Ad Hoc Networks 26 (2015) 103–113

Contents lists available at ScienceDirect

Ad Hoc Networks

journal homepage: www.elsevier.com/locate/adhoc

Maximizing precision for energy-efficient data aggregation in wireless sensor networks with lossy links

Shiliang Xiao [a,b,*], Baoqing Li [a], Xiaobing Yuan [a]

[a] Key Laboratory of Wireless Sensor Network & Communication, Shanghai Institute of Microsystem and Information Technology, Chinese Academy of Sciences, Shanghai, China
[b] University of Chinese Academy of Sciences, Beijing, China

ARTICLE INFO

Article history:
Received 23 May 2014
Received in revised form 7 October 2014
Accepted 15 November 2014
Available online 21 November 2014

Keywords:
Wireless sensor networks
Data aggregation
Data quality
Link unreliability
Algorithms

ABSTRACT

Two main factors that impact the performance of data aggregation in wireless sensor networks (WSNs) are data quality and energy efficiency. This paper exploits the tradeoff between data quality and energy consumption to maximize the data aggregation precision under heterogeneous per-node energy constraints. Unlike previous work, we explicitly account for link loss in the optimization framework. To tackle link unreliability, we need to appropriately allocate the limited energy across the incoming and outgoing links of each individual node. We present a centralized algorithm based on the Immune-Genetic heuristic to find near-optimal energy allocation strategy such that the precision of the aggregated data received by the sink is maximized. The algorithmic complexity and implementation issues are also discussed. Furthermore, we develop a localized alternative algorithm based on the Gibbs sampler, which is more scalable and can adapt to large-scale distributed WSNs. Finally, we conduct numerical simulations to demonstrate the convergence as well as the data aggregation precision performance of the proposed algorithms.

1. Introduction

Wireless sensor networks (WSNs) have long been recognized as a new information-gathering approach for many applications, such as environment monitoring, security surveillance, and industry control [1]. Besides sensing interested information, the primary task in WSNs is how to gather data from a set of scattered sensor nodes. A typical way of data gathering is to forward the sensed data to a common sink via multi-hop wireless communications, which can be termed as *many-to-one communication* [2]. Furthermore, when the summarized information is required or the sensed data are correlated, it is beneficial to aggregate data en route to the sink. In specific, intermediate nodes in the network are allowed to combine the received packets and its own packet into a single one by using some aggregation functions such as MAX, MIN, and SUM [2]. This helps in reducing redundancy as well as the number of transmissions. Such a many-to-one communication pattern under aggregation is referred to as *in-network data aggregation*, which has been well studied from various aspects in the literature, e.g. [3–6].

One performance aspect of data aggregation is data quality, which is usually measured by the precision (e.g., the quantitative error variance) of the data encapsulated in the aggregated packet [3]. Besides, energy efficiency or network lifetime is also an important performance aspect since sensor nodes are typically powered by batteries, which have finite capacities and are generally difficult to

* Corresponding author at: Key Laboratory of Wireless Sensor Network & Communication, Shanghai Institute of Microsystem and Information Technology, Chinese Academy of Sciences, Shanghai, China. Tel.: +86 13817179794; fax: +86 021 52419932.
E-mail addresses: shliangxiao@gmail.com (S. Xiao), libq@mail.sim.ac.cn (B. Li), yuanxb@mail.sim.ac.cn (X. Yuan).

http://dx.doi.org/10.1016/j.adhoc.2014.11.014
1570-8705/

replenish. However, it has been reported that these two aspects, i.e. data aggregation quality and energy consumption are mutually conflicting and cannot be optimized at the same time [4,7–10]. In some stringent applications of WSNs such as ecosystem experiments, the users are desired to receive as much detailed data from the network as possible such that they can try various physical models and test various hypothesis over the data [7]. In this case, data precision becomes the dominant factor given that energy consumption remains at a tolerable level.

In the literature, the problem of maximizing data aggregation precision under energy or lifetime constraints has been identified and investigated in [7,9]. Specifically, Qu et al. [7] focused on improving data precision by minimizing the total error bound under the constraint that the predefined network lifetime is achieved for data aggregation and they proposed optimal solutions for both single-hop and multi-hop WSNs. Based on Bernoulli sampling, Cheng et al. [9] developed an energy-efficient data aggregation algorithm that can satisfy arbitrary precision requirement. However, we recognize a common underlying assumption adopted by both [7,9], in which wireless links are always reliable such that the communication cost of each node is deterministic. Actually, due to the *transitional region phenomenon* [11], a large number of wireless links become *lossy* (or unreliable) links and data transmission over a lossy link is successfully conducted with a certain probability [11,12].

When the link unreliability is accounted for, extra challenges will arise, making the precision-energy tradeoff problem more involved. Specifically, if links are unreliable, then the precision of data aggregation will be affected not only by the precision of each individual node but also by the success probabilities of the links along the path of each node towards the sink. In order to improve link success probability, we need additional energy for the transmission over each link to overcome possible transmission failure. Given that the available energy at each node is limited and the link unreliability could be heterogeneous, we need to allocate this limited energy across the incoming and outgoing links of each individual node in an appropriate manner. Specifically, we shall solve an *energy allocation* problem with the objective that the precision of the aggregated data received by the sink is maximized, while meeting the energy constraints of all nodes in the network.

In this paper, we are interested in *maximizing data aggregation precision* (MDAP) for WSNs with lossy links and per-node energy constraints. The MDAP problem is shown to be of combinational complexity, and the exact solution to MDAP is difficult to obtain given that the number of computational steps grows exponentially with linearly increasing problem size. As such, we in this paper prefer to explore efficient algorithms that can find acceptable approximations to the optimal solution with much less time and resources. Specifically, we propose a *Centralized Energy Allocation* (CEA) algorithm based on the Immune-Genetic heuristic [13] to find near-optimal energy allocation strategy such that the precision of the aggregated data received by the sink is maximized. We further propose a *Distributed Energy Allocation* (DEA) algorithm based on the Gibbs sampler [14], which is more scalable and can adapt to large-scale WSNs. The algorithmic complexity and some implementation issues related to CEA and DEA are discussed. Finally, we conduct numerical experiments and the results demonstrate the convergence as well as the data aggregation precision performance of the proposed algorithms.

The remainder of the paper is organized as follows. Section 2 presents the related work. Section 3 introduces the network model and problem definitions. Section 4 presents the details and theoretical analysis of the centralized algorithm. Section 5 presents the details of the distributed algorithm. Section 6 provides the simulation results. Finally, Section 7 concludes the paper and gives some directions for future research.

2. Related work

As a fundamental communication pattern, data aggregation plays a key role in the research of wireless sensor networks (WSNs). Various performance aspects related to data aggregation have been richly studied in the literature. For instance, transmission scheduling aiming at minimizing data aggregation delay was studied in [5,15,16], efficient construction of data aggregation routing was investigated in [6,17], energy-efficient data aggregation algorithms were developed in [18–20], etc.

Besides energy efficiency and delay, data quality or data precision is also an important performance aspect of data aggregation. The tradeoff between data aggregation quality and energy efficiency was initially identified by Pham et al. [3]. Later, Zhu et al. [8] investigated energy-efficient Quality of Service (QoS)-constrained data aggregation for WSNs by determining when and where to perform aggregation in a distributed fashion based on the precision requirement of the particular task. Further, Tang and Xu [4] proposed to differentiate the precisions of the data collected from different sensor nodes to balance their energy consumption, thus maximizing the network lifetime. There were also several related papers aiming at maximizing data quality with given energy or lifetime constraint for data aggregation [7,9]. Specifically [7] focused on minimizing the total error bound with network lifetime requirement, while [9] tried to satisfy arbitrary precision requirement of various data aggregation applications in an energy-efficient manner.

