traditional task allocation problem in WSNs and should be concerned with.

To WSNs, one core issue is energy conservation and/or network lifetime extension; it is not an exception to *complex* application allocation in heterogeneous WSNs. Since communication is the most energy costly operation in WSNs, and the communication energy expenditure is proportional to the distance (normally second power) between the two nodes, a node is usually constrained to communicate directly with other nodes within a limited range. In order to send the collected data to the sink node with low power, a hierarchical communication network should be formed—the most popular and effective method is clustering.[15–17] By dividing the whole WSN into small parts and clustering multiple nodes into a group, a sensor node can communicate with the sink node through the cluster head (CH) in a multihop manner, which can significantly reduce power dissipation. Therefore, the *complex* application allocation should be considered in a clustered WSN.

With this background, in this article, we design a *complex* application allocation mechanism in heterogeneous clustered WSNs, which is claimed as a main contribution of this article. In comparison with previous works, the presented mechanism assumes that a *complex* application can be divided into various tasks and each task requires various kinds of resources, and accordingly, a sensor node is endowed with a sensing resource that enables it to accomplish a corresponding action. Meanwhile, the WSN has a hierarchical structure and all the sensor nodes are grouped into clusters and each cluster has a leader. The goal is to find an optimal scheme that can distribute the *complex* application to proper sensor nodes with the purpose of minimizing energy dissipation and balancing the workload while meeting the application's requirement for multiple resources. A two-stage application allocation algorithm is presented to achieve the goal. First, a dynamic programming (DP)-based algorithm is developed to assign the partitioned application to clusters according to the aim of decreasing energy consumption. Then, inside each cluster, an algorithm which considers both the energy dissipation and the workload balancing is developed to distribute the task to sensor nodes. In addition, the algorithm provides parameters for end-users to adjust the importance of distinctive objectives. Finally, extensive simulations are conducted and the results demonstrate that the proposed algorithm is feasible and efficient.

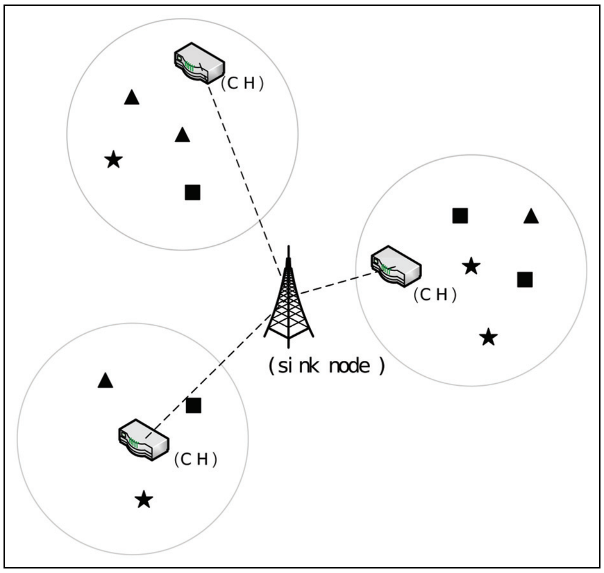

Figure 1. An example of heterogeneous WSN.

Related work

As an event-driven system, a WSN is designed and deployed to accomplish specific applications. As a tiny and resource limited unit, a sensor node can hardly execute a complicated application independently, an effective method to resolve this problem is to divide the application into small parts and assign them to appropriate sensors. Thus, task allocation has attracted a great deal of interests and becomes a hot spot in WSN.

As mentioned above, the issue of task allocation in WSN is NP-hard which makes it infeasible to be solved optimally within limited time, and therefore, heuristic-based approaches are essential to handle this problem in polynomial time. Yu and Prasanna[18] studied the problem of allocating an epoch-based real-time application in a homogeneous WSN in which each sensor node was equipped with discrete dynamic voltage scaling (DVS). They first proposed an Integer Linear Programming (ILP) formulation which can obtain the optimal solution with intensive computation. Furthermore, they presented a three-phase heuristic which can be adopted to large-scale systems. One feature of this method is that the voltage levels of tasks can be adjusted with the goal to maximize the system lifetime. Abdelhak et al.[19] proposed an energy-balancing task scheduling and allocation heuristic with the aim of prolonging the network lifetime (EBSEL). This approach is divided into two phases. In phase 1, the tasks are grouped in order to reduce communication traffic, and in phase 2, the task group which requires the highest computational energy is distributed to the nodes with the highest remaining energy. In so doing, energy balancing is achieved and the lifetime is extended. Shen et al.[20] are concerned with a specific application—digital signal processing (DSP) application—in WSNs which is defined as the energy-driven partitioning (EDP) problem. They formulated the EDP problem as a synchronous dataflow (SDF) graph and developed an algorithm that can minimize the overall energy consumption by analyzing the pattern of internal data exchange rates in the application. Similarly, Yu et al.[21] also modeled the workload distribution problem in a WSN as an SDF graph partition problem. They proposed an optimal online task allocation algorithm to maximize the network lifetime by taking the energy cost of sensing, computing, communicating, and sleeping into account. They proved that for each sensor node, by using at most two of the important partition cuts with proper weights, it can obtain the optimal solution. Li et al.[22] investigated scheduling tasks with a predefined deadline. A three-phase task scheduling (TPTS) scheme is presented that takes into account an integrated goal including overall energy dissipation reduction, workload balance, and latency constraint. First, the whole deadline of all tasks is divided into subdeadlines and each task is endowed with a subdeadline. Then, an energy-load tunable graph partitioning algorithm is developed to decompose the task graph into several disjoint partitions and distribute them to distinct clusters. Finally, a contention-aware scheduling scheme is employed to allocate tasks to appropriate sensors. One common flaw of these approaches is that they only consider homogeneous WSNs and thus cannot be utilized to address our case.

Another category of popular approaches exploits bio-inspired meta-heuristic algorithms. Compared with the aforesaid traditional heuristic approaches, these methods adopt bionic swarm intelligence which allows them to obtain good-quality solutions with high efficiency. Jin et al.[23] investigated task mapping and scheduling in multihop wireless networks (MHWNs). They designed a hybrid function that takes both network lifetime and schedule length into account, then an adaptive intelligent scheme that is based on genetic algorithm (GA) is proposed to provide real-time guarantees. Compared with GA, the particle swarm optimization (PSO) algorithm has the advantage of simplicity of implementation and the capability to converge to a reasonably good solution quickly. As a consequence, it has been widely applied in task allocation in WSNs. Guo et al.[24] considered the case that when some sensors become out of work, how to transfer the remaining tasks on these nodes to other sensors. They proposed a self-adapted task scheduling strategy which is inspired by the multi-agent system theory; in this strategy, they developed an effective discrete PSO algorithm with a well-designed particle position code and a hybrid fitness function that takes into account the task execution time, the energy expenditure, and network balance. Furthermore, in Guo et al.,[25] they proposed a real-time fault-tolerant task allocation algorithm (FTAOA). Likewise, to maximize the network lifetime, a discrete PSO algorithm is adopted. Various metrics, involving task execution time, energy dissipation, load balance, and the reliability cost of the network, are integrated into the fitness function. Compared with other similar work, the main difference of FTAOA is that it uses primary/backup technology to achieve fault tolerance in task allocation which makes WSNs sustain functionalities even when some nodes are failed or out of power. In addition, passive backup copies overlapping technology is employed to further improve the performance. One shared defect of these bio-inspired approaches is that they would easily get stuck in local optimum and the computational complexity may grow exponentially with the size of the problem which makes them not suitable for large-scale systems.

