Computer Communications 34 (2011) 2217–2226

Contents lists available at ScienceDirect

Computer Communications

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

Energy efficient joint data aggregation and link scheduling in solar sensor networks

Jianhui Zhang [a,c,*], Xingfa Shen [a], Shaojie Tang [b], Guojun Dai [a]

[a] Institute of Computer Application Technology, Hangzhou Dianzi University, Hangzhou 310018, China
[b] Illinois Institute of Technology, Chicago, IL 60616, USA
[c] SITE, University of Ottawa, Ottawa, Canada K1N 6N5

ARTICLE INFO

Article history:
Received 4 April 2011
Received in revised form 18 June 2011
Accepted 18 June 2011
Available online 25 June 2011

Keywords:
Energy harvesting
Solar node
CDS
Link scheduling

ABSTRACT

Solar sensor nodes equipped with micro-solar subsystems [1] provide a novel approach to harvest ambient energy, which partially alleviated the energy-limitation in traditional wireless sensor networks. However, it also poses new challenges that the amounts of energy harvested by nodes are dynamic and unbalanced among them thus network life cannot be necessarily prolonged if no well-designed energy-scheduling is adopted. Herein, we present an algorithm to construct Energy-efficient Data Aggregation Tree (EDAT) based on a Maximum-Weighted Connected Dominating Set (MaCDS). The EDAT aims to prolong network life by minimizing differences in energy consumption among sensor nodes. Here we assume that the amount of harvested energy $\mathcal{H}$ randomly and uniformly distributes in the interval $[\mathcal{H}_{min}, \mathcal{H}_{max}]$, where $\mathcal{H}_{max}$ and $\mathcal{H}_{min}$ are respectively the maximum and the minimum of $\mathcal{H}$. The total energy consumption difference of an EDAT is at most $\frac{5\widetilde{\mathcal{H}}|\mathcal{S}|^2}{n-1}$, where $\widetilde{\mathcal{H}} = |[\mathcal{H}_{min}, \mathcal{H}_{max}]|$, and $\mathcal{S}$ is the dominating set and n the total number of nodes. Furthermore, we designed a link-scheduling algorithm to minimize the number of time slots necessary for scheduling all links in the whole network based on EDAT. The number of time slots is bounded in the interval $[4\delta - 1, 2l_k\varrho^2]$, where l_k is determined by hops $k, \varrho = \max\left\{\Delta, \left\lfloor\frac{\mathcal{H}_{max}}{\mathcal{H}_{min}}\right\rfloor\right\}$, and δ and Δ are respectively the minimal and maximal degree of the network. We found the most efficient work period is not determined by the node with the minimum harvested energy but by the one with the minimum $\frac{\mathcal{H}}{d}$, where d is the node's degree in the EDAT. We determine the necessary condition required for every node to have sufficient energy to support consecutive operation.

© 2011 Elsevier B.V. All rights reserved.

1. Introduction

When the environment energy harvesting technologies, such as solar and vibration [2], are employed, it brings new chances to prolong the network life. One of these technologies can transform the solar [3] energy into electrical power. With these technologies, various types of platforms are designed to collect environment energy, such as Heliomote, Trio, AmbiMax, PUMA and Prometheus [1,4]. Based on these platforms, solar sensor nodes harvest solar or wind energy and store harvested energy in rechargeable batteries or ultra-capacitor. However, one node can harvest different amount of energy from others. Therefore, the renewable energy resource brings new challenge on energy management and scheduling. It is different from traditional energy-related issues, which have limited energy and can partially alleviate the conflict between network lifetime and performance [4]. Permanent network lifetime cannot be obtained. Furthermore, the energy management and scheduling problem should be reconsidered when sensor nodes try to prolong the network lifetime by harvesting energy. Because it must happen that the harvested and unused energy accumulate in the rechargeable batteries of some nodes while other nodes use up their energy when no energy management and scheduling scheme is adopted.

The energy limitation is a key constraint in traditional energy scheduling, which did not consider energy harvest [5]. Although energy can be harvested and will be abundant in part of nodes, energy harvesting can not be precisely predicted in advance [6] because of variable weather. So energy management and scheduling under the dynamic harvested energy is a new challenge.

This paper aims to design energy management and scheduling according to our definition of *energy efficiency* thus a network can be supported to work consecutively in periods when each node can harvest energy from surroundings. Firstly we construct an EDAT to balance energy consumption within a network. Based on this EDAT, we design a time-assignment and transmission schedule algorithm to achieve fairness of channel access [7]. Then we analyze the necessary condition to satisfy the requirement that the network can consecutively work in periods while guaranteeing energy efficiency. The main contributions of this paper are as follows:

* Corresponding author at: Institute of Computer Application Technology, Hangzhou Dianzi University, Hangzhou 310018, China. Tel.: +86 571 86878596.
E-mail address: jhzhang.zju@gmail.com (J. Zhang).

0140-3664/$ - see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.comcom.2011.06.011

- We constructed a Maximum-Weighted Connected Dominating Set (MaCDS) based on uncertain weight by an existing Minimum-Weighted Connected Dominating Set (MiWCDS) algorithm [8] and its total weight is not less than $\frac{1}{18\eta(5+\epsilon)}$ of the optimum, where $\eta = \frac{\mathcal{H}_{max}}{\mathcal{H}_{min}}$ and $\epsilon > 0$ is a constant. Thus, the problem whereby the amount of harvested energy cannot be easily predicted can be overcame.
- Based on the MaCDS, we constructed an EDAT by using a localized algorithm to obtain an energy efficient tree. The total difference of energy consumption within the network is not more than $\frac{5\mathcal{H}|\mathcal{S}|^2}{n-1}$ in this EDAT.
- We designed a link-scheduling to achieve the fairness in the assignment of the necessary time slots $\widehat{L}$ for scheduling all links. Here, the time is bounded in the interval $[4\delta - 1, 2l_k\varrho^2]$. Thus, we determined that the necessary condition to support consecutive network operation is $\mathcal{H}_{min} \geqslant \Delta(4\delta - 1)$.

The paper is organized as follows. Section 2 introduces the network, energy harvesting, consumption and interference models. The data aggregation scheme is briefly described in Section 3, and the algorithms to construct a MaCDS and an EDAT are also presented. The design of the link-scheduling algorithm and the determination of the consecutive network operating condition are described in Section 4. Related works in this field are discussed in Section 5. Section 6 draws several conclusions and discusses the future work.

2. System model and assumptions

2.1. Energy-efficient scheduling

Previous attempts at designing energy-efficient networks saved energy by constructing an energy-efficient topology [9] and reducing the number of active time slots needed for data transmission or reception [5] to maximize the utilization of limited energy resources.

When nodes harvests different amounts of energy from their surroundings, if energy consumption is not well scheduled, some nodes may use up their energy prematurely in some periods while others retain surplus energy. In our design, we schedule the energy consumption of each node by balancing the energy efficiency of all the nodes in a network, which means that those with more harvested energy are scheduled to handle greater data loads in each period. We herein define *energy efficiency* as the reciprocal of the maximum difference in the remaining energy for a pair of nodes after they have consumed their harvested energy in a given period T. For example, a node u retains energy E^u_{remain} and another node v retains energy E^v_{remain} at the end of a given period. The difference, E_{diff}, of the remaining energy between them is $E^{uv}_{diff} = |E^u_{remain} - E^v_{remain}|$. Thus, the energy efficiency between u and v is $\frac{1}{E^{uv}_{diff}}$. We define the network capacity, i.e., the data received by the sink in each period, as D_T, and the efficient working period for each node u that is supported by the energy harvested in each period T is defined as $T_e(u)$. The efficient working period of the whole network is defined as $T_e(G) = \min_{u \in V} T_e(u)$. Our goal is to improve the network capacity, i.e., maximize $\max D_T$, so that the harvested energy of each node can support the whole network from the current period to the next, i.e., $T_e(G) \geqslant T$.

2.2. Network model

We assume that there are n solar sensor nodes comprising a set V that are *randomly and uniformly* deployed in a $\mathbf{c} \times \mathbf{c}$ area. Each node is equipped with a micro-solar subsystem, rechargeable batteries and a single radio interface. There is only one sink node in the network. Every node u has a transmission range $\Re_u$. Node u can transmit packets to another node v successfully and directly if the Euclidean distance $\|u - v\|$ between u and v satisfies the necessary condition $\|u - v\| \leqslant \Re_u$. Therefore, the edge set E is composed of the possible communication links between possible node pairs. The whole network is described as a graph G composed of V and E. If there exists a link e between u and v, the link is denoted as L_{uv}.

2.3. Energy harvesting and consumption models

There are several existing energy harvesting technologies capable of transforming ambient energy into electric energy and storing it [4]. This work studies energy harvesting by micro-solar subsystems in sensor nodes. Although it is difficult to model the solar charging pattern, we can calculate the total amount of the energy harvested by a solar sensor node u in a period T that is divided into m equal time slots τ_i, $i = 1, \ldots, m$. Here we define $\mathcal{H}_u$ as the amount of the energy harvested by u in period T. We further assumes that the harvested energy $\mathcal{H}_u$ is *uniformly and randomly* distributed in the interval $[\mathcal{H}_{min}, \mathcal{H}_{max}]$. The probability density is given by Eq. (1):

$$F(\mathcal{H}) = \begin{cases} \frac{1}{\mathcal{H}}, & \mathcal{H} \in [\mathcal{H}_{min}, \mathcal{H}_{max}], \\ 0, & \mathcal{H} \notin [\mathcal{H}_{min}, \mathcal{H}_{max}]. \end{cases} \tag{1}$$

When we consider the energy consumption, there are three major components in each period T, i.e., sensing, communicating and computing, which are respectively denoted as E_s, E_c and E_p.

