WIRELESS COMMUNICATIONS AND MOBILE COMPUTING
Wirel. Commun. Mob. Comput. 2009; **9**:325–334
Published online 14 September 2007 in Wiley InterScience
(www.interscience.wiley.com) DOI: 10.1002/wcm.544

Lifetime-maximized cluster association in two-tiered wireless sensor networks

Wen-Zhan Song[1*,†], WeiZhao Wang[2], Kousha Moaveni-Nejad[2] and Xiang-Yang Li[2]
[1]*Washington State University, Vancouver, WA 98686, U.S.A.*
[2]*Illinois Institute of Technology, Chicago, IL, U.S.A.*

Summary

In this paper, we study the two-tiered wireless sensor network (WSN) architecture and propose the optimal cluster association algorithm for it to maximize the overall network lifetime. A two-tiered WSN is formed by number of small sensor nodes (SNs), powerful application nodes (ANs), and base-stations (BSs, or gateways). SNs capture, encode, and transmit relevant information to ANs, which then send the combined information to BSs. Assuming the locations of the SNs, ANs, and BSs are fixed, we consider how to associate the SNs to ANs such that the network lifetime is maximized while every node meets its bandwidth requirement. When the SNs are homogeneous (e.g., same bandwidth requirement), we give optimal algorithms to maximize the lifetime of the WSNs; when the SNs are heterogeneous, we give a 2-approximation algorithm that produces a network whose lifetime is within 1/2 of the optimum. We also present algorithms to dynamically update the cluster association when the network topology changes. Numerical results are given to demonstrate the efficiency and optimality of the proposed approaches. In simulation study, comparing network lifetime, our algorithm outperforms other heuristics almost twice. Copyright © 2007 John Wiley & Sons, Ltd.

KEY WORDS: network lifetime; two-tiered; clustering

1. Introduction

The deployment of tiny sensors to large-scale wireless sensor networks (WSNs) raises massive challenges. Due to large scale, it is natural to adopt the two-tiered (even multiple-tiered) architecture. A two-tiered WSN is formed by number of small sensor nodes (SNs), powerful application nodes (ANs), and base-stations (BSs, or gateways). SNs capture, encode, and transmit relevant information to ANs, which then send the combined information to BSs. In fact, some works have already addressed different issues regarding this hierarchical architecture, including minimizing the number of clusters [1,2], minimizing the total energy consumption [3] and maximizing the lifetime [4,5]. Following the work in Reference [4], we consider how to associate the SNs to ANs such that the network lifetime is maximized while every node meets its bandwidth requirement. When the SNs are homogeneous (e.g., same bandwidth requirement), we give optimal algorithms to maximize the lifetime of the WSNs; when the SNs are heterogeneous, we give a 2-approximation algorithm that produces a network whose lifetime is within 1/2 of the optimum.

*Correspondence to: Wen-Zhan Song, Washington State University, Vancouver, WA 98686, U.S.A.
†E-mail: song@vancouver.wsu.edu

We also present algorithms to dynamically update the cluster association when the network topology changes. Numerical results are given to demonstrate the efficiency and optimality of the proposed approaches. In simulation study, comparing network lifetime, our algorithm outperforms other heuristics almost twice.

1.1. Problem Definition

A two-tiered WSN consists of a set of small SNs, denoted as $S_M = \{s_1, s_2, \ldots, s_m\}$, a set of ANs, denoted as $V_N = \{v_1, v_2, \ldots, v_n\}$, and at least one BS. The ANs and SNs form *clusters*, and in each cluster there are many SNs and one AN. For simplicity, we assume that the AN v_i is in cluster $\mathcal{C}_i$ and the set of small sensors in cluster $\mathcal{C}_i$ is $S_i \subseteq S_M$. Small sensors capture and encode the environmental phenomena and broadcast the data directly to ANs within its transmission range or via the relay of other sensors. Here, if AN v_i can receive the data from the small sensor s_j, then we call v_i is a neighbor of s_j. Here, sensor s_j may have to reach AN v_i via relay of other sensors. For notational simplicity, we use $N(v_i)$ to denote the neighboring small sensors of AN v_i. Remember that although several ANs can receive the data packets from the small sensor s_j, only the AN in the same cluster as s_j processes the information. Here, we assume that once formed, the cluster formation does not change over the time. We also let r_i be the data-rate of the small sensor s_i generates and $r(S) = \sum_{s_i \in S} r_i$ be the total data-rates produced by a set of small sensors S. Usually, the data-rate $r_i(t)$ is a function over the time t instead of a constant. However, if we average the rate over a period of time T. Thus, we can define the rate r_i as the average rate over a period of time, that is, $r_i = \int_{T_0}^{T+T_0} r_i(t)/T$. It is reasonable to expect that the life time of an AN *decreases* when the number of small sensors in its cluster *increases*. Given an AN v_i, let $S_i \in S_M$ be the set of small sensors in its logical cluster. The power consumption of the AN v_i is a general function $p_i(r(S_i), N(v_i))$, where $r(S_i)$ is the total data-rate of the small sensors in S_i. Since $N(v_i)$ does not depend on the cluster formation and can be taken as a constant for a given AN v_i, we can simplify the power consumption function as $p_i(r(S_i))$. The only assumption in this paper is that function $p_i(x)$ should satisfy that $p_i(x) > p_i(x')$ when $x > x'$. Notice that, the above Monotone-increasing property is only assumed to be true for each AN. For two different ANs v_i and v_j, it is possible that $p_i(x) < p_j(x')$ when $x > x'$. In this paper, we assume that $\mathcal{P}_i$ is the initial battery power level of the AN v_i and $p_i(r(S_i))$ is its *average* energy consumption rate when the set of small sensors S_i is in the cluster $\mathcal{C}_i$. The lifetime of an individual AN v_i is define as $l_i = \mathcal{P}_i / p_i(r(S_i))$. We adopt the following *network lifetime* definitions for theoretical analysis and simulations: (1) critical AN lifetime (CANLT): the mission fails when any AN runs out of energy, that is, the lifetime L_N is $L_N = \min_{i=1}^{n}\{l_i\}$. (2) full coverage lifetime (FCLT): the mission fails when the covered area of the WSN is smaller than the originally covered area (FCLT).