All the aforementioned works focus on improving the applicability and performance of task distribution in homogeneous WSNs where a WSN is dedicated for a single sensing task. However, due to the difficulty and high cost of deployment, for example, establishing a WSN in a hostile environment to collect information, WSNs are preferred to have the ability of performing various kinds of tasks. Many researchers begin to concern about this issue in recent years, one promising solution is the software-defined wireless sensor networks (SD-WSNs).[26] A typical SD-WSN is composed of a number of sensor nodes whose functionalities can be dynamically configured by running different programs. Specifically, such software-defined sensor nodes are able to conduct different sensing tasks according to the software that is activated. In comparison with conventional WSNs, SD-WSNs is more versatile, flexible, and easy to manage. Following the line, Zeng et al.[27] studied the minimum energy sensor activation problem with the consideration of guaranteed quality of sensing tasks in SD-WSNs. They formulated the problem as a mixed-integer with quadratic constraints programming model and linearized it into a mixed-ILP problem to further lower the computation complexity. Then an efficient online algorithm is proposed to deal with the problem. Although SD-WSNs are capable of performing multiple distinctive types of tasks, each individual node becomes more complicated and costly as it requires larger storage space and a control procedure that can coordinate different functions of the node.

Problem statement

The goal of distributing a complex application is to find out a scheme of allocating all the tasks in the application to appropriate sensor nodes while extending the network lifetime. The network lifetime has diverse definitions by taking into account the number of died nodes, sensing coverage, connectivity, and so on.[28] Although these definitions are concerned with distinctive metrics, they are all related to two elements: energy consumption and workload balancing. Therefore, in this article, the purpose of our scheme is to minimize energy dissipation as well as balance workload. To a regular sensor node S_i, its energy cost contains communication energy E^i_{comm} and computation energy E^i_{comp}, and the energy consumption of a WSN E_{WSN} is the sum of the energy consumed by all the sensor nodes that are assigned with tasks, which can be expressed as follows

$$E_{WSN} = \sum_{i=1}^{n} (E^i_{comm} + E^i_{comp}) \tag{1}$$

$$E^i_{comm} = lE_{elec} + l\varepsilon_{fs}d^2 \tag{2}$$

$$E^i_{comp} = P^i t^i \tag{3}$$

where l is the amount of data to be sent and d is the transmitting distance, and E_{elec} and ε_{fs} are hardware-dependent parameters that hinge on the mode of digital coding, modulation, acceptable bit-error rate, and so on, P denotes processor's average power consumption for computation and t is the time required for processing the task.

In addition, load balancing plays a key impact on the network lifetime. Since the load of a sensor node S_i is proportional to its energy dissipation, and the workload balancing is a comprehensive measurement to the energy consumption state of all the nodes in a WSN, it can be represented as follows

$$B_{WSN} = \sqrt{\frac{1}{N}\sum_{i=1}^{n} (E^i_{resid} - avg(E^i_{resid}))^2} \tag{4}$$

where E^i_{resid} is the residual energy of node S_i, and $avg(E^i_{resid})$ denotes the average residual energy of all the nodes in the WSN.

According to the above descriptions, the problem of *complex* application allocation in a heterogeneous clustered WSN can be formulated as follows

$$\begin{aligned} &\underset{\lambda\in\Lambda}{\operatorname{argmin}}\,(E_{WSN} \quad \text{and} \quad B_{WSN}) \\ s.t. \quad & R_{T_k} = \sum_{S_i\in G_{T_k}} C_{S_i}\cdot h(S_i) \quad \text{for all} \quad T_k \in A \end{aligned} \tag{5}$$

where Λ is the set of all feasible solutions, G_{T_k} is the cluster to which task T_k is assigned, $h(S_i)$ is a binary variable, and $h(S_i) = 1$ implies that sensor node S_i will contribute its specified type of resource to the task, otherwise it equals 0. Note that, here the symbol "=" is the vector equality which means that each element of the two vectors is the same. It implies that a task must be allocated to a cluster that can provide all the demanded resources. From the above model, we can find that this is a multi-objective optimization problem. Nevertheless, these two objectives are not compatible or even conflicting to some extent, and some trade-offs must be made to achieve a solution with good performance.

Problem formulation

The network model

In general, a WSN is composed of a number of sensor nodes that are connected by wireless links. All the nodes are randomly dispersed in the target region to detect or monitor the occurrence of specific events. Accordingly, the network can be modeled by a connected undirected graph $G = (S, E)$, where $S = \{s_1, s_2, \ldots, s_n\}$ denotes the set of sensor nodes, and $E = \{e_{ij}|i,j = 1, 2, \ldots\ldots, n\}$ represents the communication links between a pair of nodes. Each sensor node can send and receive messages within a certain range; thus, $e_{ij} = 1$ if node j is within the communication range of node i; otherwise, it is 0. In this work, a WSN has a hierarchical structure that consists of three types of devices, sink node (base station), CHs and normal sensor nodes. For the sink node, it receives a complex application from end-users and decomposes the application into several tasks. Here, we are not concerned with the strategy of dividing a complex application; instead, we put emphasis on distributing these tasks to various clusters with the purpose of minimizing energy dissipation. CH is responsible for selecting proper sensor nodes to cooperatively accomplish the task; the main goal is to reduce energy consumption as well as balance workload. To the normal sensor nodes that are assigned with a task, they collect data and send them (raw or processed) back to the CH. There is only one sink node in the WSN while the WSN is grouped into multiple clusters and each cluster has a CH. A CH communicates with the sink node through one-hop link and within a cluster, normal nodes can transmit data to the CH directly. We also assume that a CH is endowed with more resources, in terms of computation and power supply, than ordinary nodes. Meanwhile, each sensor node is equipped with a GPS device that makes the location information is locally available within a cluster.

The application and the node model

In fact, the problem of task allocation has been widely discussed in WSNs, most of existing works do not consider the characteristics of the allocated tasks. They all assume that only computation and communication resources are required to execute a task. Therefore, a task can be distributed to any sensor nodes, the difference between distinctive allocation solutions is the different energy cost. However, in many real-world scenarios, to complete an application, various types of resources may be required. Here, we name this kind of applications as *complex* applications. In particular, when allocating a complex application, we need to consider not only the computation and energy features at each sensor node but also the matching between the resource available at each node and the resources required by the application. In our work, a complex application is composed of several tasks, these tasks are independent and have no precedent constraints, that is, all the tasks can be assigned and executed simultaneously. To a task, it requires various types of resources. We assume that only when all of the required resources are fulfilled, the task is deemed to be successfully accomplished. Formally, a complex application can be denoted as a set of tasks $A = \{T_1, T_2, \ldots, T_m\}$, each task $T_k \in A$ is associated with a set of specific resources and can be represented as a tuple (T_k, R_{T_k}) where k is the index of the task and $R_{T_k} = \{r^1_{T_k}, r^2_{T_k}, \ldots, r^p_{T_k}\}$ indicates the required resources for completing task T_k. p is the total number of resource type in the system and $r^q_{T_k} \in \{0, 1\}$ represents whether resource q is necessary for task T_k. Our application model is illustrated in Figure 2.