As we have pointed out, the current research related to data aggregation quality is based on an underlying assumption in which links are always reliable. Actually, it has been shown that in practical sceneries, a large number of wireless links become *lossy* links, and over a lossy link, data transmission is conducted with a certain probability [11]. In the research of WSNs, the *probability network model* was recently proposed to account for link unreliability [21]. Aware of this new network model, various traditional research topics in WSNs are gaining interest from the community once again, including the clustering problem [12], the routing problem [22,23], the topology control problem [24], etc.

The methodologies adopted by this paper are based on the Immune-Genetic heuristic and the Gibbs sampler, both

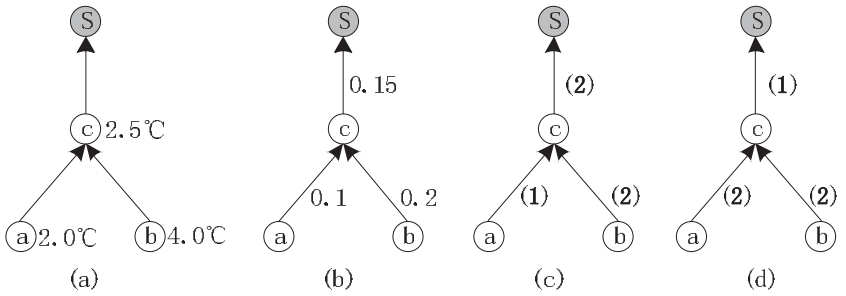


Fig. 1. A motivation example. (a) A four-node network, where *a*, *b* and *c* are sensor nodes responsible of measuring environmental temperature and *S* is the sink node. The measurement error variances of nodes *a*, *b* and *c* are respectively 2.0 °C, 4.0 °C, and 2.5 °C. (b) An unreliable network, where the decimal besides each link represents the corresponding link error probability. (c) The energy consumed for one transmission/reception are both 1 unit. With an energy budget of 5 units, node *c* allocates 1 unit of energy for receiving packets from node *a*, 2 units of energy for receiving packets from node *b*, and 2 units of energy for transmitting packets to *S*. (d) Unlike (c), node *c* allocates 2 units of energy for receiving packets from node *a*, 2 units of energy for receiving packets from node *b*, and 1 unit of energy for transmitting packets to *S*.

of which essentially belong to the *computational intelligence* (CI) discipline [25]. CI provides adaptive mechanisms that exhibit intelligent behaviour in complex and dynamic environments such as WSNs. A comprehensive survey that summarizes the recent studies inspired by CI on various challenges in WSNs such as data collection and aggregation, routing, scheduling, and localization, can be found in [25]. It needs to be emphasized that the Gibbs sampler, being a powerful distributed optimization methodology suitable for distributed settings, is gaining ever increasing research interest in wireless networking community, e.g., [14,26,27].

3. Network model and problem definition

3.1. Network model

We consider tree-based WSNs that can be modelled as $\mathbb{T}(\mathbb{V}\bigcup\{S\},\mathbb{E})$, where $\mathbb{V}$ is the set of sensor nodes, S is the root (sink), and $\mathbb{E}$ is the set of communication links. For node $v\in\mathbb{V}$, we denote its parent by $P(v)$ and its children by the set $\mathbb{C}(v)$. Note that if node v is a leaf, then $\mathbb{C}(v)$ will be empty. To model channel unreliability, each link $(v,P(v))\in\mathbb{E}$ is associated with a *link error probability* ϱ_v, which can be obtained by periodic Hello messages, or be predicted using Link Quality Index (LQI) [11]. Errors are assumed to be independent across links.

To deal with link unreliability, we can adopt various error-recovery techniques such as retransmission, and coding. A common intuition behind these techniques is using extra energy for the transmission over a link to enhance its success probability. As such, we define a general *energy-reliability function* $f_v(\cdot)$ for each link $(v,P(v))$, where $f_v(t_v)\in[0,1]$ indicates the success probability of link $(v,P(v))$ if $t_v\in\mathbb{N}$ units of energy are expended on the packet transmission over this link. For example, if we use retransmission as the error-recovery scheme, then the relationship between the link success probability $f_v(t_v)$ and the link error probability ϱ_v can be explained as follows. As retransmission is used, a node retransmits in case that a packet gets lost. Let one unit of energy be the consumption of one transmission, then t_v retransmissions can be made over link $(v,P(v))$ if t_v units of energy are consumed. In this case, $f_v(t_v)$, being the probability of a success packet transmission over link $(v,P(v))$, becomes $1-\varrho_v^{t_v}$. Generally, we do not make any specific restrictions on $f_v(t_v)$, thus enable it to model any practical recovery schemes in real applications.

Like [7], data quality or precision is measured by the inverse of the error variance of a sample. Let ϕ_v be a weight associated with node v representing the precision of its sample. If data from two nodes u and v are combined, then the overall precision of the combined data can be given by $\phi_u+\phi_v$, which is larger than both ϕ_u and ϕ_v. In light of this observation, we can define the *data aggregation precision* as the sum of the precision provided by the nodes in $\mathbb{V}$ that have participated in the data aggregation operation [7]. We take the four-node network in Fig. 1(a) as an example to explain the implication of data aggregation precision for easier comprehension. In Fig. 1(a), nodes a,b and c are sensor nodes responsible of measuring the current temperature in the environment, and S is the sink that receives an aggregated packet from the sensor nodes. The error variance of the measurement is placed besides each node. By data aggregation, the error variance of the packet received by S is given by $1/(\frac{1}{2.0}+\frac{1}{4.0}+\frac{1}{2.5})=0.87$. If we define the precision of a measurement as the inverse of the corresponding error variance, then the data aggregation precision can be calculated as $\frac{1}{2.0}+\frac{1}{4.0}+\frac{1}{2.5}=1.15$. By aggregating all the packets into a single one, we can see that the overall error variance is decreased, and accordingly, the final precision is improved.

Now, we consider general cases where links are unreliable, and hence, the data aggregation precision experienced by the sink could vary. In this regard, the data aggregation precision is given by

$$W_S=\sum_{v\in\mathbb{V}}\left(\phi_v\cdot\prod_{u\in PATH_v}f_u(t_u)\right). \tag{1}$$

Here $PATH_v$ is the node set consisting of v and all its ancestors in its path towards S, but not including S, i.e. $PATH_v=\{v,P(v),P(P(v)),\ldots\}$, and $S\notin PATH_v$. We again consider the example network in Fig. 1(a), of which the links are assumed to be unreliable and the corresponding

link error probabilities are shown in Fig. 1(b). Suppose that retransmission is used to handle link unreliability, and at most one transmission can be made over each link. In this case, the precision of data aggregation can be calculated as $\frac{1}{2.0}\times(1-0.1)\times(1-0.15)+\frac{1}{4.0}\times(1-0.2)\times(1-0.15)+\frac{1}{2.5}\times(1-0.15)\cong 0.89$. We can see that both the precision of the individual nodes and the success probabilities of the links in the network have an impact on the data aggregation precision W_S.