1.2. Related Works

Numerous literatures have discussed efficient cluster formation for wireless ad hoc and SNs. Although almost all works assumed that there are some nodes acting as *clusterheads* who are in charge of gathering the information from other nodes and sending back to some BSs, the criteria of forming the clusters vary from case to case. One fundamental difference between the cluster formation problem studied in this paper and the traditional cluster formation problems is that *every* node could be a clusterhead in the traditional methods, while only the AN can be the clusterhead for the problems studied here. In the linked cluster algorithm (LCA) [1], a node becomes the clusterhead if it has the highest identity among all nodes within 1-hop of itself or among all nodes within 1-hop of one of its neighbors. This algorithm was improved by the LCA2 algorithm [6], which generates a smaller number of clusters. The LCA2 algorithm elects the node, with the lowest ID among all nodes which are not within 1-hop of any chosen clusterheads, as a new clusterhead. The algorithm proposed in Reference [7], chooses the node with highest degree among its 1-hop neighbors as a clusterhead. In Reference [8], the authors propose a distributed algorithm that is similar to the LCA2 algorithm. The distributed clustering algorithm (DCA) uses weights associated with nodes to elect clusterheads [9]. It elects the node that has the highest weight among its 1-hop neighbors as the clusterhead. The DCA algorithm is suitable for networks in which nodes are static or moving at a very low speed. Results reported in References [4,5] are closest to this paper in spirit. In Reference [4], Pan *et al.* studied the problem of maximizing lifetime of a two-tiered WSN with focus on the top-tier. By assuming the *prior known* fixed cluster formation, the authors mainly studied how to place the BS in the network such that the lifetime of the WSN is maximized. The ANs are assumed to be homogenous in Reference [4]

DOI: 10.1002/wcm

and generalized to be heterogenous in Reference [5]. The authors also discussed how to relay the packets via ANs to some fixed based stations. In this paper, we will focus on the lower-tier of the two-tiered WSN: how to form the cluster (associate small sensors to ANs) so the network lifetime is maximized.

2. Homogeneous Small Sensors

In this section, we study the case when the small sensors are homogeneous, that is, all small sensors have the same data rate r. Thus $r(S) = r \cdot |S|$, where $|S|$ is the number of small sensors in the set S.

2.1. Homogeneous Application Nodes

In this subsection, we discuss how to maximize the lifetime of the WSN when all ANS are homogeneous, that is, their initial on-board energy are the same, say $\mathcal{P}$ and the energy consumption functions are the same, say $p(x)$. Remember that $L_N = \min_{i=1}^{n}\{l_i\} = \min_{i=1}^{n}(\mathcal{P}/p(r \cdot |S_i|))$ and $p(x)$ is increasing. Thus, maximizing the lifetime L_N of the WSN is equivalent to minimizing the maximum cluster size. For simplicity, we denote $x_{i,j} = 1$ if the sensor s_j belongs to cluster $\mathcal{C}_i$, and $x_{i,j} = 0$ otherwise. Let $N(v_i)$ be the set of sensors who are v_i's neighbors. We formalize the problem of maximizing L_N as the following Integer Programming.

$$\text{IP (1)}: \quad \min \max_{v_i \in V_N} \sum_{s_j \in S_M} x_{i,j} \tag{1}$$

Subject to constraint set (1)

$$\text{CS (1)}: \; x_{i,j} = 0, \forall v_i, \forall s_j \notin N(v_i); \quad x_{i,j} \in \{0, 1\}, \forall s_j, \forall v_i; \quad \sum_{v_i} x_{i,j} = 1, \forall s_j \tag{2}$$

Obviously, a feasible solution of the IP (1) problem is a *feasible* cluster formation. For simplicity, the set of small sensors in the cluster $\mathcal{C}_i$ is denoted as S_i in this paper, when no confusion is caused. For simplicity, let $\mathbf{x}^{\min}$ be the solution to IP (1) and $T^{\min} = \min \max_{v_i \in V_N} \sum_{s_j \in S_M} \mathbf{x}_{i,j}^{\min}$. Next, we present two different approaches to solve the IP (1) exactly.

2.1.1. Efficient centralized approach

Note that $T^{\min}$ is a non-negative integer at most m, thus it could have m possible values. For an given integer k, if we can decide whether we can find a feasible solution $\mathbf{x}$ such that (1) it satisfies constraint CS (1); (2) $\min \max_{v_i \in V_N} \sum_{s_j \in S_M} \mathbf{x}_{i,j}^{\min} = k$, then by performing a binary search on $T^{\min}$ we can find the exact value. Following, we use a Max-Flow approach to find a feasible solution for given integer k if it exists. The idea is that we construct a flow network as shown in Figure 1 with s as the source and t as the sink. There is a directional link $\overrightarrow{sv_i}$, $1 \le i \le n$ with capacity k, a directional link between $\overrightarrow{v_i s_j}$ with capacity 1 if $s_j \in N(v_i)$ and a directional link $\overrightarrow{s_j t}$ with capacity 1. Usually, for a maximum flow problem, the flow on each directional link could be any real number. Fortunately, all capacities in the graph take on only integral values. Thus, the maximum flow f has the property that $|f|$ is integer-valued. Moreover, for all vertices u and v, the flow on edge uv is an integer. Therefore, each link $\overrightarrow{v_i s_j}$ is either 0 or 1. Remember that the flow on link $\overrightarrow{v_i s_j}$ corresponds to $x_{i,j}$, which implies that $x_{i,j} \in \{0, 1\}$ for every v_i and $s_j \in N(v_i)$. Thus, a flow in Figure 1 and a solution to IP (1) has an one-to-one mapping.

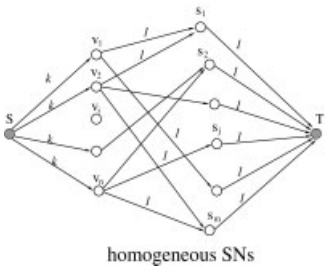


Fig. 1. A flow network for two-tiered WSN.