2.4. Interference model

In this paper, we use the RTS/CTS Model [10,7]. We say a link L_{uv} interferes with L_{xy} in the RTS/CTS model if the nodes x or y are jammed by u or v. Based on the RTS/CTS model, we also construct a *conflict graph* [7,11], denoted as $\mathcal{C}_G$. Each link in the communication graph G is a vertex in $\mathcal{C}_G$. There is an edge connecting two vertices in $\mathcal{C}_G$ if and only if the two corresponding links interfere with each other.

3. Data aggregation

We use a data aggregation scheme to collect the data from all the nodes to save energy. A widely researched method for constructing a Data Aggregation Tree (DAT), denoted as $\mathcal{T}$, is to construct a Connected Dominating Set (CDS) [12,13]. Wan et al. found a Maximal Independent Set (MIS) before constructing a CDS [9]. Du et al. presented two efficient approximation algorithms for obtaining a minimum CDS while considering that the transmission ranges of all nodes are not necessarily equal [13]. Because the nodes in WSNs have limited energy, it is not sufficient to construct a CDS. Several prior works have constructed Weighted CDSs (WCDSs) by assuming that each node is initially assigned a weight [8,14].

In a solar sensor network, we aim to maximize network capacity while increasing energy efficiency. Those nodes harvesting more energy can afford to handle more traffic than those with less energy. Therefore, we use the harvested energy to determine the weight of each node and construct a Maximum-Weighted Connected Dominating Set (MaWCDS), which is equivalent to the Minimum-Weighted CDS (MiWCDS) problem. Constructing a MaWCDS is the first step in constructing an energy efficient data aggregation tree; the second step is to select dominatees.

3.1. Data aggregation scheme

Before describing the construction of the MaWCDS, we firstly present our data aggregation scheme. In this scheme, each node samples data in every time slot τ_i, and each child node then transmits its data to its corresponding parent node in $\mathcal{T}$. Each parent collects its own data and that of its child nodes, and then aggregates the data into another one in τ_j, $i \neq j$. The parent node transmits the aggregated data to its own parent, and this is repeated until the data is fully aggregated and transmitted to the sink. The whole process is illustrated in Algorithm 1.

Algorithm 1. Data aggregation scheme

```
Input: $\mathcal{T}_G$.    Output: Data aggregation Scheme.
1: Each dominatee $u$ senses and packs its data into a packet in
   each period $T$.
2: for each $u$ that is a leaf node do
3:    $u$ sends its packet to its parent directly.
4: end for
5: while a parent node $v$ receives packets do
6:    $v$ receives the packets from all its children, and packs
      them and its own data into one packet;
7:    $v$ transmits its packet to its parent $w$.
8: end while
```

3.2. Constructing the MaWDS

First, we select the dominators for the construct of the MaWDS $\mathcal{S}$. Here we use Algorithm 2 to determine the dominators based on the original network G. We define the set containing all of the dominatees as $\mathcal{D}_G$ and the k-hop neighborhood of node u as N_u^k, where k is a positive integer. Zou et al. presented a $(5+\epsilon)$-approximation algorithm to construct an MiWCDS [8]. We adopt their algorithm as a step in our algorithm to construct an MaWDS, denoted as $\mathcal{S}$.

Algorithm 2. Constructing the MaWDS (centralized)

```
Input: $G$.    Output: MaWDS $\mathcal{S}(G)$
1: Sort all nodes in $G$ into a decreasing order O according to
   their weights.
2: Mark all nodes as WHITE.
3: while O $\neq \emptyset$ do
4:    Mark the first WHITE node $u$ in O as YELLOW;
5:    $u$ colors its one-hop neighbors $w_i$ $(i = 1, \ldots, |N_u^1|)$ as
      GRAY;
6:    Delete $u$ and its neighbors $w_i$ $(i = 1, \ldots, |N_u^1|)$ from O;
7: end while
8: Sort all YELLOW nodes into a decreasing order O according
   to their weights.
9: while O $\neq \emptyset$ do
10:   The first node $u$ in O collects the weights and IDs of all of
      its two-hop neighbors $N_2(u)$;
11:   $u$ calculates the weight sum $SW_u$ of the YELLOW nodes
      in $N_2(u)$;
12:   Using the method in [8] to construct a maximum-
      weighted set cover, denoted as $WS_u$. $u$ selects a subset from
      $N_u^2$ to cover all the nodes in $N_u^1$ including $u$.
13:   if the weight of $WS_u$ is higher than $SW_u$ then
14:      The nodes in $WS_u$ are all marked as BLACK. The
         YELLOW nodes are marked as GRAY.
15:   end if
16: end while
```

In Algorithm 2, the total weight of the MaWDS is bounded by the following lemma.

Lemma 1. Algorithm 2 *constructs a dominating set with a total weight of not less than* $\frac{1}{18\eta(5+\epsilon)}$ *times of the optimum if the network is modeled by Unit Disk Graph (UDG), where* $\eta = \frac{\mathcal{H}_{max}}{\mathcal{H}_{min}}$ *and* $\varepsilon > 0$ *is a constant.*

Proof. We firstly calculate the total weight of all YELLOW nodes comprising an MIS before step 8 in Algorithm 2, and denote the MIS as $\mathcal{M}$. Suppose u is an arbitrary YELLOW node, i.e., $u \in \mathcal{M}$, and the optimal dominator set is $\mathcal{S}_{OPT}$. For each YELLOW node, there are at most 5 optimal dominators in its one-hop neighborhood. We denote the five optimal dominators $u_i' \in \mathcal{S}$, where $i = 1, \ldots, 5$.

W.l.o.g, the node with the minimal weight in the one-hop neighborhood of u is denoted as x, and we define $\eta_u = \frac{\mathcal{H}_u}{\mathcal{H}_x}$, where $\mathcal{H}_{u_i'} \leqslant \eta_u \mathcal{H}_u$. We then obtain Eq. (2) by summing the weights of all the nodes in $\mathcal{S}_{OPT}$.

$$\sum_{u \in \mathcal{S}} \eta_u \mathcal{H}_u \geqslant \sum_{u \in \mathcal{S}} \sum_{1 \leqslant i \leqslant 5} \mathcal{H}_{u_i'} \geqslant \sum_{u_i' \in \mathcal{S}_{OPT}} \mathcal{H}_{u_i'} = \mathcal{H}_{OPT}. \tag{2}$$

Furthermore, $\sum_{u \in \mathcal{S}} \eta_u \mathcal{H}_u \leqslant \sum_{u \in \mathcal{S}} \eta \mathcal{H}_u = \eta \mathcal{H}_{\mathcal{M}}$, where $\eta = \frac{\mathcal{H}_{max}}{\mathcal{H}_{min}}$. Therefore,

$$\mathcal{H}_{\mathcal{M}} \geqslant \frac{1}{\eta} \mathcal{H}_{OPT}. \tag{3}$$

In step 12, an existing algorithm [8] is used to construct a local dominating set $\mathcal{S}_{alg}(u)$ in u's neighborhood. We denote by $\mathcal{H}_u(alg)$ the weight of $\mathcal{S}_{alg}(u)$ constructed by Algorithm 2, and denote the local optimal weight as $\mathcal{H}_u(local)$. Thus, $\mathcal{H}_u(alg)$ is at least $\frac{1}{5+\epsilon}\mathcal{H}_u(local)$, where $\varepsilon > 0$ is a small constant. We denote by $\mathcal{S}_{OPT}(u)$ the subset of $\mathcal{S}_{OPT}$ that lies within the 2-hop neighborhood of u, i.e., $\mathcal{S}_{OPT}(u) = \mathcal{S}_{OPT} \cup N_2(u)$, and denote by $\mathcal{H}_{\mathcal{S}_{OPT}(u)}$ the weight of $\mathcal{S}_{OPT}(u)$. We then have $\mathcal{H}_{\mathcal{S}_{OPT}(u)} \geqslant \mathcal{H}_u(local)$ because $|\mathcal{S}_{OPT}(u)|$ is larger than the size of local optimal dominating set; in other words, some nodes in $\mathcal{S}_{OPT}(u)$ have higher weights than those in $\mathcal{S}_{alg}(u)$. Because there are at most 18 dominators in a node's 2-hop neighborhood and $\mathcal{H}_x \geqslant \eta \mathcal{H}_y$ if a node x in $\mathcal{S}_{OPT}(u)$ has higher weight than a node y in the local optimal dominator set, $\mathcal{H}_{\mathcal{S}_{OPT}(u)} \leqslant 18\eta\mathcal{H}_u(local)$. Thus $\mathcal{H}_u(alg) \geqslant \frac{1}{18\eta(5+\epsilon)}\mathcal{H}_{\mathcal{S}_{OPT}(u)}$. The total weight of the whole network then satisfies the following equation:

$$\sum_{u \in \mathcal{M}} \mathcal{H}_u(alg) \geqslant \frac{1}{18\eta(5+\epsilon)} \sum_{u \in \mathcal{S}_{OPT}} \mathcal{H}_{\mathcal{S}_{OPT}(u)}$$
$$\Rightarrow \mathcal{H}_{\mathcal{M}} \geqslant \frac{1}{18\eta(5+\epsilon)} \mathcal{H}_{OPT}.$$

Based on Eqs. (3) and (4), we obtain $\mathcal{H}_{\mathcal{M}} \geqslant \min\left\{\frac{1}{\delta}, \frac{1}{18\delta(5+\epsilon)}\right\}\mathcal{H}_{OPT} = \frac{1}{18\eta(5+\epsilon)}\mathcal{H}_{OPT}$. □

3.3. Choosing connectors

Based on $\mathcal{S}(G)$, we can construct an MaWCDS. The whole process for constructing the MaWCDS is shown in Algorithm 3. To connect the dominators in $\mathcal{S}(G)$, several *connectors* are needed, which can be found among the dominatees. The dominators and connectors form a CDS. Several previous methods have been proposed to find these connectors [15,13,8,9,16]. An algorithm has been proposed to connect the dominating set using a Steiner tree algorithm after the dominating set is firstly constructed [17]. This algorithm yields a performance factor of $H(\Delta) + 2$, where Δ is the maximal degree in the network and H is a harmonic function. The Steiner tree algorithm was used to connect an MIS after its construction. Another group designed a constant approximation algorithm to the Strongly Connected Dominating Set (SCDS) problem using Breadth First Search (BFS_SCDS) and its improvements, termed SCDS using a Minimum number of Steiner Nodes (MSN_SCDS) [18].