3.2. Problem definition

Our objective in this paper is to maximize the data aggregation precision W_S. As WSNs are constrained by energy, we impose a heterogeneous energy bound e_v for each node $v\in\mathbb{V}$ to ensure energy-efficient operations. Specifically, the maximum energy consumption of node v during one round of data aggregation should not exceed this bound. In data aggregation, node v spends energy in receiving packets from its children in $\mathbb{C}(v)$, aggregating multiple packets into a single one, and transmitting the aggregated packet to its parent $P(v)$. We assume that the energy consumption of data processing is neglectable compared with that of data transmission. Thus, the energy consumption of node v, Ω_v, is given by

$$\Omega_v=\alpha_v^{TX}\cdot t_v+\alpha_v^{RX}\cdot\sum_{u\in\mathbb{C}(v)}t_u. \tag{2}$$

Here α_v^{TX} and α_v^{RX} are respectively the energy consumption of transmitting and receiving one data packet by node v, and t_v is a variable associated with each node v representing the amount of energy allocated for the transmission over link $(v,P(v))$. Given that the available energy at each node is bounded and the link success probabilities are heterogeneous, we need to allocate the limited energy e_v to the incoming and outgoing links of node v ($v\in\mathbb{V}$), with the ultimate goal that the data aggregation precision W_S is maximized. We again use the example network in Fig. 1 to show that different energy allocation schemes can lead to different values of data aggregation precision. Specifically, two kinds of energy allocation strategies for the unreliable network in Fig. 1(b) are shown in Fig. 1(c) and (d) respectively, where the number in the bracket besides each link indicates the amount of energy allocated for the transmission over this link. Let one unit of energy be the consumption of one transmission. Then, according to (1), the data aggregation precision in Fig. 1(c) can be calculated as $\frac{1}{2.0}\times(1-0.1)\times(1-0.15)^2+\frac{1}{4.0}\times(1-0.2)^2\times(1-0.15)^2+\frac{1}{2.5}\times(1-0.15)^2\cong 0.73$. In a similar manner, the precision in Fig. 1(d) can be obtained as 0.82. Apparently, we can see that the energy allocation scheme has explicit influence on the precision of data aggregation.

We are now able to formulate the *Maximizing Data Aggregation Precision* (MDAP) problem for WSNs with lossy links and per-node energy constraints as follows.

$$\max\ W_S \tag{3}$$

subject to

$$\Omega_v\leqslant e_v,\quad \forall v\in\mathbb{V}, \tag{4}$$

and

$$t_v\in\{0,1,\ldots,K\},\quad \forall v\in\mathbb{V}. \tag{5}$$

Constraint (4) guarantees that the maximum energy consumption of each node in $\mathbb{V}$ is bounded. Constraint (5) limits the range of each decision variable t_v, where K is a user-specified upper bound for the energy allocation over any link. Such a bound is necessary in case that too much energy is spent for an extremely lossy link, which will cause unacceptable energy inefficiency and delay. We do not consider the energy constraint of the sink S, which is more powerful than sensor nodes, and in general does not lack of energy [19]. MDAP turns out to be a non-concave, non-linear, and constrained integer programming (IP) problem, which is of combinational complexity. The exact solution to MDAP is difficult to obtain given that the number of computational steps grows exponentially with linearly increasing problem size. Next, we will develop efficient centralized and distributed algorithms that give good approximations to the optimal energy allocation scheme with much less complexity and overhead.

4. Centralized energy allocation algorithm

In this section, we introduce our Centralized Energy Allocation (CEA) algorithm to solve the MDAT problem. To elaborate, we first briefly describe the Immune-Genetic meta-heuristic on which CEA is based, then present the details of the CEA algorithm, and finally, give some analysis about the complexity and some implementation issues of CEA.

4.1. Introduction to Immune-Genetic meta-heuristic

The basic idea of CEA comes from *genetic algorithm* (GA), which provides a smart meta-heuristic for solving combinational optimization problems [25]. Through simulating the adaptive process of biological evolution and following the natural selection principle known as *survival-of-the-fittest*, GA eventually results in a satisfactory solution after many generations. However, it has been found that pure GA is likely to converge to local optimum, known as the *premature convergence phenomenon*, when the population diversity is violated by some dominant individuals (solutions) [25]. On the other hand, the *artificial immune algorithm* (AIA) has demonstrated its superiority in achieving global optimum by always maintaining the population diversity during the search process [28]. The hybridization of AIA and GA results in the Immune-Genetic heuristic, which is shown to have faster convergence performance as well as better quality of solutions compared with either pure GA or pure AIA [13]. In this paper, the principles behind the Immune-Genetic heuristic will be adopted to develop the CEA algorithm for efficiently solving the MDAP problem.

4.2. Description of CEA

We first present the pseudo-code of CEA as shown in Algorithm 1, and then elaborate the main steps of CEA in details.

Algorithm 1. Centralized Energy Allocation (CEA) Algorithm

Population initialization to generate N_p individuals;
$i = 1$;
while $i <= N_{iter}$ **do**
  Individual evaluation using the fitness, unfitness, and density functions;
  Memory library update using an elitism-based strategy to reserve some excellent individuals;
  Selection of parents based on the fitness and density values;
  Crossover based on the unfitness value;
  Mutation;
  Population update using the memory library;
  $i \leftarrow i + 1$;
end
Output the best individual in the current population as the required solution;

- **Population Initialization.** We use a one-dimension vector $A = [t_1, t_2, \ldots, t_n]$ to represent an individual, where t_v $(1 \leqslant v \leqslant |\mathbb{V}|)$ denotes the amount of energy allocated for node v, and $|\mathbb{V}|$ is the cardinality of $\mathbb{V}$ representing the number of sensor nodes in the network. Initially, we randomly generate N_p individuals by choosing a random integer between 0 and K for each t_v, resulting in an initial population. The generated individuals satisfy Constraint (5) but possibly violate Constraint (4). That is, the population may include some infeasible solutions. Our objective is not only to maximize the total precision but also to improve the feasibility of the individuals. Let $A_l (l = 1, 2, \ldots, N_p)$ denote an individual in the population, which is a vector consisting of the integer elements $t_v^l (v = 1, 2, \ldots, |\mathbb{V}|)$.
- **Individual Evaluation.** Conventionally, for a solution, its quality is accurately evaluated by the fitness score, which is determined by the fitness function [25]. However, here we use three metrics, namely *fitness*, *unfitness* and *density*, to jointly evaluate an individual. The fitness of an individual is the same as the objective function value. That is,

$$f(A_l) = W_S. \tag{6}$$

Intuitively, a higher fitness value means a larger data aggregation precision. The unfitness, denoting the degree of an individual's infeasibility, is calculated as:

$$u(A_l) = \sum_{v \in \mathbb{V}} |\max(0, \Omega_v - e_v)|. \tag{7}$$

The individual A_l is feasible if $u(A_l) = 0$ and infeasible if $u(A_l) > 0$, and lower unfitness means higher infeasibility. The density of an individual A_l, denoting the proportion of the *similar individuals* of A_l in a population, is defined as:

$$c(A_l) = \frac{1}{N_p} \sum_{j=1}^{N_p} S_{l,j}. \tag{8}$$

Here $S_{l,j}$ is an indicator where $S_{l,j} = 1$ means that the two individuals A_l and A_j are considered similar and $S_{l,j} = 0$ otherwise. To be more specific, if the corresponding vectors of A_l and A_j have at least $R_{l,j}$ identical elements (irrespective of their indices), then we have

$$S_{l,j} = \begin{cases} 1, & \text{if } \frac{R_{l,j}}{|\mathbb{V}|} > Thre \\ 0, & \text{otherwise.} \end{cases}. \tag{9}$$

Here $Thre \in [0, 1)$ is a predetermined threshold. For instance, if two individuals, each with length of 6, are $[2, 4, 3, 5, 8, 1]$ and $[5, 6, 2, 1, 6, 3]$ respectively, and $Thre$ is 0.6, then they have four identical elements (1, 2, 3 and 5), and satisfy $\frac{4}{6} > 0.6 = Thre$. Thus we can say that they are similar individuals.