We can find the solution to IP (1) by solving $\log m$ max-flow problems for different values of T. Thus, the time complexity for Max-Flow approach is $m \cdot \log m \cdot (n + m)^3$, which is very expensive and impractical. Notice that the cluster formation problem with minimum cluster size becomes the Maximum Cardinality Matching problem in a bipartite graph [10]. In Reference [10], Hopcroft and Karp presents the best-known algorithm that achieves the time complexity $\sqrt{m} \cdot nm$. This reduces the time complexity from $O((n + m)^3)$ to $O(nm \cdot (n + m)^{1/2} \log(n + m))$ for a fix value T. Therefore, we can solve the IP (1) in time $O(n \cdot m^{3/2} \log^2(m))$.

2.1.2. Efficient distributed algorithm by smoothing

Although the previous approach computes a clustering quickly in centralized manner, it may be too expensive to collect the necessary information. In this

DOI: 10.1002/wcm

subsection, we propose a different approach that can be implemented efficiently in a distributed manner. The basic idea of this approach is to construct a virtual directed graph on ANs and iteratively move the sensors from those clusters who have the largest number of small sensors to smaller clusters. In the virtual directed graph, there is an edge $\overrightarrow{v_i v_k}$ from AN v_i to v_k if there is a sensor s_j that can be moved from the cluster of v_i to the cluster of v_k. The weight of the edge is the number of such small sensors that can be moved from the cluster of v_i to the cluster of v_k. Following algorithm presents the method constructing a virtual graph based on a feasible solution $\mathbf{x}$ to CS (1).

Algorithm 1. Constructing the virtual graph.

1: Set V_N as the vertices for virtual graph VG.
2: **for** every pair of v_i and s_j such that $x_{i,j} = 1$ **do**
3: **for** every v_k such that $s_j \in N(v_k)$ **do**
4: **if** there is no directed edge $\overrightarrow{v_i v_k}$ from v_i to v_k **then**
5: Add a directed edge $\overrightarrow{v_i v_k}$ from v_i to v_k. Set the weight of the edge to $c(v_i v_k) = 1$.
6: **else**
7: Update the weight as $c(v_i v_k) = c(v_i v_k) + 1$.

In the directed virtual graph VG($\mathbf{x}$), if there is a path from v_i to v_j, then we say v_i *reaches* v_j. All vertices that v_i can reach forms a set $\mathcal{R}_i(\mathbf{x})$, called the *clique* centered at the AN v_i. Given a solution $\mathbf{x}$ of CS (1) and its corresponding virtual graph VG($\mathbf{x}$), we have the following property about cliques (its proof is omitted due to space limit).

Lemma 1. *Given a feasible assignment* $\mathbf{x}$ *of small sensors to ANs and its corresponding virtual graph* VG($\mathbf{x}$)*, for any AN* v_i *and its clique* $\mathcal{R}_i(\mathbf{x})$ *in* VG($\mathbf{x}$)*, if* $\mathbf{y}$ *is also a feasible assignment of SNs to ANs, we have* $\sum_{v_j \in \mathcal{R}_i(\mathbf{x})} |S_i(\mathbf{x})| \leq \sum_{v_j \in \mathcal{R}_i(\mathbf{y})} |S_i(\mathbf{y})|$.

The Algorithm relies on the relation between $\omega_i(\mathbf{x})$, $\omega_j(\mathbf{x})$, and $T^{\min}$ where v_i is the AN with the largest weight and v_j is the AN with the smallest weight in $\mathcal{R}_i(\mathbf{x})$.

Lemma 2. *Let* v_i *be the AN with the largest weight and* v_j *be the AN with the smallest weight in* $\mathcal{R}_i(\mathbf{x})$ *under any feasible assignment* $\mathbf{x}$*, then* $|S_j(\mathbf{x})| \leq T^{\min} \leq |S_i(\mathbf{x})|$.

Proof. The proof is omitted here due to space limit. Please refer to full version of the paper for more details. □

Given a virtual graph constructed by Algorithm 1 based on a feasible assignment of SNs to ANs, our approach to find a better solution is to iteratively apply a process called SMOOTH to reduce the maximum weight of the ANs if possible. Here, the weight of an AN v_i under a feasible assignment $\mathbf{x}$ is the number of small sensors assigned to the cluster $\mathcal{C}_i$, denoted as $\omega_i(\mathbf{x})$.

Algorithm 2. Smooth algorithm.

1: Construct virtual graph VG($\mathbf{x}$) based on the $\mathbf{x}$ using Algorithm 1.
2: **repeat**
3: Find any AN with the largest weight, say v_i.
4: Find any AN with the smallest weight in $\mathcal{R}_i(\mathbf{x})$, say v_j.
5: Apply procedure SMOOTH(v_i, v_j, VG($\mathbf{x}$), $\mathbf{x}$).
6: **until** $\omega_i(\mathbf{x}) \leq \omega_j(\mathbf{x}) + 1$

Algorithm 3. SMOOTH(v_i, v_j, VG($\mathbf{x}$), $\mathbf{x}$).

1: Let $v_{i_0} v_{i_1} \cdots v_{i_k}$ be the path connecting v_i and v_j with the minimum number of hop. Here, $v_{i_0} = v_i$ and $v_{i_k} = v_j$.
2: **for** $t = 0$ to $k - 1$ **do**
3: Assume that $x_{t,l} = 1$ for some SN s_ℓ with $s_\ell \in N(v_t)$ and $s_\ell \in N(v_{t+1})$. Set $x_{t,l} = 0$ and $x_{t+1,l} = 1$, that is, move s_ℓ from cluster $\mathcal{C}_t$ to cluster $\mathcal{C}_{t+1}$.
4: **for** every v_a such that $s_\ell \in N(v_a)$ **do**
5: Update $c(\overrightarrow{v_{i_t} v_a}) = c(\overrightarrow{v_{i_t} v_a}) - 1$. Remove directed link $\overrightarrow{v_{i_t} v_a}$ if $c(\overrightarrow{v_{i_t} v_a}) = 0$.
6: Update $c(\overrightarrow{v_{i_{t+1}} v_a}) = c(\overrightarrow{v_{i_{t+1}} v_a}) + 1$. Add a directed link $\overrightarrow{v_{i_{t+1}} v_a}$ if $c(\overrightarrow{v_{i_{t+1}} v_a}) = 1$.
7: Set $\omega_j(\mathbf{x}) = \omega_j(\mathbf{x}) + 1$ and $\omega_i(\mathbf{x}) = \omega_i(\mathbf{x}) - 1$.