When constructing a CDS, instead of finding the shortest path between each pair of dominators u and v, our method finds the most energy-efficient path between them. For any pair of neighboring dominators u and v, between which there is at least one path within three hops in the original graph, suppose there are K paths between them. For each path $P(u,v)^i$, $i = 1,\ldots,K$, we select out a node w with the minimal weight in $P(u,v)^i$, and let $W(P(u,v)^i) = \min_{w \in P(u,v)} \mathcal{H}_w$. Then w's weight, $\mathcal{H}_w$, is used to denote the weight $W(P(u,v)^i)$ of $P(u,v)^i$. When there are K paths between any pair of neighboring dominators, we choose the path with the maximal path weight $\max_{i=1,\ldots,K} W(P(u,v)^i)$ for the two dominators. The process of the connector selection is implemented in Algorithm 3.

The algorithm uses the following message and process to construct a virtual graph.

Path$(u, hops, W, list)$ is a message broadcast by a dominator u. The message contains four parameters u, *hops*, W and *list*. u is the ID of the dominator u, which creates the message and is called the transmitter. *hops* and W respectively denote the number of hops and the path weight between u and a receiver. *list* records the relay nodes in the path.

The construction of a virtual graph (VirtG). When the dominating set $\mathcal{S}$ is obtained by Algorithm 2, we can construct a connected graph, called *VirtG* [15]. Here we provide a new way to denote the weight of a path among the dominators when we construct *VirtG*. We call two dominators *neighboring dominators* if there is a path between them containing no dominators.

Algorithm 3. The Construction of the CDS

```
Inputs: $G$ and $\mathcal{S}(G)$. Output: A CDS $\mathcal{F}_G$ and a connector set $\mathcal{N}_\mathcal{F}$.
1: Each dominator $u$ broadcasts a message
   $Path(u, hops = 0, W = \mathcal{H}_u, list = NULL)$ to its one-hop neighbors.
2: When a dominatee node $w$ receives a message Path, it checks four parameters contained in the message.
3: if $hops \leqslant 2$ then
4:     $w$ sets $hops$ += 1 and updates $W = \mathcal{H}_v$ if $W > \mathcal{H}_w$, and adds itself into list.
5:     $w$ broadcasts the message to its one-hop neighbors.
6: else
7:     $w$ discards the message.
8: end if
9: When a dominator $v$ receives a message Path, $v$ checks three parameters contained in the message.
10: $v$ checks whether it receives messages from the same transmitter.
11: if The message comes from the same transmitter then
12: $v$ stores the message with the highest path weight and discards other messages from the same transmitter $u$.
13: else
14:     $v$ stores the message.
15: end if
16: Link each pair of neighboring dominators $u$ and $v$ by a virtual edge $\widetilde{uv}$. The weight of the edge $\widetilde{uv}$ is set as
    $2\mathcal{H}_{max} - W(P(u,v)^i)$. All the dominators and virtual edges compose a virtual graph VirtG.
17: Construct a MST on VirtG. Denote the MST by $V_{MST}$.
18: Keep all dominators and the connectors on the paths, which are corresponding to the edges in $V_{MST}$. The CDS $\mathcal{F}_G$ is obtained.
19: The left connectors form a connector set denoted as $\mathcal{N}_\mathcal{F}$.
```

In Algorithm 3, we select the connectors with the highest possible weight. These connectors become the parents in the EDAT in Algorithm 1. In Subsection 3.5, we show that parents consume more energy than children nodes so we must consider the path weight.

Lemma 2. *For any pair of neighboring dominators u and v, the weight of each path $P(u,v)^i$ constructed in* Algorithm 3 *is maximized.*

The following lemma insures the subgraph $\mathcal{F}_G$ obtained by Algorithm 2 is a CDS.

Lemma 3. *By* Algorithm 3, *the subgraph $\mathcal{F}_G$ is a CDS and $\mathcal{N}_\mathcal{F}$ is a connector set corresponding to $\mathcal{F}_G$.*

Proof. We need only prove that there is at least one path between an arbitrary pair of neighboring dominators u and v in $\mathcal{F}_G$ if there are paths between them in the original graph G and there is at least one path containing at most two nodes in these paths. When there are paths containing no more than two nodes between u and v in G, there is at least one path left after running Algorithm 3 because the algorithm chooses the maximum-weighted path.

When there are paths containing more than two nodes between u and v in G, the following applies. Suppose one of the paths contains three nodes, w_1,w_2 and w_3, and is denoted as $u \leftrightarrow w_1 \leftrightarrow w_2 \leftrightarrow w_3 \leftrightarrow v$. If w_2 is not the neighbor of both u and v, then w_2 is a dominator or a neighbor of another dominator x according to Algorithm 2. In the first case, the other two paths, $u \leftrightarrow w_1 \leftrightarrow w_2$ and $w_2 \leftrightarrow w_3 \leftrightarrow v$, would be constructed by Algorithm 3. In the second case, other two paths, $u \leftrightarrow w_1 \leftrightarrow w_2 \leftrightarrow x$ and $x \leftrightarrow w_2 \leftrightarrow w_3 \leftrightarrow v$, would be constructed. We can obtain a similar proof when the path between u and v contains more than three nodes. $\square$

In WSNs, the computational capability of sensor nodes is relatively limited, and we are therefore concerning the time complexity of the algorithm. Wireless communication consumes a great deal of energy, and increased message communication causes longer delays due to bandwidth limitation. We therefore analyze the message of Algorithm 3.

Lemma 4. *The time complexity of* Algorithm 3 *is $O(n^2)$ in the worst case.*

Proof. Algorithm 3 costs at most $O(|\mathcal{S}|)$ at steps 2 and 10. Each dominator has at most ℓ_k dominators in its two-hop neighborhood, where $\ell_k \leqslant (2k+1)^2 - 1$ and k is the hop number [19]. It takes at most $O(\ell_k|\mathcal{S}|)$ time by step 16 because there are $|\mathcal{S}|$ dominators in total, and we note that $|\mathcal{S}| \leqslant n$. Therefore, the first 16 steps take at most $O(n)$ time except steps 2 and 10. Because the number of edges in *VirtG* is at most $O(n)$, it takes at most $O(n\lg n)$ time to construct an MST. Here we denote the number of edges in *VirtG* as $|E(VirtG)|$. There are at most two connectors between any pair of neighboring dominators. It is easy to confirm that there are at most $2|E(VirtG)|$ connectors. Before step 18, the connector set is determined, and $|E(VirtG)| \leqslant n^2$. Therefore, step 18 takes at most a time of $O(n^2)$ in the worst case. $\square$

Theorem 5. *If the geometric information is previously known for all nodes and the network is modeled by UDG, then* Algorithm 3 *uses at most $O(n^2)$ messages under worst case.*

Proof. There are only dominators sending out *Path* messages. Thus, the total number of messages is $O(|\mathcal{S}|)$. In Algorithm 3, a dominatee can only receive messages from the nodes in its 2-hop neighborhood. This ensures that each node receives the messages in its 2-hop neighborhood by using at most $O(n)$ messages if the geometric information of each node is previously known and the network is modeled by UDG [15]. Additionally, a dominator can only receive messages from the nodes in its 3-hop neighborhood. Therefore, a dominatee, which is 3-hop away from the transmitter, would discard the messages. In other words, only dominators that locate within 3-hop away from the transmitter can receive these messages. Notice that the number of dominator in the 3-hop neighborhood is at most ℓ_3 and each dominator has at most Δ neighbors. Therefore, the message complexity of Algorithm 3 is at most $O(n^2)$ in the worst case because $\Delta \leqslant n$ and ℓ_3 is a constant. □

3.4. Dominatee selection

Algorithm 4. The Selection of dominatees

```
Input: $G$ and $\mathcal{S}(G)$. Output: a connected graph $\mathcal{T}(G)$.
1: for Each dominator $u$ in $\mathcal{S}(G)$ do
2:    Comment: $u$ roots at itself to construct a tree $\mathbf{T}_u^3$ based on $N_u^3/\mathcal{N}_{\mathcal{F}}/\mathcal{S}$;
3:    Initially $\mathbf{T}_u^3 = \{u\}$;
4:    Each node sets its harvested energy $\mathcal{H}$ by $\mathcal{H} - mE_s$, i.e., $\mathcal{H} -= mE_s$.
5:    $u$ arranges its one-hop neighbors $w$ $(w \in N_v^1)$ into an nondecreasing order $L_v^1$ according to their weight.
6:    Each node $v$ in $\mathbf{T}_u^3$ sets a temporary variable $E_v$ and let $E_v = \mathcal{H}_v$.
7:    while $E_v > 0$ and $|L_u^1| > 0$ do
8:       For the first node $w$ in $L_u^1, E_v -= \mathcal{H}_w$;
9:       if $E_v < 0$ then
10:         $E_v += \mathcal{H}_w$; Break;
11:      end if
12:      if $w \in N_v^1$ then
13:         $v$ connects to $w$ and moves $w$ from $L_u^1$ to $\mathbf{T}_u^3$;
14:         $w$ disconnects the link with those nodes in $\mathbf{T}_u^3$.
15:      else
16:         $E_v += \mathcal{H}_w$;
17:      end if
18:   end while
19:   Delete $v$ from $\mathbf{T}_u^3$.
20:   if $\mathbf{T}_u^3$ is not empty then
21:      Go to step 6.
22:   end if
23: end for
```

Different from previous works, we arrange the nodes in $V/\mathcal{F}_G$ into subtrees according to their weights in order to construct an EDAT, instead of connecting these nodes with the dominators directly. Using Algorithm 4, we obtain a connected tree that is used as the EDAT $\mathcal{T}(G)$. Suppose there exists an optimal weighted EDAT $\mathcal{T}_{OPT}(G)$, in which the total weight difference between every parent and its children is minimized.