- **Memory Library Update.** The memory library is used to reserve some excellent individuals, which will be helpful to producing the next-generation population [13]. Let N_{ml} be the size of the memory library. We employ an *elitism* based strategy for memory library update. Specifically, we pick the individuals with the highest N_{ml} fitness values, with which the old N_{ml} ones in the memory library are replaced.
- **Selection Operation.** Selection plays an important role in improving the average quality of the population by passing the high quality individuals to the next generation. Traditionally, the selection operator is designed to ensure that better individuals have a greater probability of being selected. In this case, the search space may not be global and the solutions are likely to converge to the nearest local optimum. In order to prevent some very fit individuals from gaining dominance early, we need to enhance the diversity of the individuals consistently. To this end, we associate an expected selection probability with each individual, which is defined as

$$P_l^{sel} = \theta \frac{f(A_l)}{\sum_{j=1}^{N_p} f(A_j)} + (1 - \theta) \frac{c(A_l)}{\sum_{j=1}^{N_p} c(A_j)}. \tag{10}$$

Here $\theta \in (0, 1)$ is a constant. We can see that the expected selection probability of an individual is jointly determined by its fitness and density. In this way, we make sure that not only the individuals with higher fitness are encouraged to reproduce but also the individuals with higher density are restrained to reproduce, which guarantees the population's diversity. We then use the *roulette wheel method* [13] to probabilistically select individuals for reproducing based on P_l^{sel}.
- **Crossover Operation.** We use the unfitness-based crossover method to produce a child individual and improve the quality of the descendant. Let A_{P1}, A_{P2} and A_C denote two parent individuals and a child individual, which consist of elements k_v^{P1}, k_v^{P2}, and k_v^C, respectively. If two parents have the same elements with a common index, then the child will inherit the identical element. If the corresponding elements are different, then the parent with lower unfitness will be accepted with higher probability. To be more specific, for all $v = 1, 2, \ldots, |\mathbb{V}|$, if $k_v^{P1} \neq k_v^{P2}$, then $k_v^C = k_v^{P1}$ with probability $\frac{u(A_{P2})}{u(A_{P1}) + u(A_{P2})}$ and $k_v^C = k_v^{P2}$ with probability $\frac{u(A_{P1})}{u(A_{P1}) + u(A_{P2})}$.

- **Mutation Operation.** Mutation is applied to each child individual after crossover operation and helps to avoid the loss of valuable genetic information. We adopt a single-point mutation scheme by randomly picking two elements of each child individual and exchanging their values.
- **Population Update.** We use the memory library to update the population. Specifically, we randomly choose N_{ml} individuals from the current population and replace them with the N_{ml} individuals stored in the memory library.

In CEA, the above operations are repeated until the total number of generations N_{iter} is reached. Then we will output the best individual in the current population as the approximation solution to the MDAP problem.

4.3. Algorithm analysis and discussion

In each iteration of the CEA algorithm, the time/memory consumption both linearly increase with the problem size $|\mathbb{V}|$. Additionally, we need N_{iter} iterations to finish CEA. Thus, the time/memory complexity of CEA can be roughly obtained as $\mathcal{O}(|\mathbb{V}| \cdot N_{iter})$. For real applications, we can carefully set N_{iter} to obtain a near-optimal solution within a reasonable time limit.

Note that CEA is a centralized algorithm that can be executed by the sink, which is of much stronger computational capability than sensor nodes. In addition, recent hardware implementations of GAs [29] have shown their ability of fast computation. In this regard, the proposed CEA is promising for addressing the MDAP problem in the complicated case of unreliable WSNs. In practical implementations of CEA, the sink is assumed to be aware of the global state information of the network, which actually, can be achieved via a data gathering process. Then, the sink runs the CEA algorithm to obtain near-optimal energy allocation solutions. Finally, the obtained solutions are disseminated from the sink to the entire network via a broadcast or flooding process [30].

Given that the computation of the energy allocation scheme for data aggregation is an infrequent action (possibly performed only once), the communication overload involved in the collection of network state information and the dissemination of the computed solutions appears as an acceptable up-front cost, especially for networks of static topology and small/medium sizes. However, when the network size is large or the network topology changes frequently due to node failure/movement, the centralized algorithm will be hardly applicable due to its tremendous communication overload. In these cases, localized or distributed algorithms, in which each node determines its energy allocation using only the partial information exchanged with its neighbours, are preferred.

5. Distributed energy allocation algorithm

Our Distributed Energy Allocation (DEA) algorithm is inspired by the Gibbs sampling theory [14]. In this section, we first introduce the general idea of the Gibbs sampler, then describe the details of DEA, and finally, analyze the convergence performance as well as the communication complexity of DEA.

5.1. The Gibbs sampler

Gibbs sampler is an optimizer that performs randomized local adaptation with the objective of optimizing the global system objective [31]. By exploiting the fact that the global optimum may be associated with the stationary version of a stochastic process governed by simple local interactions, Gibbs sampler can achieve very good properties (e.g., convergence and optimality) in solving global optimization problems. In general, the convergence and optimality properties follow from the classical framework of Markov Random Fields and Gibbs Measures [27].

Consider an undirected graph $\mathbb{G}$ that contains a set $\mathbb{U}$ of nodes. Each node $u \in \mathbb{U}$ can be in a state within a finite state space $\mathbb{X}_u$. The nodes in $\mathbb{U}$ are cooperative to maximize a global objective function by individually controlling their local states. In Gibbs sampling, the maximization objective can be achieved if the global function is of the form,

$$F(\cdot) = \sum_{z} \sum_{C \in \mathbb{C}_z} M(C),$$

where $\mathbb{C}_z$ is the collection of all cliques of order z, and $M(\cdot)$ is the potential function of the Gibbs sampler. $M(\cdot)$ is defined such that $M(C)$ depends only on the local states of nodes in C, and is zero if C is not a clique [26]. Then, a local objective function is associated with each node $u \in \mathbb{U}$,

$$F_u(\cdot) = \sum_{z} \sum_{C \in \mathbb{C}_z : u \in C} M(C).$$

In Gibbs sampling, each node updates its state over time, in either a synchronous or a asynchronously manner, based on some probability distribution. To be more specific, at time τ, node u makes an update on its local state, and chooses to the in state $\mu \in \mathbb{X}_u$ with probability.

$$\pi_\tau(X_u = \mu) = \frac{e^{F_u(\mu)/H_\tau}}{\sum_{\mu' \in \mathbb{X}_u} e^{F_u(\mu')/H_\tau}}. \tag{11}$$

Here H_τ denotes the temperature of the system at time τ. In order to establish the convergence of the Gibbs sampling, we can interpret the update of local states as transitions of a reversible Markov chain. According to the Hammersley–Clifford theorem [32], a Gibbs Field is defined by (11). It has been shown that as the system temperature $H_\tau \to 0$, the distribution concentrates on the set of local states $\mathbb{S}^*$ where the global objective function reaches its maximum [27]. Next we will employ the principles of the Gibbs Sampling to develop the DEA algorithm for solving the MDAP problem.

