Wireless Netw
DOI 10.1007/s11276-009-0207-9

# Approximation algorithm for minimal convergecast time problem in wireless sensor networks

**Weiping Shang · Pengjun Wan · Xiaodong Hu**



**Abstract** In this paper we consider how to collect data from sensors deployed in the Euclidean plane in a time-efficient way. We assume that all sensors could adjust their transmission ranges and aggregate data received from other sensors. We adopt a collision-free transmission model using proper schedules for data transmission. We study the problem of finding the schedule under which data from all sensors could be transmitted to the data sink in the minimal time. We propose an approximation algorithm for this NP-hard problem whose performance ratio is bounded by a constant. This significantly improves the existing approximation algorithm that does not have a constant performance ratio.

**Keywords** Convergecast · Latency · Wireless sensor networks · Approximation algorithm

## 1 Introduction

A Wireless sensor network (WSN) consists of a large number of small-sized and low-powered sensor devices spreading over a geographical area and a sink node, also called *base station*, from which end users can access sensed data in the network. All sensor devices are capable of sensing, processing data, and communicating with each other by means of a wireless ad hoc network. A wide range of tasks can be performed by these tiny devices, such as condition-based maintenance and the monitoring of a large area with respect to some given physical quantity, e.g., temperature, humidity, gravity and seismic information. One of the major communication operations in a WSN is to extract information/data from the sensed field and send them to the sink with low latency.

A communication session in a WSN is achieved either through a single-hop transmission if the communication parties are close enough, or through relaying by intermediate nodes otherwise. Depending on its transmission range, a node can reach all other nodes located within the range, and any node within its transmission range is considered to be its *neighbor*. However, when two or more sensors send their data to a common neighbor at the same time, the data *collide* at the common neighbor so that the neighbor will not receive data from any senders. In other words, a node can receive data from a sender only when no other node within its transmission range sends data at the same time (even if the data is supposed to be sent to some other nodes). Many methods were proposed to guarantee collision-free transmission such as using antenna [4] or multichannel [10].

One of the major concerns for data transmission in a WSN is energy-efficiency. In many applications, sensor nodes are deployed in a remote or dangerous area in which case servicing a node such as replacing or recharging batteries may not be possible. Thus the lifetime of a sensor node is heavily determined by its battery life. In this paper we adopt the following model to ensure energy-efficient transmission.

W. Shang
Department of Mathematics, Zhengzhou University, 450052 Zhengzhou, China
e-mail: shangweiping@gs.zzu.edu.cn

P. Wan
Department of Computer Science, Illinois Institute of Technology, Chicago, IL 60616, USA
e-mail: wan@cs.iit.edu

X. Hu (✉)
Institute of Applied Mathematics, Chinese Academy of Sciences, 100190 Beijing, China
e-mail: xdhu@amss.ac.cn

Published online: 11 September 2009

1. Each node is able to adjust its transmission range so that it may lower its transmission power if the destination node is very close to it.
2. A centralized schedule can set the time for data transmission for every sensor node so that no collision will occur.
3. Every sensor node has capability to aggregate data. This means that data can be merged all the way to the base station. Upon receiving data from other sensor nodes, a node fuses it with its own data (possibly null), and sends the aggregated data to the sink or some other node.

Note that the above assumptions (2, 3) guarantee that each node only needs to send its data once.

Another major concern for data transmission in a WSN is time-efficiency. In some applications of WSNs (such as battlefield communications, emergency disaster relief and environment monitoring), all requested or sensed data should be periodically delivered to the sink node within a certain period of time from the moment they are requested or sensed (after that data may be useless).

In this paper we study how to find a transmission range assignment and a data transmission schedule such that sensed data at all nodes could be aggregated and sent, known as *convergecast*, to the sink in the minimal number of rounds. This is called *Minimal Convergecast Time* (MCT) problem and proved NP-hard [3].

The remainder of this paper is organized as follows. In Sect. 2 we will first specify the network model and formalize MCT problem, and then present some related work. In Sect. 3 we propose an approximation algorithm for the problem with a constant performance ratio, that is, the latency of this algorithm is at most a constant times that of minimal latency for any problem instance. In Sect. 4 we conclude the paper with some remarks on future research.