In Algorithm 4, we denote the total weight difference between every parent and its children by $\mathcal{H}_{\mathcal{T}}$. Suppose the optimal total weight difference of $\mathcal{T}_{OPT}(G)$ is denoted as $\mathcal{H}_{OPT}$. We then obtain the following theorem.

Theorem 6. *$\mathcal{H}_{\mathcal{T}} - \mathcal{H}_{OPT}$ is no more than $\frac{5\widetilde{\mathcal{H}}|\mathcal{S}|^2}{n-1}$ for the networks modeled by a UDG, where $\widetilde{\mathcal{H}} = |[\mathcal{H}_{min}, \mathcal{H}_{max}]|$.*

Proof. Each a parent u has at most $|N_u^1|$ possible children. In Algorithm 4, u arranges its one-hop neighbors z_i $(z_i \in N_u^1, i = 1, \ldots, |N_u^1|)$ into an nondecreasing order L_u^1. Suppose u selects a subset $S_{OPT}(u)$ from its one-hop neighborhood, i.e., $S_{OPT}(u) \subseteq N_u^1$, in $\widetilde{\mathcal{F}}_{OPT}(G)$, and selects a subset $S'(u)$ in Algorithm 4. Then we obtain that $\sum_{v \in S_{OPT}(u)} \mathcal{H}_v \leqslant \mathcal{H}_u$, $\sum_{v \in S'(u)} \mathcal{H}_v \leqslant \mathcal{H}_u$ because $\mathcal{H}_u$ is the biggest among the undisposed nodes in N_u^1 according to Algorithm 2. Thus $\sum_{v \in S'(u)} \mathcal{H}_v \leqslant \sum_{v \in S_{OPT}(u)} \mathcal{H}_v$. Because u selects its children from the header of the order L_u^1 to its tail, there must be at least one node in $S_{OPT}(u)$; otherwise, u is not a parent but a leaf node.

Because $\sum_{v \in S_{OPT}(u)} \mathcal{H}_v \leqslant \mathcal{H}_u$ and $\sum_{v \in S'(u)} \mathcal{H}_v \geqslant \mathcal{H}_x$, where x is a node with the smallest weight in L_u^1, we have $\sum_{v \in S_{OPT}(u)} \mathcal{H}_v - \sum_{v \in S'(u)} \mathcal{H}_v \leqslant \mathcal{H}_u - \mathcal{H}_x$. Because the maximal degree is Δ, there are at most Δ parents in u's one-hop neighborhood. In the one-hop neighborhood, the total weight difference between all parents and their children satisfies $\Delta\left(\sum_{v \in S_{OPT}(u)} \mathcal{H}_v - \sum_{v \in S'(u)} \mathcal{H}_v\right) \leqslant \Delta(\mathcal{H}_u - \mathcal{H}_x)$. In the whole network, $\sum_{u \in \mathcal{S}} \Delta\left(\sum_{v \in S_{OPT}(u)} \mathcal{H}_v - \sum_{v \in S'(u)} \mathcal{H}_v\right) \leqslant \sum_{u \in \mathcal{S}} \Delta(\mathcal{H}_u - \mathcal{H}_x)$. If we arrange all nodes (in V) into an nondecreasing list L_G^1 of their weight and denote the nodes in L_G^1 by $y_i, i = 1, \ldots, |V|$, we then have Eq. (5):

$$\begin{aligned}\sum_{u \in \mathcal{S}} \Delta(\mathcal{H}_u - \mathcal{H}_x) &= \Delta\left(\sum_{u \in \mathcal{S}} \mathcal{H}_u - \sum_{u \in \mathcal{S}} \mathcal{H}_x\right) \\ &\leqslant \Delta\left(\sum_{i=1}^{|\mathcal{S}|} \mathcal{H}_{y_i} - \sum_{j=|V|-|\mathcal{S}|}^{|V|} \mathcal{H}_{y_j}\right) \\ &= \Delta \sum_{i=1}^{|\mathcal{S}|} (\mathcal{H}_{y_i} - \mathcal{H}_{y_{|V|-|\mathcal{S}|+i}}). \end{aligned} \tag{5}$$

Because the energy harvested by n nodes uniformly and randomly distributes in the interval $[\mathcal{H}_{min}, \mathcal{H}_{max}]$, as shown in Fig. 1. When the interval $[\mathcal{H}_{min}, \mathcal{H}_{max}]$ is equally divided into $n-1$ subintervals, which are respectively denoted as $0, \ldots, n-1$ in Fig. 1.

The length of each subinterval is then $D_{\mathcal{H}} = \frac{\mathcal{H}_{max} - \mathcal{H}_{min}}{n-1}$. For any pair of nodes, such as x and u, we denote the "distance" from x to u by $d(x,u)$. We firstly unify the weight of an arbitrary node u using the equation $\mathcal{H}'_u = \frac{\mathcal{H}_u - \mathcal{H}_{min}}{\mathcal{H}_{max} - \mathcal{H}_{min}}$, where $\mathcal{H}'_u$ is the unified weighted of the node u. Then $\mathcal{H}'_u \in [0, 1]$. Therefore, the probability $P(n, d'(x,u))$ that there are no pairs of nodes, between which the unified distance is less than the unified length $D'_{\mathcal{H}}$ of each subinterval, i.e., $d'(x,u) \leqslant D'_{\mathcal{H}}$, is [20]:

$$P(n, d'(x,u)) = [1 - (n-1)d']^n.$$

When the network is very large, i.e., $n \to \infty$, we can obtain that $P(n, d'(x,u)) \to 0$ because $(n-1)d' \leqslant 1$. Therefore, $\sum_{i=1}^{|\mathcal{S}|} \mathcal{H}_{y_i} \leqslant |\mathcal{S}|(\mathcal{H}_{max} - \sum_{i=0}^{|\mathcal{S}|} \frac{i}{n-1}(\mathcal{H}_{max} - \mathcal{H}_{min})$. Similarly, we can also obtain the inequality $\sum_{i=1}^{|\mathcal{S}|} \mathcal{H}_{y_{|V|-|\mathcal{S}|+i}} \geqslant |\mathcal{S}|(\mathcal{H}_{min} + \sum_{i=0}^{|\mathcal{S}|} \frac{i}{n-1}(\mathcal{H}_{max} - \mathcal{H}_{min})$. Insert the two inequalities into Eq. (5) and obtain the following equation:

$$\begin{aligned}\sum_{u \in \mathcal{S}} \Delta(\mathcal{H}_u - \mathcal{H}_x) &\leqslant \Delta\left\{|\mathcal{S}|(\mathcal{H}_{max} - \mathcal{H}_{min}) - 2\sum_{i=0}^{|\mathcal{S}|} \frac{i}{n-1}(\mathcal{H}_{max} - \mathcal{H}_{min})\right\} \\ &= \Delta\widetilde{\mathcal{H}}\left(|\mathcal{S}| - 2\sum_{i=0}^{|\mathcal{S}|} \frac{i}{n-1}\right) = \Delta\widetilde{\mathcal{H}}|\mathcal{S}|\left(1 - \frac{|\mathcal{S}|+1}{n-1}\right) \\ &= \Delta\widetilde{\mathcal{H}}|\mathcal{S}|\frac{n - |\mathcal{S}| - 2}{n-1} \leqslant \frac{5\Delta\widetilde{\mathcal{H}}|\mathcal{S}|^2}{n-1}. \end{aligned} \tag{6}$$

In Eq. (6), the last inequality is because of Lemma 7.2 in [19]. Therefore, we can obtain that $\sum_{u \in \mathcal{S}} \Delta\left(\sum_{v \in S_{OPT}(u)} \mathcal{H}_v - \sum_{v \in S'(u)} \mathcal{H}_v\right) \leqslant \frac{5\Delta\widetilde{\mathcal{H}}|\mathcal{S}|^2}{n-1}$, i.e., $\sum_{u \in \mathcal{S}}\left(\sum_{v \in S_{OPT}(u)} \mathcal{H}_v - \sum_{v \in S'(u)} \mathcal{H}_v\right) \leqslant \frac{5\widetilde{\mathcal{H}}|\mathcal{S}|^2}{n-1}$. Since $\mathcal{H}_{\widetilde{\mathcal{F}}} - \mathcal{H}_{OPT} =$

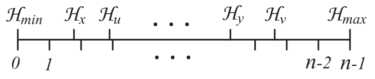

Fig. 1. Node weight distribution.

$\sum_{u\in\mathcal{S}}[(\mathcal{H}_u - \sum_{v\in S'(u)}\mathcal{H}_v) - (\mathcal{H}_u - \sum_{v\in S_{OPT}(u)}\mathcal{H}_v)] = \sum_{u\in\mathcal{S}}\left(\sum_{v\in S_{OPT}(u)}\mathcal{H}_v - \sum_{v\in S'(u)}\mathcal{H}_v\right)$, this finishes proof. □

We guarantee that the EDAT obtained in Algorithm 4 is a connected tree.

Theorem 7. *By* Algorithm 4, *the obtained subgraph $\widetilde{\mathcal{F}}(G)$ is connected if the given network G is connected.*

Proof. If the given network G is connected, we can obtain a CDS according to Lemma 3. Therefore, we need only prove that any dominatee connects with at least one dominator.

Because any dominator u must have at least one another dominator in N_u^3 in G, there must be at least one dominator in the 2-hop neighborhood of any dominatee. According to Algorithm 3, all nodes are connected in $T_{mst}(G)$. The possibility that a link in $T_{mst}(G)$ will be deleted is addressed in Algorithm 4. There are two cases: (1) N_u^3 is originally connected in $T_{mst}(G)$ before u runs from Step 5; and (2) N_u^3 is originally disconnected in $T_{mst}(G)$.