5.2. Description of DEA

Before presenting the details, we first introduce some essential notations and definitions used in DEA. We

consider an undirected graph $\mathbb{G}$ the same as the WSN topology $\mathbb{T}$, i.e. $\mathbb{G} = (\mathbb{V} \bigcup \{S\}, \mathbb{E})$. $\mathbb{G}$ is also referred to as the *interaction graph* in the Gibbs Sampling theory [27]. The edges in $\mathbb{E}$ determines the neighbour relationship among the nodes in $\mathbb{G}$. We denote the set of neighbours of node $v \in \mathbb{V}$ by $\mathbb{N}_v^{(1)} = \{u \in \mathbb{V} | (u, v) \in \mathbb{E}\}$. In our case, $\mathbb{N}_v^{(1)}$ is equal to $\mathbb{C}(v) \cup \{P(v)\}$. Further, by adding node v itself, we obtain the extended neighbour set of node v, denoted by $\mathbb{N}_v^{(1+)} = \mathbb{N}_v^{(1)} \cup \{v\}$. We associate each node in $v \in \mathbb{V}$ with a local state X_v representing its energy allocation t_v. The local state space of node v, denoted by $\mathbb{X}_v$, can be obtained as

$$\mathbb{X}_v = \left\{ X_v | 0 \leqslant X_v \leqslant K, \alpha_v^{TX} \cdot X_v + \alpha_v^{RX} \cdot \sum_{u \in \mathbb{C}(v)} X_v \leqslant e_v \right\}. \tag{12}$$

The *local objective function* $F_v(\cdot)$ of node v is defined as:

$$F_v(X_v) = \phi_v \cdot \gamma_v + \sum_{u \in \mathbb{C}(v)} (\phi_u \cdot \gamma_u). \tag{13}$$

From this definition, we can see that the local state of node v only affects the local objective functions of the nodes in $\mathbb{N}_v^{(1+)}$. In addition, the local objective function of node v depends only on the local states of nodes in $\mathbb{N}_v^{(1+)}$.

Furthermore, we define a *Gibbs sampling graph* [27] $\mathbb{G}^* = (\mathbb{V} \bigcup \{S\}, \mathbb{E}^*)$, where $(u, v) \in \mathbb{E}^*$ if and only if $u \in \mathbb{N}_v^{(2)}$, with $\mathbb{N}_v^{(2)} = (\bigcup_{u \in \mathbb{N}_v^{(1)}} \mathbb{N}_u^{(1+)}) \setminus \{v\}$ representing the *"two-tier" neighbour set* of node v. As we have defined, for each node v, $\mathbb{N}_v^{(1+)}$ denotes the set of one-hop neighbours of v, plus node v itself. According to the definition of the Gibbs sampling graph $\mathbb{G}^*$, any two nodes in $\mathbb{N}_v^{(1+)}$ are connected in $\mathbb{G}^*$. Thus, we can see that the neighbour set $\mathbb{N}_v^{(1+)}$ in $\mathbb{G}$ constitutes a clique in $\mathbb{G}^*$. Actually, this finding reveals the relationship between the interaction graph $\mathbb{G}$ and the Gibbs sampling graph $\mathbb{G}^*$.

For the clique $\mathbb{N}_v^{(1+)} (\forall v \in \mathbb{V})$ in $\mathbb{G}^*$, we define its potential as $F_v(X_v)$, and for any other clique in $\mathbb{G}^*$, we let its potential be zero. Then, the global function objective to be maximized can be obtained as:

$$F(\cdot) = \sum_z \sum_{C \in \mathbb{C}_z} M(C) = \sum_{C \in \mathbb{C}_z} \sum_{v : \mathbb{N}_v^{(1)}} F_v(\cdot) = \sum_{v \in \mathbb{V}} F_v(\cdot). \tag{14}$$

Based on (13) and (14), the global objective function $F(\cdot)$ can be derived. Now, if each node makes an update on its local state based on the probability distribution in (11), then the maximization of the global function $F(\cdot)$ will be achieved. Note that in DEA, we use $F(\cdot)$ to approximate the original objective function of the MDAP problem to facilitate distributed implementation. Thus, the solution obtained by DEA can only be sub-optimal. In the next section, we will use numerical experiments to demonstrate that the performance gap between DEA and the near-optimal CEA algorithm is not high. To summarize, the pseudo-code of DEA is shown in Algorithm 2.

Algorithm 2. Distributed Energy Allocation (DEA) Algorithm

Assumptions: Each node $v \in \mathbb{V}$ knows the initial temperature H_0 of the system, the temperature "cooling" function $h(\tau)$, the upper bound of energy allocation K, and the maximum number of iterations N_{iter}. Similar to [27], each node has a clock with continuous ticks. The time periods between two consecutive ticks of the clock are *i.i.d.* random variables. The clocks of different nodes are mutually independent;
Initialization: Each node $v \in \mathbb{V}$ randomly picks an integer from $\{0, 1, \cdots, K\}$ as the initial energy allocation. The time τ is initialized to zero. Additionally, an iteration counter i is initialized to 1;
while $i \leq N_{iter}$ **do**
  for *each node v, if its time clock ticks,* **do**
    Computes the current temperature using the equality: $H_\tau = H_0 \cdot h(\tau)$;
    Sends request messages to acquire the local states of all nodes in its two-tier neighbour set $\mathbb{N}_v^{(2)}$;
    Determines its local state space $\mathbb{X}_v$ using (12);
    Picks μ from $\mathbb{X}_v$ as the updated local state, according to probability mass function in (11);
  end
  for *each node v, if it receives a request message,* **do**
    Returns back its current local state to the sensder;
  end
  $i \leftarrow i + 1$;
end

5.3. Analysis of DEA

The convergence of DEA can be analyzed as follows. As mentioned in [31], the setting of the temperature H_τ influences the limit distribution to be reached by the system. This parameter needs to be chosen as a tradeoff between the convergence time and the optimality of the limit distribution. In [31], the authors show that if one decreases H_τ as $1/\log(\tau)$, then the Gibbs sampling can enable the system to reach a state where the global objective function attains its maximum value, starting from any initial sate. Following this known result, we set $H_\tau = H_0 \cdot 1/\ln(\tau + 1)$, where τ starts from zero. A strict proof of the convergence of Gibbs sampling can be derived similarly to that of [31], based on the properties of Markov chains. In the next section, we will use numerical study to further illustrate the convergence performance of the DEA algorithm.

For any distributed algorithm, an important performance aspect will be the communication overhead incurred by exchanging control messages while the algorithm is executed. In DEA, each iteration requires the updating node to obtain the local states of all nodes in its two-tier neighbour set. Let δ be the maximum node degree of the graph $\mathbb{G}$, and N_{iter} be the number of iterations when DEA converges. Then the worst communication complexity

of each node will be $\mathcal{O}(\delta^2)\cdot N_{iter}$. Note that if the node degree δ is high, for instance, when large number of nodes are crowded in a small area, then the communication overhead can still be significant. In this case, alternative implementations of DEA can be considered to further lower down the communication complexity. For example, each node can compute its local objective function value and communicate it to the updating node, in a "push" rather than "pull" manner [27]. Also, the updating process in DEA can be modified to use possibly outdated information. Generally, there exists a fundamental tradeoff between the communication overhead and the convergence rate of DEA. That is, a substantial reduction of the communication overhead will typically come at the expense of a lower convergence rate.

6. Performance evaluation

In this section, the performance of both CEA and DEA are evaluated using extensive MATLAB experiments. Generally, our simulations involve two parts. In the first part, we consider small networks consisting of only 20 nodes, and evaluate the performance of our algorithms with comparison to the optimal solution found by exhaustive search. In this part, we also evaluate the complexity of the centralized algorithm by measuring its practical running time, and that of the distributed algorithm by counting the number of exchanged control messages. In the second part, we consider larger deployments with 100 nodes, and compare our algorithms with other related algorithms in terms of data aggregation precision, total energy consumption, etc.