Another notable feature to our framework is the characteristics of individual sensor nodes. Most previous works assume that the nodes in a WSN are similar with respect to functionalities; they all can perform computation and communication tasks. Thus, they concentrate on how to assign computation and communication load among sensors. On the other side, although the term "Heterogeneity" has been introduced in some existing literatures, they merely referred to that the nodes had distinct initial energy state and computation capacity. In practice, to distribute a complex application, the sensor nodes in a WSN are endowed with various types of resources that can be used for detecting and collecting information of the target. In this work, this kind of WSN is regarded as a heterogeneous WSN. In particular, in a heterogeneous WSN, each sensor node is equipped with a specific type of sensing device, such as ultrasonic sensor and photoelectric sensor. As a consequence, a sensor node S_i can be indicated as a tuple (i, C_{S_i}) where i is the index of the sensor node and C_{S_i} is a binary vector which represents the type of resource owned by S_i, $C_{S_i} = \{c^1_{S_i}, c^2_{S_i}, \ldots, c^p_{S_i},\}$, $c^l_{S_i} \in \{0, 1\}$, and $\sum_{l=1}^{p} c^l_{S_i} = 1$, that is, a sensor node has only one particular type of resource. Obviously, the sensor node model here is intuitively different from that in existing works. Accordingly, the term heterogeneity in our study refers to the diversity of sensing devices possessed by sensor nodes. On the other side, we

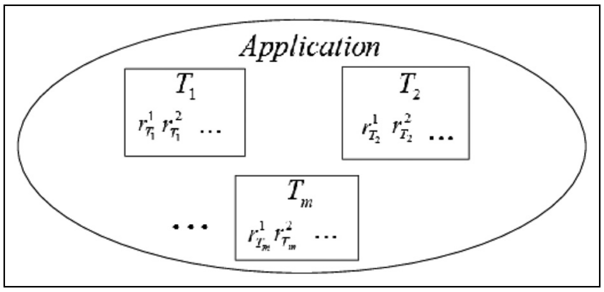
Application
T_1
$r^1_{T_1}\, r^2_{T_1}\ \ldots$
T_2
$r^1_{T_2}\, r^2_{T_2}\ \ldots$
$\ldots$
T_m
$r^1_{T_m}\, r^2_{T_m}\ \ldots$

Figure 2. Application model.

assume that to the sensor nodes with the same resource, they are identical in computation capacity and initial power supply. Thus, if allocating a task to two nodes that both can provide the required resource, the main difference comes from the energy cost for communication and the residual power state of the nodes. Another assumption is that a complex application will be assigned to a set of sensor nodes with no duplicate units, that is, a sensor node can only join one task.

The proposed algorithm

Since the task allocation issue has been proven to be an NP-complete problem, in this article, we provide a heuristic algorithm to solve it in polynomial time. The algorithm can be briefly divided into two stages, namely, inter-cluster distribution and intra-cluster distribution. The first scheme is run on the sink node and its aim is to allocate the tasks in an application to cooperative clusters with minimum power cost. After this, in each cluster, the CH is responsible for distributing the task to proper sensor nodes under the condition that the task can be successfully executed while reducing energy consumption as well as balancing workload.

The inter-cluster application allocation algorithm

In our case, a WSN has a hierarchical structure and all the sensor nodes are grouped into multiple clusters. A *complex* application consists of various tasks each of which requires distinctive resources, and only when all of the tasks are executed successfully, the *complex* application is deemed to be finished. For a task, it can be assigned to any cluster that will provide the required resources; thus, the possible solutions of distributing the *complex* application to clusters are numerous. Among these solutions, the one with the minimum energy consumption is preferred. We name this problem as inter-cluster application allocation.

To a cluster, if it is assigned with a task $T_j \in A$, the energy cost contains two parts: computation energy and communication energy. Since the sensor nodes with the same kind of sensing device are identical in the features of computation and communication, the energy dissipation is mainly determined by the distance between the nodes and the CH as shown by equation (2). Thus, to minimize the energy cost of a single task, it should be allocated to a cluster in which the sensor nodes with the necessary resources are closest to the head. On the other side, our aim is to reduce the energy consumption of the whole application as much as possible under the constraint that a cluster can undertake only one task, the ideal state is allocating all tasks to their lowest energy cost clusters with non-overlapping. However, this situation is hard to achieve since there may exist a cluster that can provide the lowest energy cost for more than one task. In such a case, the allocation of one task will influence the result of other tasks and further alter the energy cost of the application assignment. This problem can be modeled as a sequential decision-making problem. In such a framework, an agent makes a series of decisions to interact with the environment, a decision corresponds to a particular action that will change the state of the environment and after performing the action, a reward can be obtained. In general, a specific objective is set, for example, maximizing the sum of the received rewards. To our issue, the sink node can be regarded as the agent that is responsible for allocating the *complex* application, the action is distributing each task to a cluster, the reward is energy dissipation, and the aim is to find out a task allocation scheme which can minimize the total energy consumption. As a conventional problem, several approaches have been developed to solve it. In this article, we adopt and modify the DP algorithm,[29,30] a popular method due to its simplicity and high performance, to distribute the tasks in a *complex* application to various clusters. The key idea of this application distribution algorithm is the use of energy consumption value to organize and structure the search for best schemes. As long as the minimum energy cost value is found, the optimal allocation policy can be easily obtained. Specifically, the algorithm can be separated into two steps; in the first phase, the energy consumption of current allocation policy is computed, and then the policy is improved by a minimization over the overall energy dissipation. This process continues until converging to the optimal solution. From above process, we can find that the core of the application allocation algorithm is the computation of the energy cost. It can be defined as a function $o : A \times \Omega \rightarrow \mathbb{R}$, where Ω is the set of possible allocation strategies, the value equals the total energy consumption that starts from current task distribution and thereafter following a strategy $\omega \in \Omega$. To compute the function value of a specified application allocation policy, the energy consumption for executing each task in the application, including the computation and communication energy cost, is needed. As a consequence, a scheme that assigns a task to multiple sensor nodes inside a cluster is adopted to compute the exact energy consumption, the scheme is detailed in the next section. In summary, the inter-cluster application allocation algorithm is shown in Algorithm 1.

Algorithm 1. Inter-cluster application allocation algorithm

```
1:  Initialization
2:  compute the resources of each cluster $G_u$, where
        $R_{G_u} = \sum_{S_i \in G_u} C_{S_i}$
3:  compare the required resources of each task $T_j \in A$
        with $R_{G_u}$
4:  if $R_{T_j} \leq R_{G_u}$ then
5:  $\rho(T_j, G_u) = 1$
6:  end if
7:  randomly allocate task $T_j \in A$ to a cluster $G_u$ as long
        as $\rho(T_j, G_u) = 1$
8:  $V(T_j) = 0$ for all $T_j \in A$
9:  set $\epsilon$ as a small positive number
10:  optimal_flag:= 1;
11:  /* Evaluate the policy value */
12:  repeat
13:    $\delta := 0$
14:    for each task $t_j \in A$ do
15:      $v := V(T_j)$
16:      $V(T_j) := E(T_j, G_u) + V(T_j^{succ})$
17:      $\delta := max(\delta, |v - V(T_j)|)$
18:    end for
19:  until $\delta < \epsilon$
20:  /* Improve the policy */
21:  for each task $t_j \in A$ do
22:    $\alpha := \pi(T_j)$
23:    $\pi(T_j) = arg\min_{G_u} (E(T_j, G_u) + V(T_j^{succ}))$
24:    if $\alpha \neq \pi(T_j)$ then
25:      optimal_flag:= 0;
26:    end if
27:   end for
28:  if optimal-flag:= 1 then
29:    stop
30:  else
31:    go to line 12
32:  end if
```