Case (1): In Algorithm 4, any dominator u constructs a tree $\mathbf{T}_u^3$; and the only chance that a link may be deleted lies in Step 14. After Step 14, N_u^3 is divided into two components: $\mathbf{T}_u^3$ and $N_u^3/\mathbf{T}_u^3$. A node must find a parent in $\mathbf{T}_u^3$ when it joins in $\mathbf{T}_u^3$. The node then connects to its parent, so $\mathbf{T}_u^3$ is connected. Note that the links between the nodes in $\mathbf{T}_u^3$ and those in $N_u^3/\mathbf{T}_u^3$ are not disconnected. Thus, $\mathbf{T}_u^3$ remains connected with $N_u^3/\mathbf{T}_u^3$. In the remaining component, $N_u^3/\mathbf{T}_u^3$, no single node and link are deleted, so it also remains connected. Therefore, all nodes inN_u^3 remains connected after running Algorithm 4 in this case.

Case (2): Suppose that N_u^3 contains k disconnected components D_k, $k = 1,2,\ldots$. *W.l.o.g*, u belongs to the component D_1. As in *Case* (1), D_1 is still connected. Similarly, other D_k, $k = 2,\ldots$ can connect with other dominators. Because CDS is connected, Algorithm 4 would not also disconnect $T_{mst(G)}$. This finishes the proof. □

Here, we also give the time complexity of Algorithm 4.

Lemma 8. *The time complexity of* Algorithm 4 *is O(nlogn) in the worst case.*

Proof. In step 5 of Algorithm 4, each parent can use the Quicksort Method to arrange its one-hop neighbors according to their weights. The time consumed is at most $\Theta(\Delta\log\Delta)$ in the worst case because each node has at most Δ one-hop neighbors. There is a WHILE loop from steps 7 to 18. Because $Ł_u^1 \leqslant \Delta$, there are at most 6Δ steps. If the condition in the step 20 is satisfied, the step 21 will be implemented. Because the number of dominators is constant in 3-hop neighborhood of a node, the time consumed for step 21 is $O(\Delta\log\Delta)$. And $\Delta \leqslant n$. Therefore the time complexity of Algorithm 4 is $O(n\log n)$ in the worst case. □

3.5. Energy consumption

The energy consumption model is given in the Subsection 2.3. Data sensing, transmitting and aggregating are implemented according to Algorithm 1. In the following context, we consider the effect of our scheme on energy consumption.

Lemma 9. *The energy consumed for data sensing is a constant in every work period $T_e(G)$ in the whole network.*

It is obvious that the total energy consumed by the whole network when sampling data from the surrounding is $(n-1)E_s$ in each period $T_e(G)$ because each node samples data once except the sink. The communication energy E_c contains two components: receiving E_r and transmitting E_t. For each parent, $E_c = E_r + E_t$ in each period. Note that the E_r of a parent denotes the energy cost for a parent to receive a data packet from one of its children in the EDAT. Each leaf node also spends energy on receiving messages when it receives the control messages *CTS* and *ACK* under the RTS/CTS model. Here, we count the energy consumed for receiving *CTS* and *ACK* in its E_t because *CTS* and *ACK* are induced by the *DATA* message. Thus, $E_r = 0$ for each leaf node. Therefore, the E_r of a parent is dependent on the number of communications with its children in each period. Because the total number of children is $n-1$ in the whole network, we have Lemma 10.

Lemma 10. *The total energy consumed for the data reception and transmission is a constant in every period $T_e(G)$ in the whole network.*

After a parent receives data from all of its children in each period, it aggregates them into one packet, which consumes energy E_p. Thus, the total energy consumed for data aggregation is dependent on the total number of the parents in the whole network.

Lemma 11. *The energy consumed for data aggregation is at most $O\left(|\mathcal{S}|E_c\left(3+\min\left\{\Delta, \left\lfloor\frac{\widetilde{\mathcal{H}}}{2\mathcal{H}_{min}}\right\rfloor\right\}\right)\right)$ in every period T_e in the whole network.*

Proof. The construction of the dominator set $\mathcal{S}$ was described above. The dominators and their connectors are all parents in the data aggregation tree. Because there are at most two connectors between each pair of dominators, there are at most $2|\mathcal{S}|$ connectors. Therefore, the CDS forms a tree that contains at most $3|\mathcal{S}|$ parents.

In the $\mathcal{T}_G$ obtained by Algorithm 4, the number of levels of a local tree, which roots at a dominator u, is at most $\left|N_u^1\right|$. For an arbitrary node u with weight $\mathcal{H}_u$, the node can have at most $\left\lfloor\frac{\mathcal{H}_u}{\mathcal{H}_{min}}\right\rfloor$ children with a probability of $\left(\frac{\pi}{c_1^2}\right)^{\left\lfloor\frac{\mathcal{H}_u}{\mathcal{H}_{min}}\right\rfloor}$, where $c_1 = \frac{\mathbf{c}}{\mathfrak{R}}$ and is a given constant. Thus, the expected number of children for an arbitrary node is $\left\lfloor\frac{\widetilde{\mathcal{H}}}{2\mathcal{H}_{min}}\right\rfloor$ with a probability of at least $\left(\frac{\pi}{c_1^2}\right)^{\lfloor\delta\rfloor}$, and an arbitrary node has at most $\min\left\{\Delta, \left\lfloor\frac{\widetilde{\mathcal{H}}}{2\mathcal{H}_{min}}\right\rfloor\right\}$ neighbors.

We construct a virtual local tree (as shown in Fig. 2(b)) based on the original local tree (as shown in Fig. 2(a)), and cluster the nodes into lines. For example, we arrange the nodes x, and y in Fig. 2(a) into a "line" shown in Fig. 2(b). Note that a parent node is still a parent after all the nodes are arranged, and similarly a leaf node is still a leaf. There is then at least one leaf at the end of each "line", such as z in Fig. 2. We can show that there are at least

$$\left\lceil\frac{\min\left\{\Delta-1, \left\lfloor\frac{\widetilde{\mathcal{H}}}{2\mathcal{H}_{min}}\right\rfloor\right\}}{|N_u^1|}\right\rceil,$$

"lines", denoted as l_n, so there are at least l_n leaves. The number of parents in a local tree is at most

$$\min\left\{\Delta-1,\left\lfloor\frac{\widetilde{\mathcal{H}}}{2\mathcal{H}_{min}}\right\rfloor\right\}-l_n.$$

Since there are $|\mathcal{S}|$ local subtrees according to Algorithm 4, the total energy consumed $E_c(\mathcal{T}_G)$ on data aggregation in the whole network can be obtained from the following equation.

$$\begin{aligned}E_c(\mathcal{T}_G) &\leqslant \left(3|\mathcal{S}|+|\mathcal{S}|\left(\min\left\{\Delta-1,\left\lfloor\frac{\widetilde{\mathcal{H}}}{2\mathcal{H}_{min}}\right\rfloor\right\}-l_n\right)\right)E_c\\ &\leqslant |\mathcal{S}|\left(3+\min\left\{\Delta-1,\left\lfloor\frac{\widetilde{\mathcal{H}}}{2\mathcal{H}_{min}}\right\rfloor\right\}\left(1-\frac{1}{\delta}\right)\right)E_c\\ &\leqslant O\left(|\mathcal{S}|E_c\left(3+\min\left\{\Delta-1,\left\lfloor\frac{\widetilde{\mathcal{H}}}{2\mathcal{H}_{min}}\right\rfloor\right\}\right)\right)\\ &< O\left(nE_c\left(3+\min\left\{\Delta-1,\left\lfloor\frac{\widetilde{\mathcal{H}}}{2\mathcal{H}_{min}}\right\rfloor\right\}\right)\right). \qquad (7)\end{aligned}$$

This finishes proof. □

4. Network life

We can use the link-scheduling algorithm to estimate the number of time slots needed to schedule all links in the network. Fast distributed algorithms were proposed in a prior study, requiring at most $O(min(\log n, \log\psi))$ time slots for the RTS/CTS interference model [7]. Wan et al. designed a minimum-latency aggregation scheduling [21]. Xu et al. designed a scheduling partition method to implement a localized link scheduling so as to obtain the optimal capacity with the localized method [22].

In this section, we describe our design for a new link-scheduling technique using a CDS, firstly constructing a conflict graph $\mathcal{C}$ based on the previously constructed EDAT $\mathcal{T}(G)$. The link-scheduling algorithm is described in Algorithm 5. Based on the algorithm, we determine the number of time slots needed to schedule all the links in the whole network for each period T_e. We then give the necessary condition under which the network can work consecutively in periods.

Algorithm 5. Centralized Scheduling in the RTS/CTS Model

```
Input: The data aggregation tree $\mathcal{T}(G)$. Output: An
    interference-free scheduling.
1: Construct a conflict graph $\mathcal{C}$ and let $G_c = \mathcal{C}$.
2: while $G_c$ is not empty do
3:    Among the WHITE nodes, find the Minimum-Weighted
      DS $\mathcal{S}_1$, where the weight is the smallest total degree in $G_c$;
4:    Construct a VirtG based on $\mathcal{S}_1$ according to the method in
      Algorithm 3, and find the minimum weighted DS $\mathcal{S}_2$;
5:    Deletes all nodes in $\mathcal{S}_2$ and links connected with the
      nodes in $\mathcal{S}_2$ from $G_c$;
6:    Let $L_k$ denote the $|\mathcal{S}_2|$ nodes in $\mathcal{S}_2$;
7:    Process links $L_k$ from $k = 1, 2, \ldots, \widehat{L}$ and assign to $|L_k|$
      nodes in $L_k$ the earliest time slot not yet assigned to any of
      its neighbors;
8: end while
```

4.1. Link scheduling

If a transmission is active from node u to another node v on a link edge L_{uv}, we call u (or v) as the *head* (or *end*) of the link L_{uv}. We define the link set interfered by an active link L_{uv} by $I_{u,v}$.