6.1. Experiment settings

We consider such a deployment, where a set of nodes are uniformly distributed in a square region. Each node has a fixed and uniform communication range, and a link exists between two nodes if they are within the communication range of each other. The size of network scale, deployment region, and node communication range will be specified later in each set of simulations. After deployment, we can obtain a connected communication graph G, which is formed by all the nodes and the corresponding links. Among these nodes, the one that is the closest to the centre of the region is selected as the sink node, and the other nodes are considered as sensor nodes. We then construct a tree T by running a breadth-first-search (BFS) algorithm [30] starting at the sink node. This tree is referred to as data aggregation tree, and the data aggregation operation is conducted by using this tree as the underlying routing infrastructure. We assume an unreliable WSN, and the link error probabilities are randomly selected in the range of $(p-0.05, p+0.05)$, where $p\in\{0.1,0.2,\ldots,0.9\}$ is to be determined later in the following experiments. The data precision that each node (except the sink) can provide is one, i.e. $\phi_v=1,\ \forall v\in\mathbb{V}$. Each transmission/reception takes one unit of energy, i.e. $\alpha_v^{TX}=\alpha_v^{RX}=1$, for any sensor node v. In our case, retransmission is used to account for link loss. Thus, an allocation of k units of energy for node v means that at most k retransmissions can be made by node v for packet delivery. Preliminary simulations suggest that in order for CEA to perform best, the corresponding parameters of CEA need to be set as follows: the population size $N_p=40$, the memory library size $N_{ml}=5$, the number of iterations $N_{iter}=30$, and the positive constants $Thre=0.6,\ \theta=0.7$. Besides, for DEA, the initial system temperature H_0 is set as 1, and the temperature "cooling" function $h(\tau)$ is chosen as $\ln(\tau+1)$. For DEA, the maximum number of iterations is also set as 30.

Table 1
Comparison of CEA and optimal algorithm in a small-scale network.

p	Aggregation precision		Running time (hours)	
	CEA	Optimal	CEA	Optimal
0.1	17.236	18.025	0.086	5.412
0.2	16.202	17.470	0.079	5.256
0.3	15.034	15.973	0.082	5.387
0.4	13.188	14.242	0.081	5.692
0.5	10.860	12.038	0.092	5.256
0.6	8.132	9.493	0.078	5.353
0.7	6.287	7.045	0.086	5.379
0.8	3.579	4.022	0.083	5.582
0.9	1.835	2.101	0.074	5.776

6.2. Small-scale network simulations

We first consider a small deployment, where 20 nodes are uniformly distributed in a 50×50 square region, and each node has a communication range of 20. Given that the number of nodes is small, either standard optimization tools such as CVX [33] or brute-force exhaustive search method can be used to find the optimal solution to the MDAP problem. Such an *optimal* method serves as a good benchmark to evaluate the performance of the centralized CEA algorithm. In this set of simulations, we assume that each node has an energy budget of 5 units. Besides, the upper bound of energy allocation, i.e. the parameter K, is also set to 5.

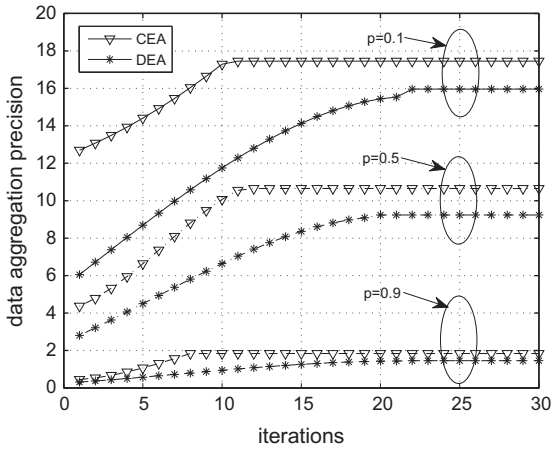


Fig. 2. Snapshot iteration processes of CEA and DEA under different values of p.

Under the same simulation environment of Intel Core2 2 GHz CPU and MATLAB 7.1 tool, we obtain the running time and the total data aggregation precision of the compared algorithms as summarized in Table 1. The results shown here are the average results of 30 random instances. From Table 1, we observe that under all values of p, CEA performs very closely to the optimal algorithm. Specifically, CEA can achieve at least 83.8% of the optimal precision (when $p = 0.5$), and at most 95.6% of the optimal precision (when $p = 0.1$). We also can observe that CEA needs much shorter running time than the optimal exhaustive search method. For example, when $p = 0.3$, the average running time of CEA is only 0.082 h (about 5 min), while the exhaustive search method needs as long as 5.387 h to find the optimal solution. Thus, we can conclude from Table 1 that CEA is indeed an efficient approximation algorithm that achieves nearly optimal data aggregation precision using little computational resources.

Note that CEA and DEA are both iterative algorithms. For any iterative algorithm, an important performance aspect would be the convergence. Our simulations show that both CEA and DEA converge very well. For the above small-scale network and under the same configurations, we obtain some snapshot iteration processes of CEA and DEA as shown in Fig. 2 to explicitly demonstrate their convergence performance. We can observe that both algorithms converge quickly to steady status. Specifically, CEA converges in about 10 iterations, and DEA converges in about 20 iterations. To compare, we can see that DEA has slower convergence rate and slightly worse precision than CEA. Essentially, this can be explained that CEA is a centralized algorithm capable of finding near-optimal energy allocation solutions, while the localized DEA algorithm can only achieve sub-optimal ones.

For the distributed DEA algorithm, we further evaluate its complexity (a.k.a, overhead) by counting the number of exchanged control messages. To be more specific, we measure the maximum/average number of messages a node has transmitted when the DEA algorithm terminates, and Fig. 3 shows such results. For fairness, we do not count the number of lost control messages (due to link unreliability) while the algorithm is executed iteratively. We can see that each node transmits 90 messages on average (dashed line) regardless of the value of p. The maximum overhead (solid line) of nodes in the network is only about 1.5 times of the average overhead. The simulation results show that the overhead of the network is tolerable and thus our distributed algorithm works. In general, for an iterative algorithm requiring a number of iterations to converge, the overhead of each node would be mainly determined by the algorithm's convergence rate. To cut down per-node overhead and save more energy, we need to investigate how to further speed up the convergence process of DEA, which we leave as part of our future work.

6.3. Large-scale network simulations

We then consider a larger deployment with 100 nodes uniformly distributed in a 150×150 square region. Unless specially pointed out, the other configurations and parameters are the same as above. Under the large deployment,

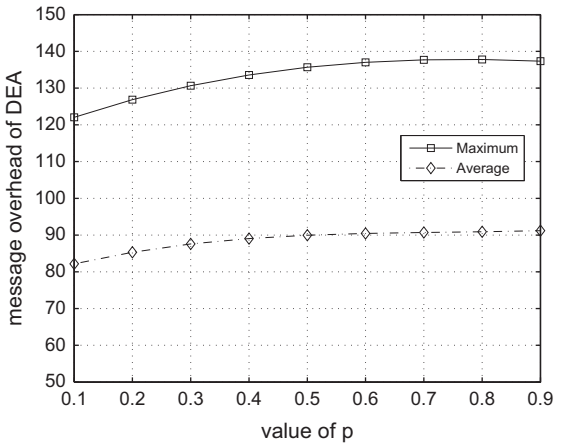


Fig. 3. Maximum/average number of messages a nodes transmits under the DEA algorithm.