The first component of Algorithm 1 (lines 1–10) is initialization. In this part, the resources of a cluster are calculated, and they are equal to the sum of resources of all sensor nodes in the cluster. To a task T_j, if its required resources are covered by the resources of a cluster G_u, G_u is regarded as a potential cluster to T_j (line 5). Then, an initial policy of application allocation is generated by randomly distributing a task to a cluster that can perform it. Meanwhile, the original value of each task is set to 0. From lines 11–19, the value of each task under current allocation policy is computed iteratively. Note that in line 16, $E(T_j, G_u)$ represents the energy cost of assigning task T_j to cluster G_u. Moreover, the value of task T_j contains not only the energy cost of executing T_j but also the energy consumption for executing succeeding tasks following current application allocation policy thereafter, which is denoted as $V(T_j^{succ})$. Here, the variable $V(T_j)$ is not the value of task T_j, it indicates the value of a task allocation state. From lines 21–27, the current policy is improved if it is not the optimal one. In line 23, the cluster which can provide the minimum energy cost for present task as well as subsequent tasks is preferred. By evaluating and improving current application allocation policy iteratively, the optimal application distribution scheme can be obtained.

The intra-cluster task allocation algorithm

Through inter-cluster application allocation, all tasks in an application have been allocated to multiple clusters with the least energy expenditure. Then, the CH which is associated with a task is responsible for distributing the task to variable sensor nodes in the cluster. Thus, the CH plays an important role in the process of intra-cluster task allocation. However, in this article, we are concerned with task allocation and the selection of the CH is beyond our scope. On the other side, as there is no specific requirement for the CH, any previous method can be adopted to determine the CH in our approach. In our model, a task can be partitioned into several parts, each of which demands for a specific type of resource; the selected nodes should have one type of the required resource and therefore the task can be guaranteed to perform. Our global purpose of allocating a *complex* application is extending network lifetime; it can be achieved from two aspects: energy reduction and load balancing. In previously described stage, the main metric is the energy consumption of different allocation schemes; in this phase, both the energy cost and workload balancing are taken into account.

To reduce the power consumption of distributing a task inside a cluster, it is important to minimize the energy cost for communication between each pair of allocated nodes and the CH. This is due to the fact that in our case, each normal sensor node with the identical type of resource will consume equal energy when undertaking the same amount of computation, that is, from the viewpoint of computation energy consumption, there is no difference among the nodes that can provide the same type of resource. On the other side, the communication energy cost is highly related to the distance between a sensor node and the CH, the larger the distance, the more energy needed. Consequently, to minimize the total energy expenditure of allocating a task inside a cluster, the nodes that are closest to the CH are preferred. However, if these nodes are always chosen out to execute tasks, their power will be overused and they die quickly. In such a case, the system lifetime will be significantly shortened. Therefore, it is hard to find a coherent solution that can simultaneously optimize energy dissipation and workload balancing, although there may be an optimal solution for each one of the individual objectives. Even worse, as mentioned above, to prolong the network lifetime, these two objectives are a pair of contradictory metrics to a certain extent, and a solution that provides good performance for one target may worsen the performance for another one. Thus, some trade-offs between the two metrics must be made to achieve the global optimization. Here, we develop an effective method to handle the issue. The main idea is that to each sensor node which has the demanded type of resource, it is associated with a probability which indicates its possibility to be selected as a member for performing the task. Obviously, the probability is determined by two elements where one is energy related and the other is load related. To achieve a good load balance among sensor nodes, the workload assigned to a node should match the remaining energy of it. In general, the lower residual energy of a node, the smaller probability of it to be selected. By integrating these two factors, we can adopt the following equation to evaluate the probability of a node to contribute its resource q to the task

$$p(S_i) = (1-\mu)\frac{\eta_i}{\sum_{S_i \in G_u \cap S_i \text{ provides resource } q} \eta_i} + \mu \frac{E^i_{resid}}{\sum_{S_i \in G_u \cap S_i \text{ provides resource } q} E^i_{resid}} \tag{6}$$

where $\eta_i = (1/E^i)$ represents the heuristic information from energy cost of node S_i, E^i_{resid} is the residual energy of S_i, and μ is an important parameter that allows users to adjust the preference tolerance between energy cost and load balancing. The smaller value of μ indicates the more importance of energy consumption while less importance of even load distribution. To calculate $p(S_i)$, two parameters, E^i and E^i_{resid}, are needed. E^i is the total energy consumption of sensor node S_i to execute the task, which can be easily calculated by equations (2) and (3) as all variable values are known in advance. Meanwhile, as the initial energy and the power consumption are known to the CH, it can construct a list which records the residual of each sensor node, and the element is updated when a node joins a task. Overall, we propose an algorithm to allocate a task inside a cluster which can control the trade-offs between the objectives. The pseudocode is given in Algorithm 2.

In Algorithm 2, $\mathbb{B}$ indicates the set of sensor nodes which is selected out for performing the task and it is initialized to be empty. G_u^q is the set of nodes that have resource q in current cluster G_u. From lines 6–10, the nodes that have the required resource of the task are classified according to their resource type. Then, the nodes with the maximum probability computed by equation (6) in each category are determined and they eventually form a group to collaboratively execute the task.

Algorithm 2. Intra-cluster task allocation algorithm

```
1:  **Initialization**
2:  assign values to parameters $\mu$
3:  $\mathbb{B} = \emptyset, G_u^q = \emptyset$
4:  $p\_max = 0$
5:  for each $r_{T_k}^q = 1, r_{T_k}^q \in R_{T_k}$ do
6:      for each $S_i \in G_u$ do
7:          if $C_{S_i}^q = 1$ then
8:              $G_u^q = G_u^q \cup S_i$
9:          end if
10:     end for
11:     for each $S_i \in G_u^q$ do
12:         compute $p(S_i)$
13:         if $p(S_i) > p\_max$ then
14:             $p\_max = p(S_i)$
15:             $\theta = i$
16:         end if
17:     end for
18:     $\mathbb{B} = \mathbb{B} \cup S_\theta$
19: end for
20: return $\mathbb{B}$
```

Computational complexity analysis

To the inter-cluster application allocation algorithm, its computational complexity is the same as that of policy iteration algorithm of DP, that is, $O(m^3)$, where m is the number of tasks in a *complex* application. To the intra-cluster task allocation scheme, we assume that $|G_u|$ is the number of sensor nodes in cluster G_u, $|G_u^q|$ is the number of nodes in cluster G_u that has resource q, and $|R_{T_k}|$ is the number of required resource type of task T_k. The iteration from lines 6–10 executes at most $|G_u|$ times, and the iteration from lines 11–17 is executed in $O(|G_u^q|)$ steps. Moreover, the loop at line 5 runs $|R_{T_k}|$ times. Thus, the complexity of the proposed algorithm is $O(m^3(|R_{T_k}|(|G_u| + |G_u^q|)))$. From the above analysis, we can find that the amount of tasks plays a notable impact on the computation load; nevertheless, it is essentially a polynomial-time algorithm.

Experimental results

In this section, extensive experiments are conducted to objectively evaluate the performance of our two-phase *complex* application allocation algorithm in heterogeneous hierarchical WSNs, which is referred to as TP-CAA-HH. Since no similar works (dealing with the issue of *complex* application allocation algorithm in heterogeneous hierarchical WSNs) have been proposed so far, we develop an intuitive method as well as modify an existing method to solve the problem, namely, greedy-based distribution method and binary PSO-based method, and compare our approach with these two methods.