Theorem 12. *In the RTS/CTS model, the scheduling designed in* Algorithm 5 *is free of interference.*

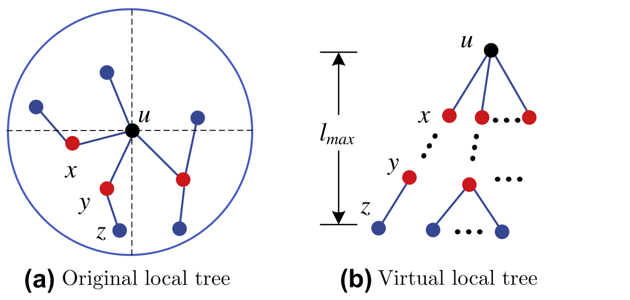

Fig. 2. Local tree level.

Proof. In the RTS/CTS model, the interference on an arbitrary link L_{xy} is $N_x^1 \cup N_y^1$ in the original graph G. When a link L_{xy} is active, there are at most three hops between the nodes (u and v) and the head or end of the link L_{xy}. Thus, any two nodes will not interfere with each other if they are more than 4 hops apart.

According to the definition of a DS, any pair of dominators are at least two hops apart so the dominator nodes in $\mathcal{S}_1$ are at least 2 hops apart. Note that $\mathcal{S}_1$ is obtained from the conflict graph $\mathcal{C}$. Thus, any pair of dominators are at least 4 hops apart in $\mathcal{S}_2$. □

Note $\widehat{L}$ denotes the total number of the WHILE cycle in Algorithm 5.

Theorem 13. *In the RTS/CTS model, the maximum number of time slots in the link scheduling is $2l_k\varrho^2$, and the minimum number of time slots is $4\delta-1$ in* Algorithm 5, *i.e., $4\delta-1 \leqslant \widehat{L} \leqslant 2l_k\varrho^2$, where $\varrho = \max\left\{\Delta,\left\lfloor\frac{\mathcal{H}_{max}}{\mathcal{H}_{min}}\right\rfloor\right\}$.*

Proof. First, we derive the minimum delay.

In the data aggregation tree $\mathcal{T}(G)$, we denote the degree of an arbitrary node x as d_x. Suppose r is one of its neighbor as shown in Fig. 3. The edge L_{xr} is then a node in the conflict graph $\mathcal{C}$. Under RTS/CTS, there are $d_x + d_r - 1$ nodes in L_{xr}'s interference region. Because the $d_x + d_r - 1$ nodes form a subtree of $\mathcal{T}(G)$, there are $d_x + d_r - 2$ edges in the sub-tree. Note that the $d_x + d_r - 2$ edges corresponds to the $d_x + d_r - 2$ points in $\mathcal{C}$. When the point L_{xr} is active in a given time slot, there are another $d_x + d_r - 2$ points that cannot be active in the same time slot with L_{xr}.

In Algorithm 5, $\mathcal{S}_2$ is a DS of the virtual graph based on DS_1, and the points in the corresponding DS_2 with the same L_k are assigned the same time slot. There is at least one connector between each pair of dominator nodes. Therefore, there are at least three points between each pair of active points in $\mathcal{S}_2$. For example, the points L_{xr}, L_{ys} and L_{zt} can form a DS in $\mathcal{S}_1$. The points L_{xr} and L_{tz} can then form a DS in $\mathcal{S}_2$. In each time slot, the point L_{ys} cannot be active when L_{xr} or L_{zt} are active because L_{ys} is a neighbor of L_{xr} or L_{zt} in $\mathcal{S}_1$. A dominator in $\mathcal{S}_1$ has at least one neighbor and the nodes r, x, s, y, t and z are connected. Furthermore, the points L_{xr} and L_{xs} cannot be active when L_{sy} is active. Therefore, there are another $d_x + d_r + d_s + d_y - 2$ nodes that cannot be active when L_{sy} is active.

Because the degree of a parent is at least δ, the minimal degree of the whole network, i.e., $d_v \geqslant \delta, \mathcal{C} - 2 \geqslant 4\delta - 2$. Therefore, there are at least $4\delta - 2$ nodes that cannot be active when an arbitrary point L_{sy} is active. In other words, the $4\delta - 2$ nodes cannot locate within the same $\mathcal{S}_2$ with L_{sy} at the same time L_k. Because each

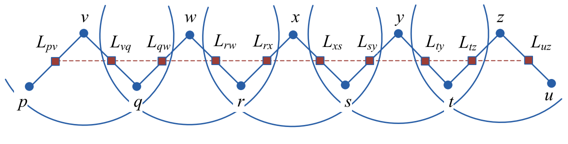

Fig. 3. A conflict graph.

active point renders another $4\delta - 2$ nodes inactive in each time slot, there are at least $4\delta - 2 + 1 = 4\delta - 1$ time slots required to finish activating all the points, i.e., $\widehat{L} \geqslant 4\delta - 1$.

Second, we derive the maximum delay.

There are at most two connectors between each pair of dominators. Suppose the points L_{pv}, L_{rw}, L_{xy} and L_{uz} are dominators in the conflict graph $\mathcal{C}_G$ in Algorithm 5. They are three hops away from each other and form a DS $\mathcal{S}_1$. *W.l.o.g*, there are two dominators L_{pv} and L_{uz} that are at most three hops away.

For an arbitrary link L_{uv}, the *heads* or *ends* of the links in the interference set $I_{u,v}$ are contained in $N_v^2 \cup N_u^2$. Because each node u has at most $\max\left\{\Delta, \left\lfloor\frac{\mathcal{H}_{max}}{\mathcal{H}_{min}}\right\rfloor\right\}$ node in its one-hop neighborhood $N^1(u)$, i.e., $|N^1(u)| \leqslant \max\left\{\Delta, \left\lfloor\frac{\mathcal{H}_{max}}{\mathcal{H}_{min}}\right\rfloor\right\}$, each dominator in $\mathcal{S}_1$ has at most $2\max^2\left\{\Delta, \left\lfloor\frac{\mathcal{H}_{max}}{\mathcal{H}_{min}}\right\rfloor\right\}$ neighbors. If two links can be active simultaneously, the distance between their heads or ends must be at least $\Re$. Therefore, there are at most l_k dominators (see Lemma 7.3 in [19]) in a disk with a radius of k-units centered on any dominator. Therefore, there are at most $2l_k\max^2\left\{\Delta, \left\lfloor\frac{\mathcal{H}_{max}}{\mathcal{H}_{min}}\right\rfloor\right\}$ links in a disk with a radius of k-units centered on a dominator in $\mathcal{S}_2$. Because there is at least one active point in a dominator's 9 hops range in each WHILE cycle in Algorithm 5, there are at most $2l_k\varrho^2$ cycles, where $k = 9$ and $\varrho = \max\left\{\Delta, \left\lfloor\frac{\mathcal{H}_{max}}{\mathcal{H}_{min}}\right\rfloor\right\}$. □

4.2. Sufficient condition for consecutive operation

For an arbitrary node u, its harvested energy is $\mathcal{H}_u$ in each work period $T_e(u)$ and the energy consumed by u is $c_T(u) = E_s + E_c + \beta E_p$ in each time slot τ, where $\beta = 0,1$. When a node is leaf node, $\beta = 0$. Otherwise, $\beta = 1$. Therefore, u can operate in $\frac{\mathcal{H}_u}{c_T(u)}$ time slots. Only if $T_e(u) \geqslant T$ can node u work consecutively in periods.

We next compute the necessary number of time slots in which the harvested energy can support the work of a node for the whole period T. Therefor, we first analyze the length of the working period $T_e(G)$.

Theorem 14. *The most efficient working period $T_e(G)$ of the network is determined by the node satisfying the condition $\min_{v\in\mathcal{F},u\in V/\mathcal{F}}\left\{\frac{\mathcal{H}_v}{d_v}, \frac{\mathcal{H}_u}{\varepsilon}\right\}$, where $\varepsilon = \frac{E_s+E_t+E_p}{E_r}$.*

Proof. The nodes in the whole network can be divided into two classes: leaf nodes and parents. The energy consumed by each leaf node u for sensing and transmitting data is E_{son}. Therefore, the total energy consumed by a leaf node is $E_{son} = E_s + E_t$ in each τ. The energy consumed by each parent node v for sensing, receiving, data aggregating and transmitting is E_{parent}. Therefore, the total energy consumed by each parent node is $E_{parent} = E_s + d_vE_r + E_t + E_p$ in each τ, where d_v is the number of children in the data aggregation tree $\mathcal{T}_G$. It is easy to demonstrate that the leaf node with the minimum harvested energy is the first one among all the leaf nodes to use up all of its energy. Suppose there are two parents v and w. If $\frac{\mathcal{H}_v}{d_v} \leqslant \frac{\mathcal{H}_w}{d_w}$, then we have:

$$\frac{T_e(v)}{T_e(w)} = \frac{\frac{\mathcal{H}_v}{E^v_{parent}}}{\frac{\mathcal{H}_w}{E^w_{parent}}} = \frac{\frac{\mathcal{H}_v}{E_s+d_vE_r+E_t+E_p}}{\frac{\mathcal{H}_w}{E_s+d_wE_r+E_t+E_p}} = \frac{\frac{\mathcal{H}_v}{d_vE_r+\varepsilon E_r}}{\frac{\mathcal{H}_w}{d_wE_r+\varepsilon E_r}} = \frac{\frac{\mathcal{H}_v}{d_v+\varepsilon}}{\frac{\mathcal{H}_w}{d_w+\varepsilon}} = \frac{\mathcal{H}_v}{\mathcal{H}_w}\frac{d_w+\varepsilon}{d_v+\varepsilon} \leqslant \frac{\mathcal{H}_v}{\mathcal{H}_w}\frac{d_w}{d_v} \leqslant 1,$$