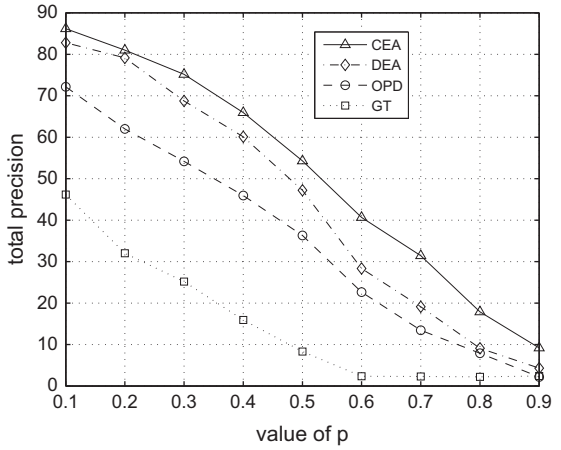


Fig. 4. Comparison of data aggregation precision under various link loss probabilities (p).

we compare CEA and DEA with two related algorithms, including the OPD algorithm in [7] and a heuristic algorithm named Greedy Transmission (GT), in terms of data aggregation precision, total energy consumption, etc. As an algorithm developed by assuming always reliable links, OPD does not account for link loss and allows each node to make at most one transmission during one round of data aggregation. On the other hand, GT is a greedy energy allocation algorithm. That is, starting from the leaves and in a bottom-up manner, each node greedily makes at most K attempts to transmit until either a transmission is successful or the energy bound is reached.

Fig. 4 shows the total data aggregation precision of the four algorithms with varied link loss probabilities. We can observe that as the link loss probability p increases, the aggregation precision values of the algorithms all decrease quickly. This can be explained as follows. As the link quality becomes more worse, each node needs more energy in order to handle the link unreliability. However, each node

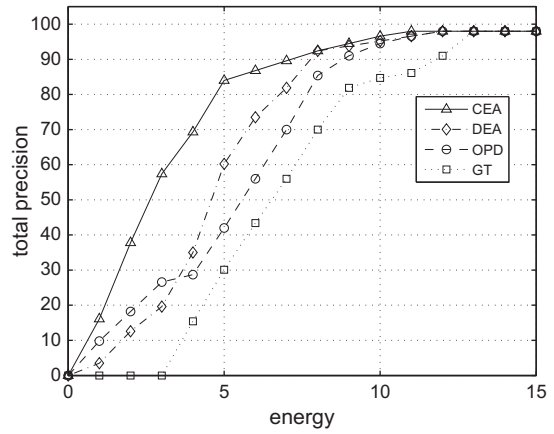


Fig. 5. Comparison of data aggregation precision under various per-node energy bounds.

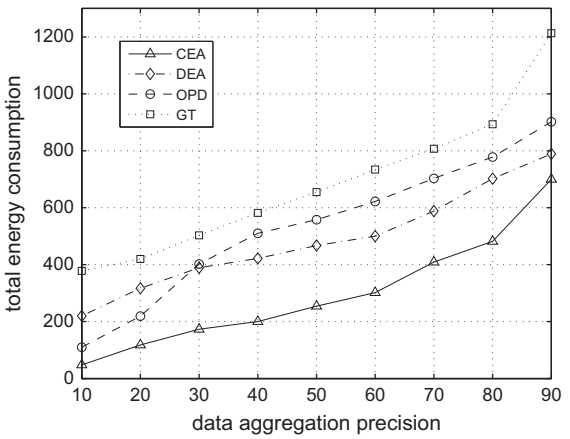


Fig. 6. Comparison of energy consumption under various data aggregation precision requirements.

has a finite energy budget (specifically, in our case, an energy bound of 5 units is imposed). Thus, the nodes have to drop some packets during the packet forwarding process, which means that some precision will be lost and the total precision received by the sink will be reduced. Nevertheless, we can see that CEA and DEA perform much better than the other two algorithms. For instance, when $p = 0.3$, the precision of CEA is about 1.5 times that of OPD, and 3 times that of GT.

Fig. 5 shows the total data aggregation precision of the four algorithms with varied per-node energy constraints. Here we fix the link loss probability p as 0.2. We can observe that when the energy bounds are high ($e_i > 12$), the performance of all algorithms are close, which is not surprising since each node has sufficient energy to enhance link reliability. However, when the energy bounds are lower, CEA and DEA perform better than the other two algorithms. Specifically, we can see that the OPD performs worse for the majority of the energy bounds compared to CEA. The reason behind is that OPD does not take the link errors into consideration, while CEA allows near-optimal energy utilization for all nodes, resulting in more rational transmission attempts. In addition, we can see that although GT accounts for link unreliability, it still performs the worst. The reason is that GT makes greedy instead of near-optimal retransmissions, which may causes some nodes to exhaust their energy before transmitting the aggregated data to their parents. Thus, the precision of the aggregated packet obtained by the sink could diminish significantly. In both Figs. 4 and 5, we can see that the performance of DEA is inferior to that of CEA. However, the performance gap is not high. Recall that DEA is a distributed algorithm requiring only partial network information to make energy allocation decisions. Thus, DEA appears as a practical solution to the MDAP problem when sub-optimal performance is enough for satisfying the application's requirement.

Finally, we conduct some simulations to compare the energy consumption of the four algorithms under specific data aggregation precision requirement. We fix the link loss probability p as 0.2, and assume that each node has infinite energy resources. Given an imposed data aggregation precision constraint, we measure the total energy consumption (in units) of the four algorithms, and Fig. 6 shows such results. We can see that as the data precision requirement increases, the energy consumption also increases monotonously. Similar to previous experiment results, the performance of CEA is better than the other algorithms, and the performance of GT is still the worst. The reason behind can be explained in a similar manner as the previous discussions, and we omit it here for compactness.

7. Conclusion and future work

We have studied the problem of optimizing data aggregation precision for a practical type of WSNs where links are unreliable. For solving the problem, we have developed both centralized and distributed algorithms followed by analysis of algorithmic complexity and discussions of implementation issues. Numerical simulations were also conducted to demonstrate the convergence as well as the data aggregation precision performance of the proposed algorithms. Future work will be conducted through extending the current research to account for other important constraints in WSNs such as wireless interference and delay.

Acknowledgements

This work was partially supported by National Program on Key Basic Research Project of China (973 Program) under Grant No.2011CB302906, and National Science and Technology Major Project of the Ministry of Science and Technology of China under Grant No. 2010ZX03006-004. The authors would also thank the reviewers for their helpful comments and advices to improve the presentation of the paper.

References

[1] F. Wang, J. Liu, Networked wireless sensor data collection: issues, challenges, and approaches, IEEE Commun. Surv. Tutorials 13 (4) (2011) 673–687.

[2] E. Fasolo, M. Rossi, J. Widmer, M. Zorzi, In-network aggregation techniques for wireless sensor networks: a survey, IEEE Wirel. Commun. 14 (2) (2007) 70–87.

[3] T. Pham, E.J. Kim, M. Moh, On data aggregation quality and energy efficiency of wireless sensor network protocols-extended summary, in: IEEE First International Conference on Broadband Networks, 2004, pp. 730–732.

[4] X. Tang, J. Xu, Optimizing lifetime for continuous data aggregation with precision guarantees in wireless sensor networks, IEEE/ACM Trans. Netw. 16 (4) (2008) 904–917.

[5] S. Xiao, J. Huang, L. Pan, Y. Cheng, J. Liu, On centralized and distributed algorithms for minimizing data aggregation time in duty-cycled wireless sensor networks, Wireless Netw. 20 (7) (2014) 1727–1741.

[6] H.-C. Lin, F.-J. Li, K.-Y. Wang, Constructing maximum-lifetime data gathering trees in sensor networks with data aggregation, in: IEEE International Conference on Communications, 2010, pp. 1–6.