Greedy-based method (G-CAA-HH). To this approach, the *complex* application is first distributed to clusters randomly, that is, for all clusters that own the required resources of a specific task, they have the same probability to be selected out to perform the task. Afterward, a task is allocated to various sensor nodes within a cluster by a greedy heuristics; in more detail, the CH computes the energy dissipation of each sensor node that can match the demanded resources of the task, the ones with the minimum cost will cooperatively execute the task.

Binary particle swarm optimization–based method (BPSO-CAA-H). To compare with the state-of-the-art, we modify the BPSO-based task allocation method proposed in Yang et al.[31] The original approach is capable of allocating tasks in a homogeneous nonhierarchical WSN, we do a few modifications to make this scheme suitable for our situation. In particular, all the sensor nodes are partitioned into several groups according to their resource type, in each group, a sensor node is determined by using BPSO algorithm with the consideration of both the energy consumption and load balancing. Note that in this framework, the WSN has a flat structure and all nodes communicate directly with the sink node.

Experimental setup

In our simulation, the WSN is composed of 400 sensor nodes that are uniformly distributed in a circular area with a radius of 500 m, the sink node is located at the center. We assume there are six distinctive types of resources in the system. To a task, the number of the type of required resources is generated randomly; similarly, a sensor node is endowed with a random resource. To energy consumption model, several popular parameters are adopted[32,33] as follows: $E_{elec} = 50$ nJ/b, $\varepsilon_{fs} = 10$ pJ/b/m^2, the amount of data l is uniformly distributed in the range of [40,000, 60,000] bits. Our simulation platform includes MATLAB R2016a run on an AMD 3.6 GHz CPU and 4GB RAM.

To comprehensively evaluate the performance of TP-CAA-HH, the following metrics are measured:

- *Energy consumption.* It is the energy dissipation for executing the applications. In addition, the energy consumption for performing each task in an application is also recorded for comparison.
- *Stand deviation of residual energy.* It is the stand deviation of residual energy of all nodes, as defined by equation (4). It indicates the load balancing of the system.
- *Number of unallocated task.* It is the number of tasks that cannot be successfully allocated for

the lacking of some specific resources. This is due to the fact that with the running of WSN, some nodes may deplete the power and their resources become unavailable.

- *Number of living nodes*. It is the number of nodes that do not exhaust their energy. In the simulation, we regard this metric as a measurement of network life since in most existing literatures, network lifetime is defined as the proportion of alive nodes to all nodes in the WSN.

Effect of parameters

The first set of experiments is conducted to investigate the effect of parameter μ of the algorithm. In the experiment, the number of applications is 10 and an application consists of 15 tasks. The results are demonstrated in Figure 3. In Figure 3(a), when the value of μ varies from 0.1 to 0.9, the energy consumption of an application increases gradually, this is because that by equation (6), a higher value of μ indicates that the residual energy of a node has a more impact on the probability $p(S_i)$. Specifically, when μ is 0.8 or 0.9, the intra-cluster task allocation algorithm puts more emphasis on the even power distribution among sensor nodes, the nodes which have been used several times will be endowed with a low possibility to be selected again, while the ones with sufficient energy are more likely to join the task even though more power is needed for computation and communication. In contrast, Figure 3(b) demonstrates that with the increasing of μ, the standard deviation of remaining energy of all nodes decreases. It is easy to understand because a high value of μ implies more importance on load balancing; therefore, the energy consumption can be distributed among nodes uniformly.

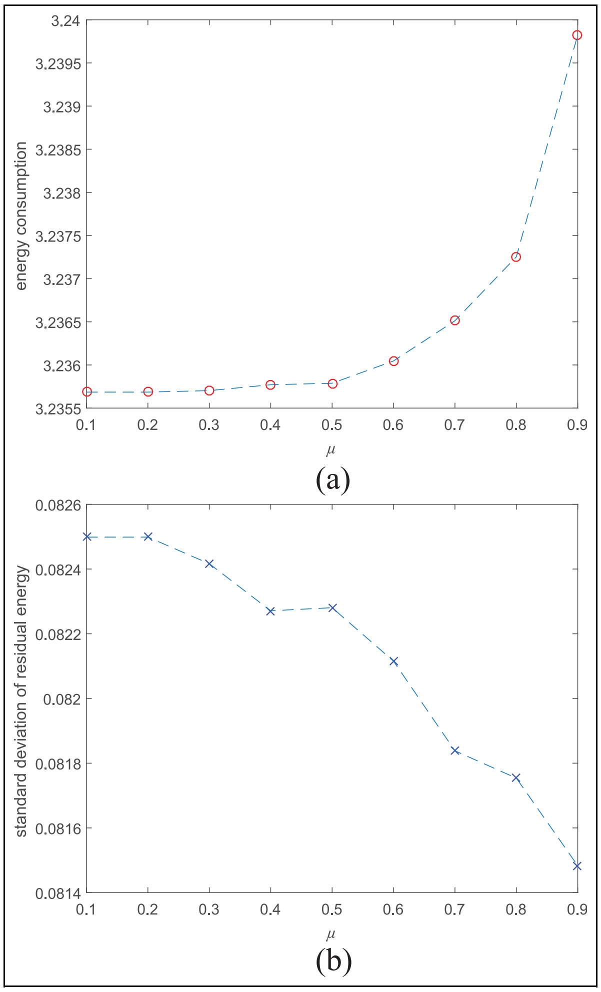

Figure 3. Performance with parameters: (a) energy consumption and (b) standard deviation of residual energy.

Performance comparison

Effect of the number of applications. To further illustrate the efficiency of the proposed algorithm, we compare it with two benchmarks. Here, we focus on the effect of the number of *complex* applications which varies from 1 to 350, the parameter μ is set to 0.5. Figure 4 shows the energy state of different methods. Based on Figure 4(a), we observe that the energy consumption of the three approaches all goes up when the number of applications increases, whereas TP-CAA-HH and G-CAA-HH noticeably outperform BPSO-CAA-H. This is because BPSO-CAA-H distributes all tasks in a non-hierarchical WSN in which all sensor nodes need to communicate with the sink node directly. In contrast, to TP-CAA-HH and G-CAA-HH, the sensor nodes only send information to the CH and the CH forwards it to the sink node, which consumes much less energy. Furthermore, TP-CAA-HH is better than G-CAA-HH due to the fact that TP-CAA-HH allocates applications to clusters with the goal of minimizing energy cost, while G-CAA-HH allocates applications randomly. It illustrates that Algorithm 1 is efficient in allocating tasks to clusters. Figure 4(b) shows the standard deviation of the residual energy of all nodes in WSN, we find that the standard deviation of the residual energy increases with the execution of applications. However, the proposed algorithm is superior to the other two methods which implies that TP-CAA-HH can distribute tasks more evenly to the nodes. An interesting observation is that when the application number is large (greater than about 190), the residual energy standard deviation of BPSO-CAA-H decreases. This is because to BPSO-CAA-H, the remaining power of the