where $\varepsilon = \frac{E_s+E_t+E_p}{E_r}$. Generally, $E_t > E_r$, so $\varepsilon > 1$. Between a leaf node and a parent node, if $\frac{\mathcal{H}_v}{d_v} \leqslant \frac{\mathcal{H}_u}{\varepsilon}$, we then have:

$$\frac{T_e(v)}{T_e(u)} = \frac{\frac{\mathcal{H}_v}{E^v_{parent}}}{\frac{\mathcal{H}_u}{E^u_{son}}} = \frac{\frac{\mathcal{H}_v}{E_s+d_vE_r+E_t+E_p}}{\frac{\mathcal{H}_u}{E_s+E_t}} \leqslant \frac{\frac{\mathcal{H}_v}{E_s+d_vE_r+E_t+E_p}}{\frac{\mathcal{H}_u}{E_s+E_t+E_p}} = \frac{\frac{\mathcal{H}_v}{d_vE_r+\varepsilon E_r}}{\frac{\mathcal{H}_u}{\varepsilon E_r}} = \frac{\frac{\mathcal{H}_v}{d_v+\varepsilon}}{\frac{\mathcal{H}_u}{\varepsilon}} = \frac{\mathcal{H}_v}{\mathcal{H}_u}\frac{\varepsilon}{d_v+\varepsilon} \leqslant \frac{\mathcal{H}_v}{\mathcal{H}_u}\frac{\varepsilon}{d_v} \leqslant 1.$$

We choose a parent node v such that it has the minimum value of $\frac{\mathcal{H}_v}{d_v}$ among all parent nodes and a leaf node u such that it has minimum value $\frac{\mathcal{H}_u}{\varepsilon}$ among all leaf nodes. The most efficient work period $T_e(G)$ is thus $\min_{v\in\mathcal{F},u\in V/\mathcal{F}}\left\{\frac{\mathcal{H}_v}{d_v}, \frac{\mathcal{H}_u}{\varepsilon}\right\}$. Therefore, the leaf node can work for at least $\frac{\mathcal{H}_{min}}{E_s+E_t}$ time slots and the parent node can work for at least $\frac{\mathcal{H}_{min}}{E_s+d_vE_r+E_t+E_p}$. □

According to the above theorem, the node u with the highest value of $\frac{\mathcal{H}_u}{d_u}$ or $\frac{\mathcal{H}_u}{\varepsilon}$ will survive the longest. The most efficient work period is $T_e(G) = \min_{v\in\mathcal{F},u\in V/\mathcal{F}}\left\{\frac{\mathcal{H}_v}{d_v}, \frac{\mathcal{H}_u}{\varepsilon}\right\}$. According to Theorem 13, $4\delta - 1 \leqslant \widehat{L} \leqslant 2l_k\varrho^2$. Let K be a positive integer; then, it is necessary to operate the network consecutively, period by period, by satisfying $T_e(G) \geqslant T = K\widehat{L}$ and $K = 1$. It is easy to show that $T_e(G) \geqslant \frac{\mathcal{H}_{min}}{\Delta}$ because $\varepsilon \leqslant \Delta$. Therefore, we can obtain the following lemma.

Lemma 15. *The necessary condition for the network to operate consecutively is $\mathcal{H}_{min} \geqslant \Delta(4\delta - 1)$.*

5. Related work

Sufficient energy sources and high energy efficiency are two significant factors in prolonging network life. Several technologies such as solar sensor node [4], WCDS [8,14] and link scheduling [10,7] [22] have been developed to achieve high network performance with respect to these two factors.

5.1. Solar sensor node

Micro-solar systems in WSNs have become a active research topic in recent years, and several solar systems have been developed [1,23–26,3]. Popular systems include the Heliomote [23] solar system, Trio [24], AmbiMax [25], PUMA [26] and Prometheus [3].

Taneja et al.. [1] designed a micro-solar power subsystem for climate-monitoring nodes, called HydroWatch, to study the hydrological cycles in forest watersheds. The authors created a model for each of the constituent components, and then designed a solar-energy harvesting module based on the energy budget predicted by an astronomical model for incident sunlight at the deployment location, a deep forest.

A storage service, called Solar-Store, was developed for solar-powered storage-centric sensor networks [27]. This service maximizes the retrievable data in the face of node failures by implementing reliable storage, where reliability is achieved by redundancy. Based on the Solar-Store system, a later refinement utilizes the energy surplus, harvested by solar sensors, to adaptively adjust the redundancy level of erasure codes used in communication, so that the delivery reliability is improved while the network life is still conserved [28]. Several related works have developed systems for cases in which WSNs can harvest energy, such as energy sustainable support of routing algorithms [29].

5.2. WCDS

Previous researchers have designed methods to minimize the size of the backbone or *dominating set* (DS), whereas others have tried to minimize the weight of the constructed backbone or DS. The DS is used in many areas, and it has been proposed to reduce of eliminate the communication overhead of broadcasting tasks by applying the localized dominating sets [30]. Constructing a DS is usually the first step for constructing a CDS.

In wireless sensor and ad hoc networks, CDS is the virtual backbone, playing an important role in routing, data aggregation and other functions [31]. The CDS problem was studied in UDG [32]. Many methods have aimed to establish a minimum connected dominating set (MCDS) based on a given connected network [32,13,16,33–35,30]. However, it is well known that finding an MCDS in UDG is an NP-hard problem [32]. Therefore, to achieve polynomial-time solutions, only methods using distributed approximation algorithms are practical for wireless ad hoc networks [33]. Das et al. proposed a distributed approximation algorithm for MCDS [34] that has an $\Theta(\log n)$ approximation factor. Stojmenovic et al. [30] and Wu and Li [35] designed distributive algorithms using an $\Theta(n)$ approximation factor. Wan et al. further designed an algorithm to construct a more size-small CDS with a constant approximation factor of at most 8, $O(n)$ time complexity and $O(n\log n)$ message complexity [33]. In practice, the transmission range of all nodes are not necessarily equal. Du et al. modeled a network as a disk graph, and introduced the CDS problem in disk graphs with bidirectional links (DGB) [13]. The authors presented two constant approximation algorithms for the CDS problem in DGB with a bounded transmission range ratio.

The construction problem of MWCDS is a generalization of the MCDS which each node assigned a weight by a mapping $f : V \to R$. For UDG, Clark et al. showed that the MDS problem is NP-hard [32]. For weighted UDG (WUDG), Ambühl et al. presented the first constant-approximation algorithm for MWDS in WUDG [36]. Huang et al. improved their approximation ratio from 72 to $6+\epsilon$ and presented a $10+\epsilon$ approximation for MWCDS in WUDG [37]. Dai et al. achieved a $5+\epsilon$-approximation algorithm for MWDS and a 4-approximation ratio to connect the DS, and their algorithm yields a $9+\epsilon$ approximation algorithm for MWCDS in WUDG [14]. Zou et al. designed a $4+\epsilon$-approximation algorithm for an MWDS based on a dynamic programming algorithm for a Min-Weight Chromatic Disk Cover, and proposed a $1+\epsilon$-approximation algorithm for the connecting part and thus obtained a $5+\epsilon$-approximation algorithm for an MWCDS [8].

Other works have focused on the minimum dominating set and the minimum (weighted) edge dominating set problems [38–41].

5.3. Link scheduling

In wireless sensor and ad hoc networks, nodes communicate with each other by wireless radio, and simultaneous transmissions among neighboring nodes can cause interference and so reduces network capacity. One scheme to achieve robust and collision free communication is to utilize a random access MAC layer, another is to carefully construct a transmission scheduling scheme [7], i.e., link scheduling.

Alicherry et al. mathematically formulated the joint channel assignment and routing problem by considering the interference constraints, the number of channels in the whole network and the number of radios available at each mesh router [10]. They also developed an algorithm to optimize the overall network throughput subject to fairness constraints on the allocation scarce wireless capacity among mobile clients. The algorithm obtains a constant factor ratio for any optimal algorithm. Wang et al. developed link scheduling for a multi-hop wireless network with a single channel and g gateways [7]. This work, respectively, studied the conditions sufficient for interference-free performance in the RTS/CTS model and the protocol-interference model with fixed transmission power. They proved that the number of time-slots needed by the faster distributed algorithms are at most $O(\min(\log n, \log\psi))$ in the paper. Ma et al. proposed an interference-free TDMA sleep scheduling solution, called contiguous link scheduling that assigns sensors with consecutive time slots to reduce the frequency of state transitions [42]. Both references [7,42] also designed centralized and distributed algorithms that use a number of time slots that is at most a constant factor times the optimum.

The link scheduling problem can be reformulated as a coloring problem, and the link scheduling scheme can be used to achieve the upper bound of a wireless network's capacity. Grandham et al. formulated the link scheduling in sensor networks under a TDMA MAC protocol as an edge coloring problem, and designed a distributed edge coloring algorithm that requires at most $\Delta+1$ colors [39]. Xu et al. proposed a *scheduling partition* methodology to address the feasibility of maximum capacity scaling in large-scale wireless networks, and then designed a simple localized link scheduling algorithm [22].

Time scheduling can be easily reduced to the *edge and vertex coloring* problem. Refer to the related work of time scheduling in Section VIII of the reference [7].

6. Conclusions

We designed an EDAT based on a CDS to use the energy harvested by sensor nodes most efficiently by minimizing the weight differences among each pair of nodes. We prove analytically that the energy consumption difference of this EDAT is no more than $\frac{5\mathcal{H}|\mathcal{S}|^2}{n-1}$. We also designed a new link-scheduling scheme to calculate the number of time slots $\widehat{L}$ necessary for scheduling all the links in the EDAT. The number of time slots needed is bounded by the interval $[4\delta-1, 2l_k\varrho^2]$. Because the amount of the energy harvested changes with the weather, it is challenging but important to carefully allocate the energy for each node to allow the whole network to operate consecutively. Therefore, we determined that the necessary condition to support consecutive network operation is $\mathcal{H}_{min} \geqslant \Delta(4\delta-1)$.