[7] W. Qu, K. Li, M. Kitsuregawa, T. Nanya, An efficient method for improving data collection precision in lifetime-adaptive wireless sensor networks, in: IEEE International Conference on Communications, 2007, pp. 3161–3166.

[8] J. Zhu, S. Papavassiliou, J. Yang, Adaptive localized QoS-constrained data aggregation and processing in distributed sensor networks, IEEE Trans. Parall. Distrib. Syst. 17 (9) (2006) 923–933.

[9] S. Cheng, J. Li, Q. Ren, L. Yu, Bernoulli sampling based (ε, δ)-approximate aggregation in large-scale sensor networks, in: Proceedings of the 29th Conference on Information Communications, 2010, pp. 1181–1189.

[10] Z. Taghikhaki, N. Meratnia, P.J. Havinga, Energy-efficient trust-based aggregation in wireless sensor networks, in: IEEE Conference on Computer Communications Workshops, 2011, pp. 584–589.

[11] S. Ji, R. Beyah, Z. Cai, Snapshot/continuous data collection capacity for large-scale probabilistic wireless sensor networks, in: Proceedings of IEEE INFOCOM, 2012, pp. 1035–1043.

[12] D. Gong, Y. Yang, Z. Pan, Energy-efficient clustering in lossy wireless sensor networks, J. Parall. Distrib. Comput. 73 (9) (2013) 1323–1336.

[13] J. Lu, N. Fang, D. Shao, C. Liu, An improved immune-genetic algorithm for the traveling salesman problem, in: IEEE Third International Conference on Natural Computation, vol. 4, 2007, pp. 297–301.

[14] C.S. Chen, F. Baccelli, Self-optimization in mobile cellular networks: power control and user association, in: IEEE International Conference on Communications, 2010, pp. 1–6.

[15] P. Wang, Y. He, L. Huang, Near optimal scheduling of data aggregation in wireless sensor networks, Ad Hoc Netw. 11 (4) (2013) 1287–1296.

[16] H. Li, C. Wu, Q.-S. Hua, F. Lau, Latency-minimizing data aggregation in wireless sensor networks under physical interference model, Ad Hoc Netw. 12 (2014) 52–68.

[17] K. Kalpakis, K. Dasgupta, P. Namjoshi, Maximum lifetime data gathering and aggregation in wireless sensor networks, in: IEEE International conference on networking, 2002, pp. 685–696.

[18] I. Tan, Power efficient data gathering and aggregation in wireless sensor networks, ACM Sigmod Record 32 (4) (2003) 66–71.

[19] K. Kalpakis, K. Dasgupta, P. Namjoshi, Efficient algorithms for maximum lifetime data gathering and aggregation in wireless sensor networks, Comput. Netw. 42 (6) (2003) 697–716.

[20] R.R. Rout, S.K. Ghosh, Adaptive data aggregation and energy efficiency using network coding in a clustered wireless sensor network: an analytical approach, Comput. Commun. 40 (0) (2014) 65–75.

[21] S. Guo, Y. Gu, B. Jiang, T. He, Opportunistic flooding in low-duty-cycle wireless sensor networks with unreliable links, in: Proceedings of the ACM 15th Annual International Conference on Mobile Computing and Networking, 2009, pp. 133–144.

[22] H. Yousefi, M.H. Yeganeh, A. Movaghar, Long lifetime routing in unreliable wireless sensor networks, in: IEEE International Conference on Networking, Sensing and Control, 2011, pp. 457–462.

[23] Y. Gu, T. He, Dynamic switching-based data forwarding for low-duty-cycle wireless sensor networks, IEEE Trans. Mob. Comput. 10 (12) (2011) 1741–1754.

[24] G. Xing, C. Lu, X. Jia, R. Pless, Localized and configurable topology control in lossy wireless sensor networks, Ad Hoc Netw. 11 (4) (2013) 1345–1358.

[25] R.V. Kulkarni, A. Forster, G.K. Venayagamoorthy, Computational intelligence in wireless sensor networks: a survey, IEEE Commun. Surv. Tutorials 13 (1) (2011) 68–96.

[26] P. Arora, N. Xia, R. Zheng, A gibbs sampler approach for optimal distributed monitoring of multi-channel wireless networks, in: IEEE Global Telecommunications Conference, 2011, pp. 1–6.

[27] S. Borst, M. Markakis, I. Saniee, Nonconcave utility maximization in locally coupled systems, with applications to wireless and wireline networks, IEEE/ACM Trans. Netw. 22 (2) (2014) 674–687.

[28] L. Jiao, L. Wang, A novel genetic algorithm based on immunity, IEEE Trans. Syst. Man Cybern. Part A: Syst. Humans 30 (5) (2000) 552–561.

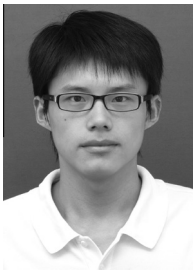
[29] T. Lu, J. Zhu, Genetic algorithm for energy-efficient QoS multicast routing, IEEE Commun. Lett. 17 (1) (2013) 31–34.

[30] S. Xiao, J. Pei, X. Chen, W. Wang, Minimum latency broadcast in the SINR model: a parallel routing and scheduling approach, IEEE Commun. Lett. 18 (6) (2014) 1027–1031.

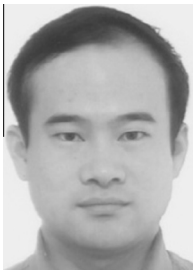
[31] I.-H. Hou, P. Gupta, Distributed resource allocation for proportional fairness in multi-band wireless systems, in: IEEE International Symposium on Information Theory Proceedings, 2011, pp. 1975–1979.

[32] P. Bremaud, Markov Chains: Gibbs Fields, Monte Carlo Simulation, and Queues, vol. 31, Springer, 1999.

[33] M. Elsersy, J. Rao, Linear approximation for energy and throughput optimization in wireless sensor networks, in: IEEE Consumer Communications and Networking Conference, 2012, pp. 705–707.



Shiliang Xiao received the B.S. degree in Electronic and Communication Engineering from Zhejiang University, Hangzhou, China, in 2010 and is currently pursuing the Ph.D. degree in Communication and Information System at Shanghai Institute of Microsystem and Information Technology, Chinese Academy of Sciences (CAS), Shanghai, China. He is also with the University of Chinese Academy of Sciences, Beijing, China. Dr. Xiao is a Student Member of CCF (China Computer Federation) and a Student Member of the IEEE. His research interests include wireless ad hoc and sensor networks, algorithm design and analysis, and cloud computing.



Baoqing Li received the B.S. degree in Physics from Nanchang University, Nanchang, China, in 1993, and Ph.D. degree in Electronic Engineering from Shanghai Microsystem and Information Technology (SIMIT), Chinese Academy of Sciences, Shanghai, China, in 1998. From 1998 to 2001, he worked at SIMIT as a branch manager. From 2001 to 2005, he worked at New Jersey Institute of Technology, Newark, USA as a researcher. He has been a Professor and a Ph.D. supervisor at SIMIT since 2006. Prof. Li's research interests lie in wireless sensor networks, MEMS systems, etc.

Xiaobing Yuan received the B.S. degree in Electronic Engineering from Zhejiang University, Hangzhou, China, in 1991, and the Ph.D. degree in Electronic Engineering from Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Sciences. From 2001 to 2002, he worked at Shanghai Engineering Center for Microsatellites as a Post-doctor. He has been a Professor and a Ph.D. supervisor at Shanghai Institute of Microsystem and Information Technology (SIMIT), Chinese Academy of Sciences, since 2003. He also leads the Wireless Sensor Networks Laboratory at SIMIT. Prof. Yuan's research interests include wireless sensor networks and signal processing.