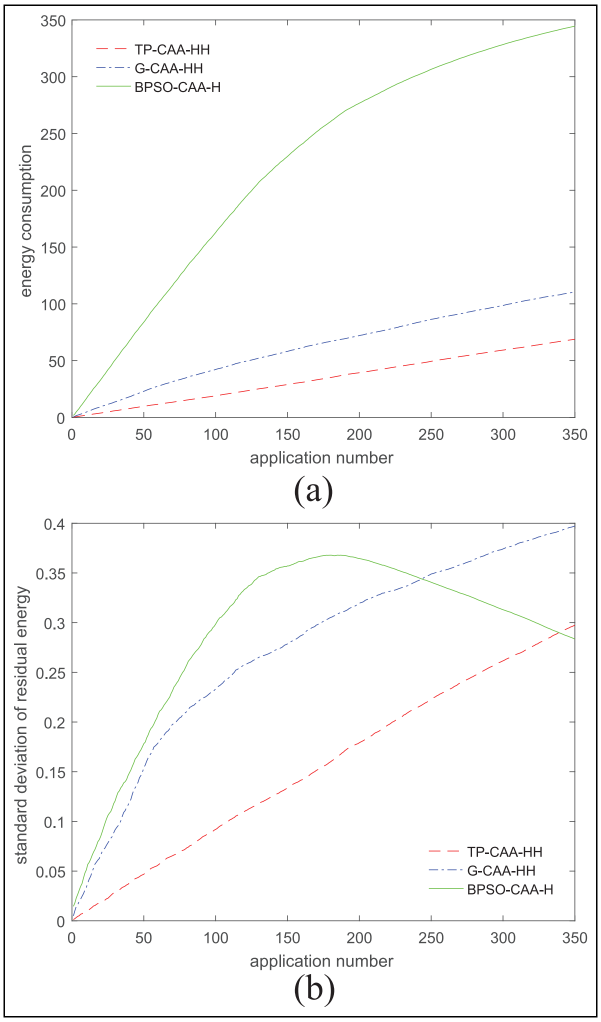

Figure 4. Performance comparison with the number of applications: (a) energy consumption and (b) standard deviation of residual energy.

nodes reduces remarkably when more applications are allocated, and eventually, the residual energy of most sensor nodes drops to 0 or near 0. In such a case, the standard deviation will also decline.

In our configuration, some applications may not be successfully allocated due to the lack of some specific types of resources, as the corresponding sensor nodes deplete their energy and die. Thus, we compare the performance of the three methods in terms of unallocated task number, the results are shown in Figure 5. From Figure 5(a), we observe that with the application distribution, G-CAA-HH is the first approach where the case of unallocated tasks occurs, followed by BPSO-CAA-H, while TP-CAA-HH is capable of allocating all the applications with no failure. This can be explained by the fact that to G-CAA-HH, it assigns tasks with greedy heuristics which will lead to the overuse of some nodes. If these nodes exhaust their power prematurely,

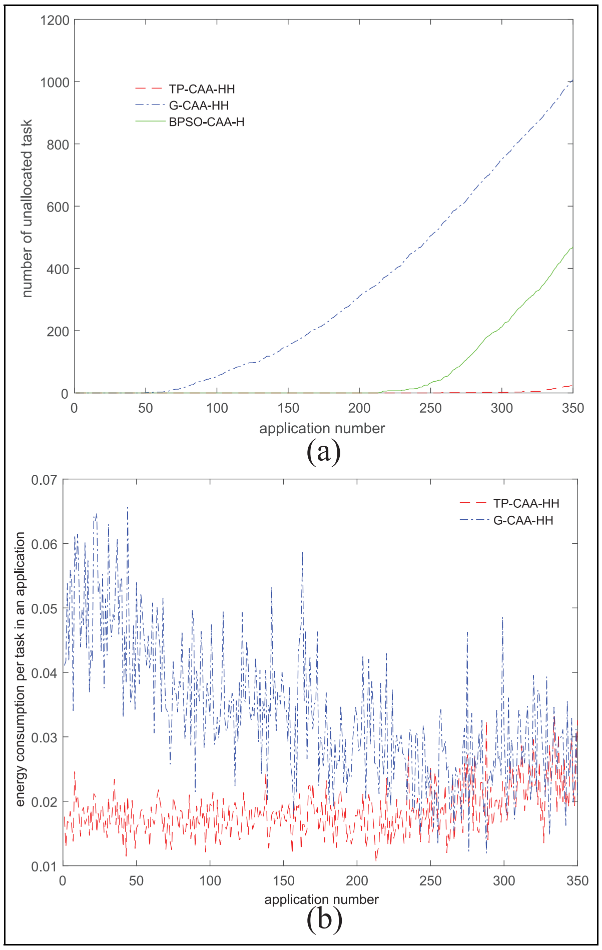

Figure 5. Performance comparison with the number of applications: (a) number of unallocated tasks and (b) energy cost of each task.

the tasks that demand the corresponding resources can no longer be executed. In contrast, TP-CAA-HH considers both energy cost and workload balancing which significantly defer the death of the nodes; thus, all the tasks are allocated successfully. It should be noted that BPSO-CAA-H can select the desired sensor nodes from the whole WSN that makes it predominant on the possibility of task assignment. However, the presented approach is better which provides a further verification of the superior performance of TP-CAA-HH. Figure 5(b) shows the average energy dissipation of a task in an application, the result seems a little strange. To G-CAA-HH, it allocates task to nodes with the least energy cost and therefore should have better performance. Nevertheless, the simulation exhibits opposite result. We attribute this to the fact that in TP-CAA-HH, it distributes all tasks in an application to various clusters by Algorithm 1, which aims at minimizing energy consumption, whereas to G-CAA-HH, it allocates an application randomly which consumes more

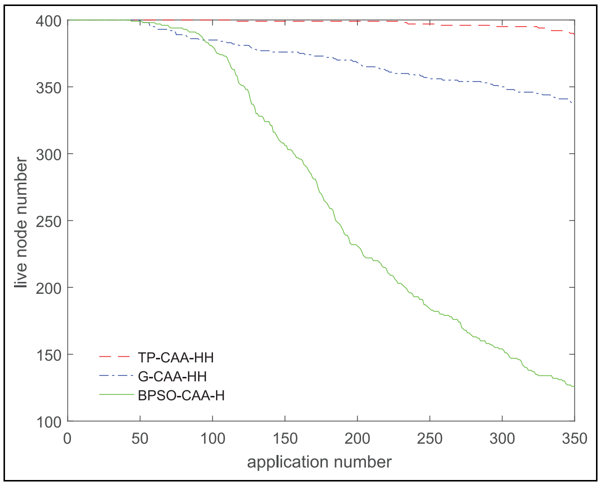

Figure 6. Number of living sensor nodes.

power. Consequently. the proposed inter-cluster allocation algorithm is certified to be efficient again.

The last experiment is conducted to investigate the performance with respect to the number of living nodes which provides an indirect illustration of the network lifetime. The result is shown in Figure 6. As expected, with the increase of application number, the amount of living nodes of BPSO-CAA-H descends drastically while only very few nodes die of TP-CAA-HH, G-CAA-HH achieves a medium performance. This can easily be explained by the fact that TP-CAA-HH adopts the technology of both reducing energy consumption and balancing workload which is significantly beneficial to prolong the lifetime of individual sensor nodes. In contrast, to BPSO-CAA-H, nodes transmit information directly to the sink node that expends the power of sensor nodes seriously. Therefore, a great deal of nodes die within a short period. To G-CAA-HH, although sensor nodes communicate with the CHs, it neglects load balance and the overused nodes deplete their energy quickly.