## 2 Problem specification and related work

### 2.1 Problem specification

The data communication model in the paper is based on the data transmission in WSNs via aggregation for weather or environment monitoring. In such applications, we may be only interested in the highest temperature in a sensed field (for issuing fire-alarm warning), data kept at each node could just contain the local temperature and its location. When a node receives data from another node, it first compares the highest temperature stored in the data with the local temperature, and then chooses the higher one and put it into the current data along with the corresponding location. When a node has received and aggregated all data which are set to be sent to it, it sends the aggregated data to another node or the data sink directly.

Considering the above specified applications of WSNs, we assume that the underlying WSN consists of $n$ sensor nodes deployed on the Euclidean plane and it is static (all nodes cannot move). All nodes have data of the same size, which need to be sent to the data sink. We assume that the size of resulting data after aggregation remains unchanged and all nodes could finish (aggregated) data transmission in one time round of unit length. In other words, if a node has received one packet from its neighbor before its scheduled transmission time, then it can merge this packet and send the packet of merged data later. Moreover, data communication is deterministic and proceeds in synchronous rounds controlled by a global clock. In each time round,

- each node can either send data or receive data but cannot do both;
- each node can receive data from at most one of its neighbors;
- data packet sent by any sender reaches simultaneously all its neighbors;
- any node can receive data only if exactly one of its neighbors sends data.

In this paper we adopt the collision-free communication in WSNs which is guaranteed by proper schedules of data transmission. A *convergecast schedule* is denoted by a sequence $\{(S_1,R_1),(S_2,R_2), \ldots, (S_l,R_l)\}$ such that $S_i$ (resp. $R_i$) is the set of senders (resp. receivers) in the $i$-th round, $i = 1, 2, \ldots, l$, where $\{S_1, S_2, \ldots, S_l\}$ is a partition of the given set of $n$ nodes, which means that each node just sends data once, and $R_l = \{s\}$, which means that at the last round (i.e., $l$-th round) data should be sent to the sink. Note that every $(S_i,R_i)$ gives implicitly an one-to-one correspondence between $S_i$ and $R_i$ in a way that $v \in S_i$ corresponds to its receiver in $R_i$ which is the only neighbor of $v$ in $R_i$. The value $l$ is called the *convergecast latency* of the schedule.

We assume that each node is able to adjust its transmission range within (0,1]. As usual (e.g., [2]), we also assume that each sensor node knows its geometric position in the network, which is considered the unique ID of the sensor. Moreover, the sink has global knowledge of IDs of all sensors in the WSN. When it needs some data of particular interests at some sensor nodes, it informs those nodes (by multicasting) of the schedule $\{(S_1,R_1), \ldots, (S_l,R_l)\}$ which may be represented by IDs of senders and receivers. Upon receiving the request, sensor nodes will send their data or receive data from others as specified in the schedule.

A transmission range assignment can be represented by a directed graph $G = (V,E)$, called a *transmission graph*. Vertex-set $V$ is the set of given nodes, and there is an arc $(u,v) \in E$ from $u \in V$ to $v \in V$ if $v$ is in the transmission range of $u$. A *transmission range assignment* determines the power level chosen by each node in a certain round. Under an assignment algorithm, each node could adjust its transmission range within (0,1], but if we have determined its range in a certain round, it can not be changed any more. Given a transmission range assignment, a convergecast schedule can be represented by a directed tree $T$ of $G$ rooted at data sink $s$. The sink $s$ has a number of children, which equals the *in-degree* of $s$, that need to send their aggregated data to it one by one, and each of those children may have a number of its own children, which are the grandchildren of $s$, that need to send their aggregated data to it sequentially. Two grandchildren of $s$ can send their data to their parents, respectively as long as no collision occurs. Note that $T$ specifies where a node should send its data. Moreover, the latency of schedule corresponding to $T$ is no less than the depth of $T$, denoted by $d_T$, which is the maximal number of edges in paths from leaves towards root $s$. In fact, it can be determined as follows: Start data transmission from the leaf-nodes of current $T$, and then remove those leaf-nodes after their data are sent out, after that repeat these operations in the next round.

MCT problem is equivalent to computing a transmission graph and finding a directed tree that corresponds to a schedule of minimal latency. This problem is NP-hard even if the transmission graph is a unit disk graph [3].

### 2.2 Related work

Broadcast and convergecast are two most fundamental and useful operations in WSNs. Broadcast algorithms have been studied extensively most of them assuming fixed transmission ranges. Convergecast, which is also called *data aggregation*, by comparison, is a relatively new area but its importance cannot be overemphasized.