Theorem 3. *Algorithm* 2 *terminates after at most* m *iterations, with an solution to* IP (1).

Proof. From Lemma 2, we have $\omega_j(\mathbf{x}) \leq T^{\min} \leq \omega_i(\mathbf{x})$. If Algorithm 2 does not stop at this iteration, we have $\omega_i(\mathbf{x}) > \omega_j(\mathbf{x}) + 1$, which implies that $T^{\min} > \omega_j(\mathbf{x}) + 1$. For a feasible solution $\mathbf{x}$, we define $\delta_i(\mathbf{x}) = |\mathcal{S}_i(\mathbf{x})| - |\mathcal{S}_i(\mathbf{x}^{\min})|$ if $\omega_i(\mathbf{x}) > T^{\min}$ and 0 otherwise. Let $\Delta(\mathbf{x}) = \sum_{v_i \in V_N} \delta_i(\mathbf{x})$, it is not difficult to observe that $\Delta(\mathbf{x})$ will be decreased by 1 for each iteration. Thus, Algorithm 2 terminates after at most m iterations.

Remember when Algorithm 2 terminates, we have $\omega_i(\mathbf{x}) \leq \omega_j(\mathbf{x}) + 1$. Combining with the relation $\omega_j(\mathbf{x}) \leq T^{\min} \leq \omega_i(\mathbf{x})$, we have $\omega_j(\mathbf{x}) \leq T^{\min} \leq \omega_i(\mathbf{x}) \leq \omega_j(\mathbf{x}) + 1$. This implies that

DOI: 10.1002/wcm

$T^{\min} = \omega_i(\mathbf{x}) - 1$ or $T^{\min} = \omega_i(\mathbf{x})$. First, we consider the case when $T^{\min} = \omega_i(\mathbf{x}) - 1$. In this case, we have $\omega_j(\mathbf{x}) \geq T^{\min}$ for every $v_j \in \mathcal{R}_i(\mathbf{x})$ which implies $\sum_{v_j \in \mathcal{R}_i(\mathbf{x})} |S_i(\mathbf{x})| \geq T^{\min} \cdot |\mathcal{R}_i(\mathbf{x}) - 1| + T^{\min} + 1$. For the solution $\mathbf{x}^{\min}$ of IP (1), every AN's weight is not greater than $T^{\min}$. Thus, $\sum_{v_j \in \mathcal{R}_i(\mathbf{x})} |S_i(\mathbf{x}^{\min})| \leq T^{\min} \cdot |\mathcal{R}_i(\mathbf{x})|$. From Lemma 1, we have $\sum_{v_j \in \mathcal{R}_i(\mathbf{x})} |S_i(\mathbf{x}^{\min})| \geq \sum_{v_j \in \mathcal{R}_i(\mathbf{x})} |S_i(\mathbf{x})|$. This implies that $T^{\min} \cdot |\mathcal{R}_i(\mathbf{x})| \geq T^{\min} \cdot |\mathcal{R}_i(\mathbf{x}) - 1| + T^{\min} + 1$, which is a contradiction. Thus, $T^{\min} = \omega_i(\mathbf{x})$. Remember that $\mathbf{x}$ is a solution to the CS (1). Therefore, $\mathbf{x}$ is a solution to IP (1). This finishes our proof. □

Now we analyze the time complexity of Algorithm 2. In procedure SMOOTH(v_i, v_j, VG($\mathbf{x}$), $\mathbf{x}$), there are at most n nodes on the path between v_i, v_j, and up to n iterations in the 'FOR' loop between line 4 and 7. Thus, the time complexity of SMOOTH(v_i, v_j, VG($\mathbf{x}$), $\mathbf{x}$) is $O(n^2)$. From Theorem 3, it takes at most $O(m \cdot n^2)$ for Algorithm 2 to terminate. Constructing the virtual graph based on a feasible solution $\mathbf{x}$ could take time $O(m \cdot n^2)$. Thus, the total time complexity of smoothing algorithm is also $O(m \cdot n^2)$. If $n = o(\sqrt{m})$, then Algorithm 2 outperforms the best known max-flow algorithm by $\log^2 m$; when n is a constant the time complexity becomes $O(m)$ which is optimal.

2.1.3. Efficient distributed implementation

So far we have illustrated the basic idea of the Smoothing algorithm, which clearly can be implement in a distributed manner. In the remainder of the section, we will describe how this method can be implemented efficiently. Given an AN v_i, we say v_i is adjacent to AN v_j if there is a small sensor s_k in the cluster $\mathcal{C}_i \cap N(v_j)$. If v_i and v_j are not adjacent, then we define the distance between v_i and v_j as the smallest number of hops between them if we consider the adjacent graph of the ANs. For an AN v_i that is adjacent v_j, let ℓ be the largest non-negative integer such that $p_j(r \cdot \sum_{s_k \in S_M} x_{j,k} + \ell \cdot r)/\mathcal{P}_j < \omega_i(\mathbf{x})$. We define the *difference* of v_i and v_j as $\mathrm{dif}_{i,j}(\mathbf{x}) = \ell$. Based on the notation of difference, we have following localized algorithm.

Regarding the distributed Algorithm 4, we have the following theorem:

Theorem 4. *Algorithm* 4 *converges in at most* $m \cdot n$ *rounds and total message complexity is* $O(n^2 \cdot m)$ *if the ANs are homogeneous.*

Algorithm 4. Distributed smoothing algorithm for AN v_i.