Effect of cluster size. In this set of experiments, we investigate the performance of various algorithms with changing the cluster size. The cluster size is the number of nodes in a cluster; here, it is varied from 16 to 30 with a step 2. In the simulations, the number of applications is 200; each application contains five tasks, and the radius of the area is set to 2000 m. From Figure 7(a), we observe that the energy dissipation of the WSN increases when the network size gets larger. The explanation is that, when the quantity of sensor nodes increases in a cluster, more tasks can be successfully allocated which will inevitably lead to more energy consumption. TP-CAA-HH and G-CAA-HH dissipate much less power than BPSO-CAA-H; this situation

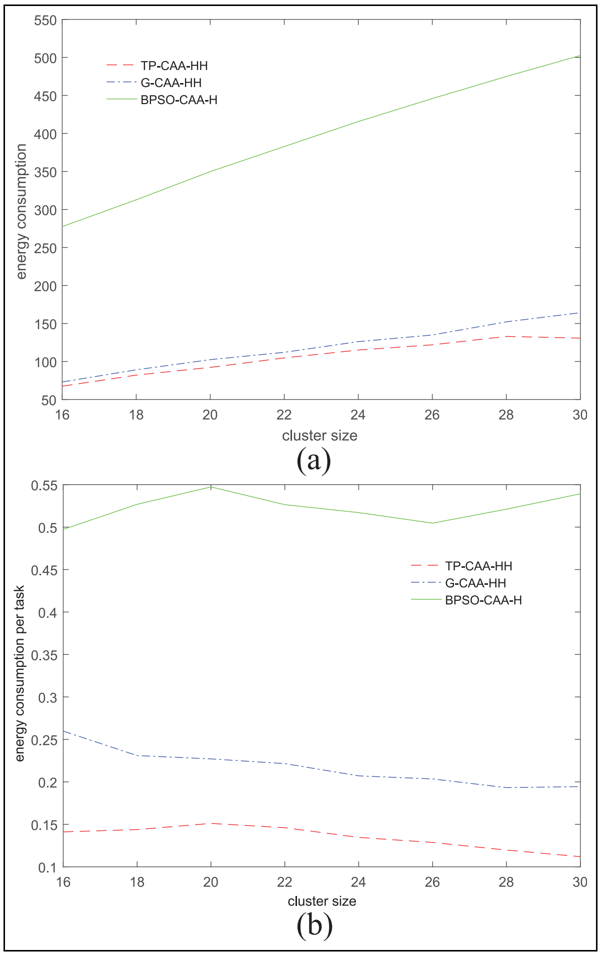

Figure 7. Performance comparison with cluster size: (a) energy consumption and (b) energy cost of each task.

comes from two aspects. On one side, to a single task allocation, BPSO-CAA-H consumes much more energy than TP-CAA-HH and G-CAA-HH because it searches appropriate nodes in the whole WSN, while TP-CAA-HH and G-CAA-HH assign a task within a cluster. On the other side, when the sensing resources of a WSN are limited, that is, only few tasks can be completed, BPSO-CAA-H can allocate more tasks than the other two methods and the energy consumption will improve accordingly. Figure 7(b), CAA-HH, shows the best performance even compared with G-CAA-HH, which allocates tasks with a energy-greedy strategy within a cluster. In summary, if the number of tasks is large and the WSN is resource constrained, BPSO-CAA-H can be adopted to allocate tasks as much as possible, while if the network size is large, TP-CAA-HH is preferred to achieve good performance in task allocation as well as energy conservation.

Figure 8 shows the performance of the three algorithms with regard to the number of unallocated task and the number of living nodes. As illustrated in

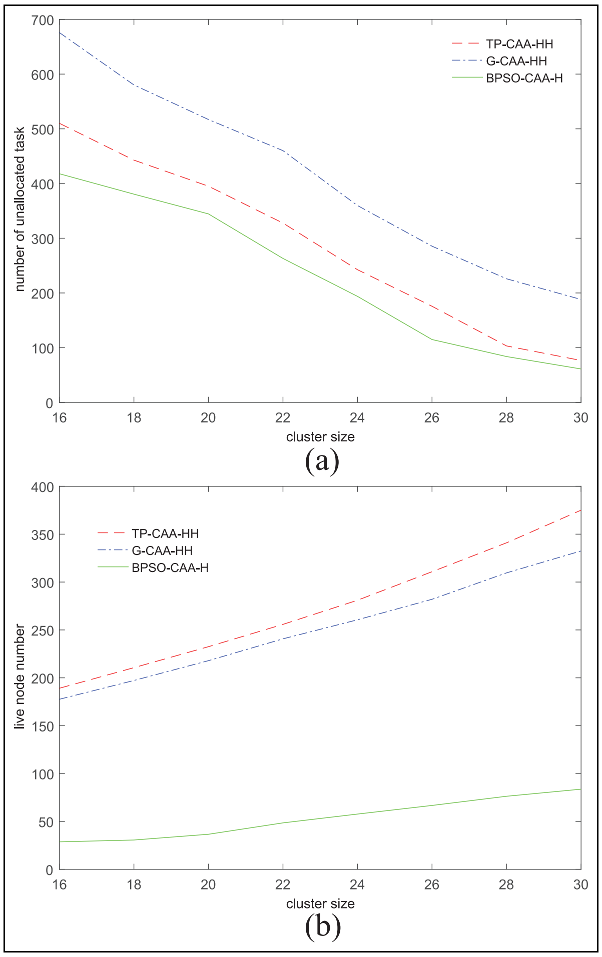

Figure 8. Performance comparison with cluster size: (a) number of unallocated tasks and (b) number of living nodes.

Figure 8(a), the number of unallocated tasks decreases as there is an increase in cluster size. This is due to the fact that more sensor nodes in a cluster indicates a higher possibility for distributing a task; hence, less tasks will not be allocated. BPSO-CAA-H achieves the best performance because all the nodes in WSN can be used for executing tasks, whereas to TP-CAA-HH and G-CAA-HH, only the nodes in the cluster can be allocated with a task. However, it should be noted that with the increase in the cluster size, the gap between BPSO-CAA-H and TP-CAA-HH is narrowed down which implies that when the cluster size is large, TP-CAA-HH can allocate almost the same number of tasks as BPSO-CAA-H while costs much less energy. Figure 8(b) shows that when the cluster size becomes larger, the number of living nodes increases proportionally, yet TP-CAA-HH outperforms the other two methods. This can be ascribed to the fact that TP-CAA-HH makes a good tradeoff between energy consumption and load balance, which can prolong the lifetime of nodes significantly.

Conclusion

In this work, we discussed the issue of *complex* application distribution in a heterogeneous clustered WSN. Unlike most previous studies that assume all sensor nodes in WSN are homogeneous, in our system all nodes are associated with distinctive kinds of resources each of which can be utilized to perform a specific action. Accordingly, a *complex* application is composed of several tasks and each task demands various types of resources. Our objective is to distribute the *complex* application to appropriate sensor nodes with the purpose of prolonging the system lifetime while ensuring that the required resources of each tasks can be covered by the resources of the nodes. To achieve this goal, a two-phase-based application allocation algorithm is proposed. First, a DP-based scheme is employed to distribute all tasks in the *complex* application to multiple clusters which aims at minimizing the energy dissipation. Then, inside a cluster, the task is further assigned to sensor nodes by a feedback mechanism which prevents the overuse of individual nodes. In particular, a parameter μ is provided to allow end-users to tune the importance between energy conservation and workload balancing. Experimental results show that compared with approaches based on Greedy and BPSO, the proposed algorithm achieves superior performance in terms of energy consumption as well as load balancing.