Regarding convergecast algorithms, Annamalai et al. [1] designed a heuristic method for both broadcast and convergecast for time-efficiency. The convergecast tree constructed in their algorithm can be used for broadcast as well. Upadhyayula et al. [11] designed another heuristic algorithm aiming at reducing energy and latency of convergecast. The performance evaluation of both algorithms were carried out through simulations (their theoretically guaranteed performances were not obtained). Kesselman and Kowalski [8] designed a distributed algorithm using a randomization technique that has a expected latency bound of $O(\log n)$ and consumes at most $O(n\log n)$ times of the minimum energy, where $n$ is the number of sensor nodes in a WSN and each node can adjust its transmission range.

For the practical study on data aggregation, especially about the Media Access Control (MAC) layer, Huang and Zhang [6] studied issues of packet loss reliability in data aggregation. Zhang et al. [13] addressed the bursty convergecast in real-time applications and focused on improving channel utilization and reducing retransmission incurred channel contention. Krishnamachari et al. [9] considered a general case of data aggregation where data from only a subset of nodes need to be convergecasted to the base station. Intanagonwiwat et al. [7] investigated how to increase the amount of path sharing and reduce energy consumption by evaluating the impact of greedy aggregation.

Closely related to our work, Yu et al. [12] considered the problem of scheduling data transmission and tradeoffing energy consumption and latency in WSNs. Their work focused on how to reduce sensor nodes energy dissipation subject to some latency constraints. Recently, Chen et al. [3] studied the minimal convergecast time problem with all sensor nodes having unit transmission range. They proved this problem is NP-hard and proposed a $(\Delta - 1)$-approximation algorithm, where $\Delta$ is the maximum degree of the transmission graph. More recently, Huang et al. [5] consider the same problem and designed an algorithm with latency $(27R + \Delta - 22)$, where $R$ is the network radius. If $\Delta$ is large in the network, their algorithms have bigger latency, and have no constant approximation performance. In order to save energy, in this paper we assume that each node could adjust its transmission range and make an improvement by proposing a new approximation algorithm with a constant performance ratio.

## 3 Approximation algorithm

Our convergecast algorithm uses a simple greedy method that schedules the data transmission from nodes in a subset $X \subset V$ to another subset $Y \subset V$, where each node in $X$ has at least one neighbor in $Y$ and $X \cap Y = \emptyset$. It works as follows: Initially, set $i$ to be zero. In each iteration, set $i$ to be $i + 1$; Compute a maximal (under containment) subset $S_i \subset X$ in an incremental way such that all nodes in $S_i$ can send their data to some nodes in $R_i \subset Y$ without causing conflict, where, initially, $S_i$ and $R_i$ are both set to be empty sets. $S_i$ is a maximal subset in current $X$, which means each node in $X \backslash S_i$ has at least one neighbor in $R_i$ (that is, we can not add any more nodes into $S_i$). In fact, we may choose a node $y \in Y$ such that it has the largest number of neighbors in $X$, and consider one of them, say $x$, and then let $x$ send its data to $y$ if it does not conflict previously scheduled transmission in $S_i$; After that remove $x$ from $X$ and put $x$ into $S_i$ while $y$ into $R_i$. Repeat the iteration with current $X$ being set to be $X \backslash S_i$ until $X$ is an empty set. Suppose the algorithm terminates after $k$ rounds. $X,Y)-$*schedule* is a schedule $\{(S_1,R_1), (S_2,R_2), \ldots, (S_k,R_k)\}$ such that $S_i$ (resp. $R_i$) is the set of senders (resp. receivers) in the $i$-th round, $i = 1, 2, \ldots, k$.

**Lemma 1** *Let $\Delta_X$ be the maximal number of nodes in X that share a common neighbor. Then under(X,Y)-schedule all nodes in X can finish data transmission in time rounds of* $\Delta_X$.