1: When v_i receive an UPDATE-LEAVE or UPDATE-JOIN message from an adjacent AN v_j, it updates $\gamma_{i,j}$ if necessary.
2: Let v_j be one of v_i's adjacent AN with the maximum difference. Here, we break the tie arbitrarily.
3: **if** $\mathrm{dif}_{i,j}(\mathbf{x}) \geq 1$ **then**
4: Send a REQUEST message to AN v_j.
5: When v_j receives all REQUEST messages the ANs that adjacent to it, it sends out an ACK message to the AN that has the maximum weight and REJECT messages to all other ANs.
6: **if** v_i receives an ACK message from v_j **then**
7: Choose one SN, say s_k, in $\mathcal{C}_i \cap N(v_j)$. Set $x_{i,k} = 0$ and send SUCC message with the ID k to v_j.
8: Upate $\gamma_{i,j} = \gamma_{i,j} - 1$ and send the UPDATE-LEAVE message with ID k to all adjacent ANs.
9: When v_j receives the SUCC message from v_i with ID k, it first sets $x_{j,k} = 0$ and $\gamma(j,i) = \gamma(j,i) + 1$. After that it also sends UPDATE-JOIN message with ID k to all adjacent ANs.

Remark: Afterward, we also say that the small sensor s_k is *migrating* from cluster $\mathcal{C}_i$ to $\mathcal{C}_j$.

Proof. Given an assignment $\mathbf{x}$, we denote $\kappa_i(\mathbf{x})$ as the number of small sensors in ith largest cluster. Let $\Gamma_i(\mathbf{x}) = \sum_{j=1}^{i} \kappa_j(\mathbf{x})$, and $\mathbf{x}^k$ be the assignment of sensors in round k. Considering $\Gamma^k = \sum_{i=1}^{n} \Gamma_i(\mathbf{x}^k)$. If there is a small sensor joining $\mathcal{C}_{i^k}$ and leaving $\mathcal{C}_{j^k}$ in round k, then $|S_{i^k}| > |S_{j^k}| + 1$ and $i^k < i^k$. Notice that after the small sensor migrating from cluster $\mathcal{C}_{i^k}$ to $\mathcal{C}_{j^k}$, $\Gamma_\ell(\mathbf{x}^k)$ decreases by 1 if $j < \ell \leq i$ and does not change otherwise. Thus, Γ^k decreases by 1 for every small sensor migrating. It is not difficult to observe that if there is no small sensor migrating in round k, then Algorithm 4 terminates. Since $\Gamma^1 < n \cdot m$, Algorithm 4 terminates in at most $n \cdot m$ rounds.

In every round, every AN sends only one REQUEST message and receives at most one REJECT message. Thus, there is at most $O(n)$ REQUEST and REJECT messages. It is also not difficult to observe that every AN sends at most one ACK messages. Thus, there are at most $O(n^2 \cdot m)$ REQUEST, ACK and REJECT messages in total. On the other hand, there is exact one UPDATE-LEAVE and UPDATE-JOIN message for every small sensor migrating. Thus, there are at most $O(n \cdot m)$ UPDATE-LEAVE and UPDATE-JOIN messages. Therefore, the overall message complexity is $O(n^2 \cdot m)$. □

DOI: 10.1002/wcm

Notice that the message complexity analysis is very pessimistic. In simulations, it is much smaller than the worst case analysis. Observe that when Algorithm 4 terminates, it not necessarily gives an optimal solution. However, Algorithm 4 gives the best solution among all localized algorithms in which every AN can only know the information of its adjacent ANs. Furthermore, if we define the diameter of the network as the largest distance of the ANs, we have the following theorem (its proof is omitted due to space limit).

Theorem 5. *When Algorithm* 4 *terminates, it gives an assignment with maximum cluster size at most* $T \leq T^{\min} + D$ *where D is the diameter of the network.*

2.2. Heterogeneous Application Nodes

In Subsection 2.1, we discuss how to form the clusters when both the small sensors and ANs are homogeneous. However, in practice, such node homogeneity cannot always be guaranteed. For example, the initial onboard energy of ANs built by different vendors may not be proportional to the bit-rate at which they generate, or the ANs could be redeployed (e.g., new ANs join the system long after old ANs have been activated). Furthermore, two different ANs may consume different energy to receive, process and send the information to the BS even given the same set of small sensors. Thus, it is more practical to assume the ANs are heterogenous. In this paper, we consider the heterogeneity in two ways: the initial on board energy $\mathcal{P}$ and energy consumption function $p(x)$ where x is the sum of the rate of the small sensors in the cluster.

In this subsection, we redefine *weight* of a AN v_i for assignment $\mathbf{x}$ as $\omega_i(\mathbf{x}) = p_i(r \cdot \sum_{s_j \in S_M} x_{i,j})/\mathcal{P}_i$, where $\mathcal{P}_i$ is the initial onboard energy and $p_i(x)$ is energy consumption function. Here, the lifetime of the network is defined as $L = \max \min_{v_i \in V_N} (\mathcal{P}_i / p_i(r \cdot \sum_{s_j \in S_M} x_{i,j})) = \min \max_{v_i \in V_N} \omega_i(x)$.

Thus, maximizing the lifetime is equivalent to minimizing the maximum weight over all ANs. Similar to the approach for the homogenous AN case, we formalize the problem as an Integer Programming as follows.

$$\text{IP (2)}: \quad \min \max_{v_i \in V_N} \frac{p_i\left(r \cdot \sum_{s_j \in S_M} x_{i,j}\right)}{\mathcal{P}_i}$$

Subject to constraint set (2):

$$\text{CS (2)}: \quad x_{i,j} = 0, \ \forall v_i, \forall s_j \notin N(v_i);$$
$$x_{i,j} \in \{0, 1\}, \forall s_j, \forall v_i; \quad \sum_{v_i} x_{i,j} = 1, \forall s_j$$

2.2.1. Smoothing algorithm

In this subsection, we shows that our smoothing Algorithm 2 also applies to the heterogenous case with only minor modification.

Algorithm 5. Smoothing algorithm for heterogenous ANs.