In future work, it remains an open issue how to design an adaptive algorithm to dynamically schedule the harvested energy to best adapt to the unpredictable weather. We plan to adopt adaptive control theory to achieve the minimum difference in the harvested energy when the harvested energy $\mathcal{H}_u$ of each node u is related to time, i.e., $\mathcal{H}_u(T_i) \neq \mathcal{H}_u(T_j)$, where $i \neq j$.

Acknowledgements

This work is supported by the National Natural Science Foundation of China under Grants No. 61003298 and 60803126 and the Zhejiang Provincial Key Innovation Team on Sensor Networks under Grant No. 2009R50046 and its sub-foundation under Grant No. 2011R09046-07.

References

[1] J. Taneja, J. Jeong, D. Culler, Design, modeling, and capacity planning for micro-solar power sensor networks, in: IPSN, 2008, pp. 407–418.
[2] S. Roundy, P. Wright, J. Rabaey, A study of low level vibrations as a power source for wireless sensor nodes, Computer Communications 26 (11) (2003) 1131–1144.
[3] X. Jiang, J. Polastre, D. Culler, Perpetual environmentally powered sensor networks, in: IEEE Fourth International Symposium on Information Processing in Sensor Networks (IPSN), 2005, pp. 463–468.
[4] T. Zhu, Z. Zhong, Y. Gu, T. He, Z. Zhang, Leakage-aware energy synchronization for wireless sensor networks, in: Proceedings of the Seventh International Conference on Mobile Systems, Applications, and Services (Mobisys), ACM, 2009, pp. 319–332.
[5] W. Chen, U. Mitra, Energy efficient scheduling with individual packet delay constraints, in: IEEE INFOCOM, 2006.
[6] Y. Yang, L. Wang, D. Noh, H. Le, T. Abdelzaher, Solarstore: enhancing data reliability in solar-powered storage-centric sensor networks, in: Proceedings of the Seventh International Conference on Mobile Systems, Applications, and Services(MobiSys), ACM, 2009, pp. 333–346.
[7] Y. Wang, W. Wang, X. Li, W. Song, Interference-aware joint routing and tdma link scheduling for static wireless networks, IEEE Transactions on Parallel and Distributed Systems (TPDS) (2008) 1709–1725.
[8] F. Zou, Y. Wang, X. Xu, X. Li, H. Du, P. Wan, W. Wu, New approximations for minimum-weighted dominating sets and minimum-weighted connected dominating sets on unit disk graphs, Theoretical Computer Science.

[9] P. Wan, K. Alzoubi, O. Frieder, Distributed construction of connected dominating set in wireless ad hoc networks, Mobile Networks and Applications 9 (2) (2004) 141–149.

[10] M. Alicherry, R. Bhatia, L. Li, Joint channel assignment and routing for throughput optimization in multi-radio wireless mesh networks, in: Proceedings of the 11th annual international conference on Mobile computing and networking (Mobicom), ACM, 2005, pp. 58–72.

[11] K. Jain, J. Padhye, V. Padmanabhan, L. Qiu, Impact of interference on multi-hop wireless network performance, Wireless networks 11 (4) (2005) 471–487.

[12] D. Li, H. Du, P. Wan, X. Gao, Z. Zhang, W. Wu, Construction of strongly connected dominating sets in asymmetric multihop wireless networks, Theoretical Computer Science 410 (8-10) (2009) 661–669.

[13] M. Thai, F. Wang, D. Liu, S. Zhu, D. Du, Connected dominating sets in wireless networks with different transmission ranges, IEEE transactions on mobile computing (TOMC) (2007) 721–730.

[14] D. Dai, C. Yu, A $5+\epsilon$-approximation algorithm for minimum weighted dominating set in unit disk graph, Theoretical Computer Science 410 (8–10) (2009) 756–765.

[15] Y. Wang, W. Wang, X. Li, Distributed low-cost backbone formation for wireless ad hoc networks, in: Proceedings of the Sixth ACM International Symposium on Mobile Ad hoc Networking and Computing (MobiHoc), 2005, pp. 2–13.

[16] J. Wu, M. Cardei, F. Dai, S. Yang, Extend dominating set and its applications in ad hoc networks using cooperative communication, IEEE Transactions on Parallel and Distributed Systems (TPDS) 17 (8) (2006) 1–12.

[17] S. Guha, S. Khuller, Approximation algorithms for connected dominating sets, Algorithmica 20 (4) (1998) 374–387.

[18] M.T. Thai, R. Tiwari, D. zhu Du, On construction of virtual backbone in wireless ad hoc networks with unidirectional links, IEEE TOMC 7 (9) (2008) 1098–1109.

[19] X. Li, Wireless Ad Hoc and Sensor Networks: Theory and Applications, Cambridge University Press, 2008.

[20] T. Chandra, D. Chatterjee, A First Course in Probability, Alpha Science Intl Ltd, 2005.

[21] P. Wan, C. Scott, L. Wang, Z. Wan, X. Jia, Minimum-Latency Aggregation Scheduling in Multihop Wireless Networks, in: MobiHoc, ACM, New York, NY, USA, 2009, pp. 185–194.

[22] Y. Xu, W. Wang, Scheduling Partition for Order Optimal Capacity in Large-scale Wireless Networks, in: MobiCom, ACM, 2009, pp. 109–120.

[23] K. Lin, J. Yu, J. Hsu, S. Zahedi, D. Lee, J. Friedman, A. Kansal, V. Raghunathan, M. Srivastava, Heliomote: enabling long-lived sensor networks through solar energy harvesting, in: Proceedings of the Third International Conference on Embedded Networked Sensor Systems (SenSys), ACM, 2005. 309–309.

[24] P. Dutta, J. Hui, J. Jeong, S. Kim, C. Sharp, J. Taneja, G. Tolle, K. Whitehouse, D. Culler, Trio: enabling sustainable and scalable outdoor wireless sensor network deployments, in: Proceedings of the Fifth International Conference on Information Processing in Sensor Networks (IPSN), ACM, 2006, pp. 407–415.

[25] C. Park, P. Chou, Ambimax: autonomous energy harvesting platform for multi-supply wireless sensor nodes, in: Third Annual IEEE Communications Society on Sensor and Ad Hoc Communications and Networks(SECON), vol. 1, 2006, pp. 168–177.

[26] C. Park, P. Chou, Power utility maximization for multiple-supply systems by a load-matching switch, in: Proceedings of the International Symposium on Low Power Electronics and Design, 2004, pp. 168–173.

[27] L. Wang, Y. Yang, D. Noh, H. Le, J. Liu, T. Abdelzaher, M. Ward, AdaptSens: an adaptive data collection and storage service for solar-powered sensor networks, in: 30th IEEE Real-Time Systems Symposium (RTSS), 2009, pp. 303–312.

[28] Y.Y., S.L., G.Y., T.F. Abdelzaher, Solarized: utilizing erasure codes for reliable data delivery in solar-powered wireless sensor networks, in: IEEE INFOCOM, 2010, pp. 1–5.

[29] E. Lattanzi, E. Regini, A. Acquaviva, A. Bogliolo, Energetic sustainability of routing algorithms for energy-harvesting wireless sensor networks, Computer Communications 30 (14–15) (2007) 2976–2986.

[30] I. Stojmenovic, M. Seddigh, J. Zunic, Dominating sets and neighbor elimination-based broadcasting algorithms in wireless networks, IEEE Transactions on Parallel and Distributed Systems (TPDS) (2002) 14–25.

[31] K. Alzoubi, P. Wan, O. Frieder, New distributed algorithm for connected dominating set in wireless ad hoc networks, in: HICSS, 2002, pp. 297–297.

[32] B.N. Clark, C.J. Colbourn, D.S. Johnson, Unit disk graphs, Discrete Mathematics 86 (1990) 165–177.

[33] P.J. Wang, K.M. Alzoubi, O. Frieder, Distributed construction of connected dominating set in wireless ad hoc networks, Mob. Netw. Appl. 9 (2) (2004) 141–149.

[34] B. Das, V. Bharghavan, Routing in ad-hoc networks using minimum connected dominating sets, in: IEEE International Conference on Communications (ICC), vol. 1, 1997, pp. 376–380.

[35] J.Wu, H.L.Li, On calculating connected dominating set for efficient routing in ad hoc wireless network, in: DIAL-M, 1999, pp. 7–14.

[36] C. Ambühl, T. Erlebach, M. Mihalák, M. Numkesser, Constant-factor approxiamtion for minimum-weight connected dominating sets in unit disk graphs, in: Approx-Random, 2006, pp. 3–14.

[37] Y. Huang, X. Gao, Z. Zhang, W. Wu, A better constant-factor approximation for weighted dominating set in unit disk graph, Journal of Combinatorial Optimization 18 (2) (2009) 179–194.

[38] M. Yannakakis, F. Gavril, Edge dominating sets in graphs, SIAM Journal on Applied Mathematics (1980) 364–372.

[39] S. Gandham, M. Dawande, R. Prakash, Link scheduling in wireless sensor networks: distributed edge-coloring revisited, Journal of Parallel and Distributed Computing 68 (8) (2008) 1122–1134.

[40] T. Fujito, H. Nagamochi, A 2-approximation algorithm for the minimum weight edge dominating set problem, Discrete Applied Mathematics 118 (3) (2002) 199–207.

[41] A. Berger, O. Parekh, Linear time algorithms for generalized edge dominating set problems, Algorithmica 50 (2) (2008) 244–254.

[42] J. Ma, W. Lou, Y. Wu, X. Li, G. Chen, Energy efficient tdma sleep scheduling in wireless sensor networks, in: IEEE INFOCOM, 2009, pp. 630–638.