In this article, an application can be allocated with no time constraints. However, in some scenarios, an application is required to be distributed and performed within limited time. In the future, we will extend our approach to take application deadline into account which can provide real-time guarantees. Meanwhile, in recent years, the technology of mobile sink has emerged as a promising method to further reduce energy consumption of WSNs;[34,35] therefore, another interesting future direction is to address the problem in a WSN with mobile sink.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work is supported in part by the National

Natural Science Foundation of China under Grant No. 61472344, 61772454, 6171101570, the Natural Science Foundation of Jiangsu Province under Grant No. BK20150460.

ORCID iD

Jin Wang https://orcid.org/0000-0001-5473-8738

References

1. Akyildiz IF, Su W, Sankarasubramaniam Y, et al. Wireless sensor networks: a survey. *Comput Netw* 2002; 38(4): 393–422.
2. Vercauteren T, Guo D and Wang X. Joint multiple target tracking and classification in collaborative sensor networks. *IEEE J Sel Area Comm* 2005; 23(4): 714–723.
3. Shen J, Tan H, Moh S, et al. An efficient RFID authentication protocol providing strong privacy and security. *J Internet Technol* 2016; 17(3): 443–455.
4. Reina DG, Askalani M, Toral SL, et al. A survey on multihop ad hoc networks for disaster response scenarios. *Int J Distrib Sens N* 2015; 2015(3): 1–16.
5. Wang J, Abid H, Lee S, et al. A secured health care application architecture for cyber-physical systems. *Control Eng Appl Inf* 2011; 13(3): 101–108.
6. Valera M and Velastin SA. Intelligent distributed surveillance systems: a review. *IEE P: Vis Image Sign* 2005; 152(2): 192–204.
7. Wen Y, Xue H and Yang JD. A heuristic-based hybrid genetic-variable neighborhood search algorithm for task scheduling in heterogeneous multiprocessor system. *Inform Sci* 2011; 181(3): 567–581.
8. Kong Y, Zhang M and Ye D. A belief propagation-based method for task allocation in open and dynamic cloud environments. *Knowl-Based Syst* 2017; 115: 123–132.
9. Jiang Y, Zhou Y and Wang W. Task allocation for undependable multiagent systems in social networks. *IEEE T Parall Distr* 2013; 24(8): 1671–1681.
10. Pilloni V and Atzori L. Energy-efficient task allocation for distributed applications in Wireless Sensor Networks. In: *Proceedings of the GLOBECOM workshops*, Houston, TX, 5–9 December 2011, pp.321–326. New York: IEEE.
11. Wang J, Ma T, Cho J, et al. An energy efficient and load balancing routing algorithm for wireless sensor networks. *Comput Sci Inf Syst* 2011; 8(4): 991–1007.
12. Giannecchini S, Caccamo M and Shih C-S. Collaborative resource allocation in wireless sensor networks. In: *Proceedings of the 16th Euromicro conference on real-time systems (ECRTS 04)*, Catania, 2 July 2004, pp.35–44. New York: IEEE.
13. Xiao W, Low S, Tham C, et al. Prediction based energy-efficient task allocation for delay-constrained wireless sensor networks. In: *Proceedings of the 6th IEEE annual communications society conference on sensor, mesh and ad hoc communications and networks*, Rome, 22–26 June 2009, pp.1–3. New York: IEEE.
14. Li W, Delicato FC, Pires PF, et al. Efficient allocation of resources in multiple heterogeneous wireless sensor networks. *J Parallel Distr Com* 2014; 74(1): 1775–1788.
15. Heinzelman WB, Chandrakasan AP and Balakrishnan H. An application-specific protocol architecture for wireless microsensor networks. *IEEE T Wirel Commun* 2002; 1(4): 660–670.
16. Liao Y, Qi H and Li W. Load-balanced clustering algorithm with distributed self-organization for wireless sensor networks. *IEEE Sens J* 2013; 13(5): 1498–1506.
17. Wang J, Cao Y, Li B, et al. Particle swarm optimization based clustering algorithm with mobile sink for WSNs. *Future Gener Comp Sy* 2017; 76: 452–457.
18. Yu Y and Prasanna VK. Energy-balanced task allocation for collaborative processing in wireless sensor networks. *Mobile Netw Appl* 2005; 10(1–2): 115–131.
19. Abdelhak S, Gurram CS, Ghosh S, et al. Energy-balancing task allocation on wireless sensor networks for extending the lifetime. In: *Proceedings of the 53rd IEEE international midwest symposium on circuits and systems, Seattle, WA*, 1–4 August 2010, pp.781–784. New York: IEEE.
20. Shen CC, Plishker WL, Ko D, et al. Energy-driven distribution of signal processing applications across wireless sensor networks. *ACM T Sensor Network* 2010; 6(3): 1–32.
21. Yu W, Huang Y and Garcia-Ortiz A. Distributed optimal on-line task allocation algorithm for wireless sensor networks. *IEEE Sens J* 2018; 18(1): 446–458.
22. Li W, Delicato FC and Zomaya AY. Adaptive energy-efficient scheduling for hierarchical wireless sensor networks. *ACM T Sensor Network* 2013; 9(3): 1–34.
23. Jin Y, Jin J and Gluhak A. An intelligent task allocation scheme for Multihop Wireless Networks. *IEEE T Parall Distr* 2012; 23(3): 444–451.
24. Guo W, Xiong N, Chao HC, et al. Design and analysis of self-adapted task scheduling strategies in wireless sensor networks. *Sensors* 2011; 11: 6533–6554.
25. Guo W, Li J, Chen G, et al. A PSO-optimized real-time fault-tolerant task allocation algorithm in wireless sensor networks. *IEEE T Parall Distr* 2015; 26(12): 3236–3249.
26. Luo T, Tan HP and Quek T. Sensor OpenFlow: enabling software-defined wireless sensor networks. *IEEE Commun Lett* 2012; 6(11): 1896–1899.
27. Zeng D, Li P, Guo S, et al. Energy minimization in multi-task software-defined sensor networks. *IEEE T Comput* 2015; 64(11): 3128–3139.
28. Dietrich I and Dressler F. On the lifetime of wireless sensor networks. *ACM T Sensor Network* 2009; 5(1): 1–39.
29. Howard RA. *Dynamic programming and Markov processes*. Cambridge, MA: The MIT Press, 1960.
30. McMahan HB, Likhachev M and Gordon GJ. Bounded real-time dynamic programming: RTDP with monotone upper bounds and performance guarantees. In: *Proceedings of the 22nd international conference on machine learning*, Bonn, 7–11 August 2005, pp.569–576. New York: ACM.
31. Yang J, Zhang H, Ling Y, et al. Task allocation for wireless sensor network using modified binary particle swarm optimization. *IEEE Sens J* 2014; 14(3): 882–892.
32. Wang A and Chandrakasan A. Energy-efficient DSPs for wireless sensor networks. *IEEE Signal Proc Mag* 2002; 19(4): 68–78.

33. Shih E, Cho S, Ickes N, et al. Physical layer driven protocol and algorithm design for energy-efficient wireless sensor networks. In: *Proceedings of 7th annual international conference on mobile computing and network*, Rome, 16–21 July 2001, pp.272–286. New York: ACM.

34. Wang J, Cao Y, Ji S, et al. Energy-efficient cluster-based dynamic routes adjustment approach for wireless sensor networks with mobile sinks. *J Supercomput* 2017; 73(7): 3277–3290.

35. Wang J, Li B, Xia F, et al. An energy efficient distance-aware routing algorithm with multiple mobile sinks for wireless sensor networks. *Sensors* 2014; 14: 15163–15181.