1: Find a feasible solution $\mathbf{x}$, for example, randomly assign every SN to a neighboring AN.
2: Construct a virtual graph VG($\mathbf{x}$) based on $\mathbf{x}$ by applying Algorithm 1.
3: **repeat**
4: Choose any one of AN with the largest weight randomly, say v_i.
5: Define $\omega_k^+(\mathbf{x}) = \frac{p_k(r \cdot \sum_{s_j \in S_M} x_{k,j} + r)}{\mathcal{P}_k}$.
6: Find any AN v_j with the smallest $\omega_j^+(\mathbf{x})$ in $\mathcal{R}_i(\mathbf{x})$. If there are more than one such ANs, choose one randomly.
7: Apply SMOOTH_HETE(v_i, v_j, VG($\mathbf{x}$), $\mathbf{x}$) if $\omega_i(\mathbf{x}) > \omega_j^+(\mathbf{x})$
8: **until** $\omega_i(\mathbf{x}) \leq \omega_j^+(\mathbf{x})$

Algorithm 6. SMOOTH_HET(v_i, v_j, $VG(\mathbf{x})$, $\mathbf{x}$).

1: Let $v_{i_0}(v_i) v_{i_1} \cdots v_{i_k}(v_j)$ be the path connecting v_i and v_j with the minimum number of hop. Here, $v_{i_0} = v_i$ and $v_{i_k} = v_j$.
2: **for** $t = 0$ to $k - 1$ **do**
3: Assume $x_{t,l} = 1$ and $s_\ell \in N(v_{t+1})$. Set $x_{t,l} = 0$ and $x_{t+1,l} = 1$.
4: **for** every v_a such that $s_\ell \in N(v_a)$ **do**
5: Update $c(v_{i_t} v_a) = c(v_{i_t} v_a) - 1$. Remove directed link $v_{i_t}(v_a)$ if $c(v_{i_t} v_a) = 0$.
6: **for** every v_b such that $s_\ell \in N(v_b)$ **do**
7: $c(v_{i_{t+1}} v_b) = c(v_{i_{t+1}} v_b) + 1$. Add a directed link $v_{i_{t+1}} v_b$ if $c(v_{i_{t+1}} v_b) = 1$.
8: Update $\omega_j(\mathbf{x}) = \omega_j^+(\mathbf{x})$ and $\omega_i(\mathbf{x}) = \frac{p_i(r \cdot \sum_{s_j \in S_M} x_{i,j} - r)}{\mathcal{P}_i}$.

Lemma 6. *Let* v_i *be the AN with the largest weight and* v_j *be the AN with the lowest weight that is reachable by* v_i *in a feasible assignment* $\mathbf{x}$. *Then* $\omega_j(\mathbf{x}) \leq T^{\min} \leq \omega_i(\mathbf{x})$.

Theorem 7. *Algorithm* 5 *outputs a solution of* IP (2) *and terminates after m iterations.*

The proof of this theorem is omitted here due to space limit. Surprisingly, the time complexity of Algorithm 5 is also $O(m \cdot n^2)$, which is exactly the same as in the

DOI: 10.1002/wcm

homogenous case. This reduces the time complexity by an order of $\sqrt{m}\log^2 m$ and more importantly, Algorithm 4 also works for the heterogenous case with only modification of the definition of difference. However, we only have the following conjecture for the convergence and message complexity of localized smoothing algorithm. It is an open and interesting problem to either prove or disprove the following conjecture.

Conjecture 1. *Algorithm* 4 *terminates after at most* $n \cdot m$ *rounds and the total message complexity* $O(n^2 \cdot m)$ *when the ANs are heterogenous.*

3. Heterogeneous Small Sensors

Usually in WSNs, several different kinds of sensors cooperate together to fulfill some certain goals. Some sensors may generate data at a higher rate than others do, for example, the visual sensors have a bit-rate that is much higher than the bit-rate generated by a temperature sensor. Even in scenarios when all small sensors are of same type, sometimes sensors located at different locations may need to sample the data at a different time interval. Thus, it is more reasonable to assume that in a WSN different type of sensors produce different bit-rates.

By assuming that every small sensor has its own data rate r_i, we formalize the problem of maximizing the lifetime as an Integer Programming as follows:

$$\text{IP (3)}: \quad \min \max_{v_i \in V_N} \frac{p_i(\sum_{s_j \in S_M} r_j \cdot x_{i,j})}{\mathcal{P}_i}$$

Subject to constraint set (3)

$$\text{CS (3)}: \quad x_{i,j} = 0, \ \forall v_i, \forall s_j \notin N(v_i); \quad x_{i,j} \in \{0,1\}, \forall s_j, \forall v_i; \quad \sum_{v_i} x_{i,j} = 1, \forall s_j$$

Unlike the case for homogenous SNs in which we can find the solution that maximizes the lifetime exactly, Theorem 8 shows that it is NP-Hard to find the solution to IP (3).

Theorem 8. *We can not find the solution of* IP (3) *in polynomial time if* $P \neq NP$.

Proof. We consider the special case when ANs are homogeneous. In this case, since $p_i(x) = p(x)$ is increasing, it is equivalent to minimizing the maximum $\sum_{s_j \in S_M} r_j \cdot x_{i,j}$ subject to constraints set (3). If every AN v_i satisfies that $N(v_i) = S_M - v_i$, then the problem becomes the traditional job scheduling problem [11,12], which is known to be NP-Hard. This finishes our proof. □

Since solving IP (3) is NP-hard, we will present an algorithm approximating the optimal solution by borrowing some ideas from job scheduling [13,14]. Again we transform IP (3) into Integer Programming as follows:

$$\text{IP (4)}: \quad \min T$$

Subject to constraints set (4)

$$\text{CS (4)}: \quad x_{i,j} = 0, \forall v_i, \forall s_j \notin N(v_i); \quad x_{i,j} \in \{0,1\}, \forall s_j, \forall v_i; \quad \sum_{v_i} x_{i,j} = 1, \forall s_j; \sum_{s_j \in S_M} r_j \cdot x_{i,j} \leq k_i, \forall v_i$$

Here $k_i = p_i^{-1}(\mathcal{P}_i \cdot T)$. Let $\mathbf{x}^{\min}$ be the solution to IP (4) and $T^{\min}$ be the $\min T$ under solution $\mathbf{x}^{\min}$. It is easy to observe that $\mathbf{x}_{ij}^{\min}$ satisfies the following constraint:

$$x_{i,j} = 0 \qquad \forall v_i, \forall s_j \quad r_j > k_i \tag{3}$$

If we relax the constraint $x_{i,j} \in \{0,1\}$, we obtain a Linear Programming (4). Let $x^\star$ be the solution to LP (4) plus constraint 3 and $T^\star$ be the value of $\min T$ under solution $x^\star$. Then $T^\star \leq T^{\min}$. By binary search on $T^\star$, we can find the solution $x^\star$ to LP (4) plus constraint 3 in polynomial time. Furthermore, we can find a solution $x^\star$ that has some special properties. For a small sensor s_j, if there exists an AN v_i such that $0 < x_{i,j} < 1$, we call s_j is fractionally assigned to cluster $\mathcal{C}_i$. We construct a graph with vertex $V_N \cup S_M$ and add an edge $s_j v_i$ if and only if $0 < x_{i,j} < 1$. Obviously, it is a bipartite graph and it is generally known [14,15] that we can transform the solution $x^\star$ to another solution x^* such that its corresponding bipartite graph is composed of forests with(or without) a line. Remember that every node in S_M connects to at least two nodes in A_N, thus there is a matching such that every node in S_M can connect to a distinct node in A_N. The final solution is to assign s_j to cluster with head v_i if one of the following two conditions holds: (1) $x_{ij}^* = 1$ (2)s_j is connected with v_i in the matching.

In this section, to make sure that we can guarantee the performance of the above job-scheduling based

DOI: 10.1002/wcm

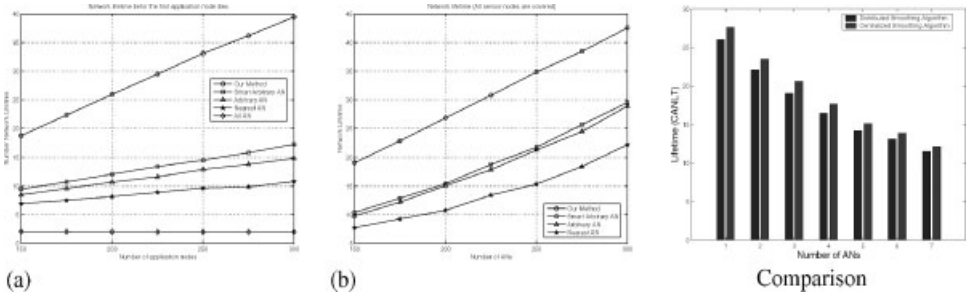


Fig. 2. Comparison of lifetime for different methods. (a) CANLT and (b) FCLT.

approach, we add one more requirement for the power consumption function p_i. We assume that the marginal cost of $p_i(x)$ is not increasing, that is, for $x_1 \geq x_2$, $p_i(x_1 + \delta) - p_i(x_1) \leq p_i(x_2 + \delta) - p_i(x_2)$. This assumption is almost universally satisfied. If this assumption is not satisfied, we can construct examples to show that the above approach (based on job scheduling) cannot provide any theoretical performance guarantees, although its practical performance may still be good.

Theorem 9. *Our job scheduling based method produces a cluster formation such that the lifetime of the WSN is at least* $\frac{1}{2}$ *of the maximum lifetime of the WSN.*

4. Performance Studies

We mainly study the case with heterogeneous ANs and homogeneous SNs. We randomly placed 2000 SNs in a 800 ft $\times$ 800 ft square region, the transmission range of each SN is set to 50 ft and the sensing range is set to 10 ft. Then we put a different number of ANs, from 150 to 300 (with incremental 25) and measured the network lifetime. In addition, the initial battery power of each SN is a random value between 100 and 200 U. A SN node is called an *alive sensor* if it has power remaining and has at least one alive AN in neighborhood. We compare our Algorithm 2 with other heuristics listed below: (1) [-*Nearest*] Each SN is assigned to the nearest AN. (2) [-*Arbitrary*] Each SN is *randomly* assigned to one of the neighboring ANs. (3) [-*Smart-Arbitrary*]: The probability of a SN s_j assigned to a neighboring AN v_i is the ratio of the remaining power of v_i over the total remaining power of all neighboring ANs of this SN s_j. (4) [-*All*] Here, each SN is assigned to *all* the ANs that are inside the SN's transmission range. This is clearly the worst method. Thus, we will not compare with this method in most simulations.

4.1. Lifetime

We compare the lifetime of four different methods under two different definitions of lifetimes: CANLT, FCLT. Figure 2a, b show the lifetime of different assignment methods under lifetime definition CANLT, FCLT, respectively. We generate 100 random WSNs and all results are the average over the performance of these 100 WSNs.

As can be seen, the network lifetime increases almost linearly with the number of ANs available initially for all methods, except the simplest ALL approach that does not perform any logic cluster at all. A striking observation is that, as we expected, our smoothing based method outperforms all other tree methods under all four definitions of lifetimes regardless of the density of the ANs. In all simulations, we found that our method generally outperforms the other methods by almost 100%. In other words, the network lifetime is almost *doubled* when our method is used to form the cluster.

We also compare the performance of the centralized smoothing algorithm 2 (CSA) and localized smoothing algorithm 4 (LSA). We fixed the number of the ANs to 50 and varies the number of SNs from 200 to 500. Figure 3c shows difference of the lifetime (CANLT) between CSA and LSA, and it is not difficult to observe that the lifetime of LSA and CSA only differs about 5% to 8%. This corroborates our theoretical analysis and we will only compare the lifetime of CSA with other four methods afterwards.

4.2. Load Balancing

As mentioned in Subsection 2.2, for heterogeneous ANs case, ANs have different initial battery powers, and the objective of the Algorithm 2 is to assign less SNs to ANs that have lower remaining battery power and more SNs to ANs that have higher battery power. To see how good the load balancing of our algorithm is, we run simulation for the networks with 150 ANs till all ANs die. As can be seen in Figure 3, our algorithm

DOI: 10.1002/wcm

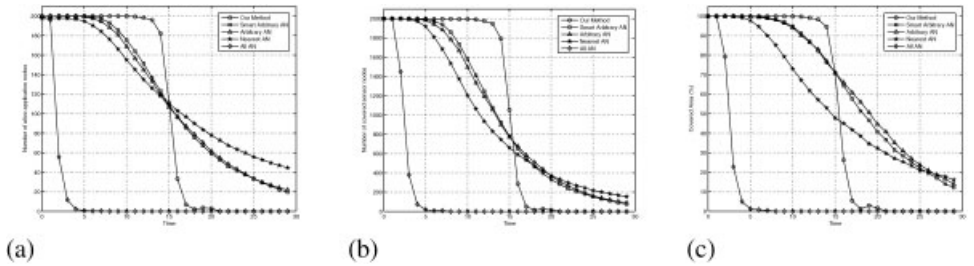


Fig. 3. Comparison for different methods. (a) Number of ANs alive, (b) number of SNs alive, and (c) area coverage ratio.

achieves a very good load balancing meaning that all ANs consume energy at a rate proportional to their initial battery power and then they all die together. The result for number of alive SNs and also the percentage of coverage area are basically the same as shown in Figure 3b, c.

5. Conclusion

In this paper, we studied how to organize the WSN to form logic clusters to maximize the lifetime of the networks. We also showed that it is NP-hard to find the optimum cluster formation. Our theoretical results are corroborated by extensive simulation studies. Our simulations show that our algorithms actually perform very well.

References

1. Baker DJ, Ephremides A. The architectural organization of a mobile radio network via a distributed algorithm. *IEEE Transactions on Communications* 1981; **29**(11): 1694–1701.
2. Parekh AK. Selecting routers in ad-hoc wireless networks. In *Proceedings of ITS*, 1994.
3. Bandyopadhyay S, Coyle E. An energy efficient hierarchical clustering algorithm for wireless sensor networks. In *Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies (Infocom)*, 2003.
4. Cai L, Shi Y, Pan J, Hou YT, Shen SX. Topology control for wireless sensor networks. In *Proceedings of the 9th Annual International Conference on Mobile Computing and Networking*. ACM Press, 2003; pp. 286–299.
5. Cai L, Shi Y, Pan J, Hou YT, Shen SX. Optimal base-station locations in two-tiered wireless sensor networks. *IEEE Transactions on Mobile Computing* 2005; **4**(5): 458–473.
6. Wieselthier J, Ephremides A, Baker DJ. A design concept for reliable mobile radio networks with frequency hopping signaling. In *Proceedings of IEEE* 1987; **75**: 56–73.
7. Parekh AK. Selecting routers in ad-hoc wireless networks. In *Proceeding ITS*, 1994.
8. Lin CR, Gerla M. Adaptive clustering for mobile wireless networks. *IEEE Journal on Selected Areas in Communications* 1997; **15**: 1265–1275.
9. Basagni S. Distributed clustering for ad hoc networks. In *Proceedings of the 1999 International Symposium on Parallel Architectures, Algorithms and Networks (ISPAN'99)*. IEEE Computer Society, 1999; p. 310.
10. Hopcroft JE, Karp RM. $n^{5/2}$ algorithm for maximum matchings in bipartite graphs. *SIAM Journal on Computing* 1973; **2**: 225–231.
11. Graham R. Bounds for multiprocessing timing anomalies. *SIAM Journal on Applied Mathematics* 1969; **17**: 416–429.
12. Hochbaum DS, Shmoys DB. Using dual approximation algorithms for scheduling problems theoretical and practical results. *Journal of the ACM* 1987; **34**(1): 144–162.
13. Tardos E, Lenstra JK, Shmoys DB. Approximation algorithms for scheduling unrelated parallel machines. *Mathematical Programming* 1990; **46**(3): 259–271.
14. Jansen K, Porkolab L. Improved approximation schemes for scheduling unrelated parallel machines. In *Proceedings of the Thirty-first Annual ACM Symposium on Theory of Computing*. ACM Press, 1999; pp. 408–417.
15. Shmoys DB, Tardos E. An approximation algorithm for the generalized assignment problem. *Mathematical Programming* 1993; **62**(3): 461–474.

Authors' Biographies

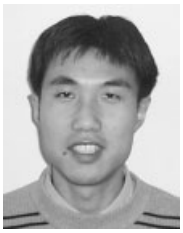
Dr Wen-Zhan Song has been an Assistant Professor of Computer Science at Washington State University Vancouver since 2005. He received Ph.D. (2005) degree from Illinois Institute of Technology, and B.S. (1997) and M.S. (2000) from Nanjing University of Science and Technology. His current research interest is mainly focus on protocol and algorithm design in distributed systems including wireless networks, sensor networks, and Peer-to-Peer overlay networks. He is currently leading a NASA research project on sensor webs. He is a member of the *IEEE*, *ACM* and *AGU*.

Dr WeiZhao Wang is a Research Engineer at Google Inc. He received his Ph.D. degree in Computer Science from Illinois Institute of Technology in 2006, and B.S. and M.S. degree in computer science from Shanghai Jiaotong University, China, in 1999 and 2002. His current research interests include wireless networks, game theory,

DOI: 10.1002/wcm

algorithm design, and next generation Internet. He is a member of the *IEEE* and *ACM SIGACT*.

Kousha Moaveni-Nejad received the Bachelor in Computer SoftEngineering from Sharif University of Technology, Tehran/Iran, in 1997. He joined the Department of Computer Science of Illinois Institute of Technology in 2000, as a M.S. student and received the Masters degree in 2002. He then continued his study as a Ph.D. student at Illinois Institute of Technology and his current research interests include wireless ad hoc networks, computationalgeometry, algorithm design, and mobile computing.



Dr Xiang-Yang Li is an Associate Professor of Computer Science at the Illinois Institute of Technology. He hold M.S. (2000) and Ph.D. (2001) degree at Computer Science from University of Illinois at Urbana-Champaign. He received Bachelor degree at Computer Science and Bachelor degree at Business Management from Tsinghua University, P.R. China in 1995. His research interests span the wireless ad hoc networks, non-cooperative computing, computational geometry, cryptography and network security, and optical networks. He served as a guest-editor of special issue of *ACM MONET*, *IEEE JSAC*. He is a Member of the *ACM* and *IEEE*.

DOI: 10.1002/wcm