*Proof* We prove the lemma by mathematical induction on $\Delta_X$. When $\Delta_X = 1$, every node in $Y$ has (at most) one neighbor in $X$, so the lemma is obviously true. Now suppose that the lemma is true for $\Delta_X = k \geq 1$ and consider the case of $\Delta_X = k + 1$. By the induction hypothesis, it suffices to show that after the 1-st round, $\Delta_X \leq k$. Suppose, by contradiction, that after the 1-st round, $\Delta_X = k + 1$, that is, there exists a node $y \in Y$ that has $(k + 1)$ neighbors in $X$, say $x_1, x_2, \ldots, x_k, x_{k+1}$ and $y$ could not receive data from any of them. Note that this case occurs only when there exists a node $x \in X$ that is $y'$s neighbor in the original $X$ and set to send its data to another node $y' \in Y$ in the 1-st round. However, this is impossible since after the 1-st round $x$ is removed from $X$ and it is not $y'$s neighbor in the current $X$, $\Delta_X \geq k + 2$. A contradiction! The lemma is then proved. □

When all nodes are assigned transmission range of 1, we will obtain a special transmission graph $G(V,E)$, called a *unit disk graph*, in which there is an edge between two nodes if and only if the distance between them is at most 1. Denote by $T_{\mathrm{BFS}}$ the Breadth-First-Search tree rooted at sink $s$ in $G$. For each $0 \leq i \leq d(T_{\mathrm{BFS}})$, denote each level of $T_{\mathrm{BFS}}$ by $L_i(T_{\mathrm{BFS}}) = \{v: dist_{\mathrm{BFS}}(s,v) = i\}$, where distance $dist_{\mathrm{BFS}}(s,v)$ is defined as the number of arcs in the path of $T_{\mathrm{BFS}}$ from $v$ to $s$. Clearly, the set $L_i(T_{\mathrm{BFS}})$ for $0 \leq i \leq d(T_{\mathrm{BFS}})$ compose a partition of $V$. Fig. 1 shows such an example with $d(T_{\mathrm{BFS}}) = 3$. In the following we will outline our convergecast algorithm that is based on $T_{\mathrm{BFS}}$.

First the algorithm sorts all nodes in $V$ in the increasing order of $dist_{\mathrm{BFS}}(s,\cdot)$, and then compute a *Maximal Independent Set* (MIS) by using the greedy First-Fit algorithm in such an order. An MIS is such a subset $W$ of $V$ that no edge in $E$ is between a pair of two nodes in $W$ and no superset of $W$ has this property. Given an order of all nodes in $V$, each time the algorithm removes the first node in the current order and puts it into the set if it is not adjacent with any node in the set. Note that the selected nodes compose a dominating set $D$, so they are called *dominators*. A subset $W$ of $V$ is a *dominating set* if every node in $V$ is either in $W$ or adjacent with at least one node in $W$. It is further called a *connected dominating set* if the subgraph $G[W]$ induced by set $W$ is connected. In the example of Fig. 1(a), $D$ consists of the sink node and five black nodes, but it is not connected.

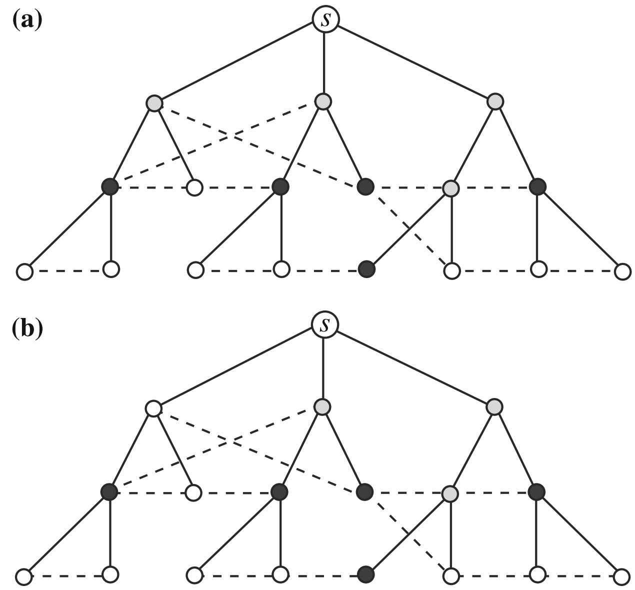


**Fig. 1** $T_{\mathrm{BFS}}$ consists of solid links while other links in $G(V,E)$ are *dashed*

Note that by the way we produce the MIS, $D$ contains sink node $s$ but not any node in $L_1(T_{\mathrm{BFS}})$, so $D$ is not a connected dominating set. As all dominators in $D$ can be connected through their parents, we call them *connectors* and denote the set of connectors by $C_0$. In fact, we can find a minimal (under containment) subset $C \subset C_0$ of nodes that connect all dominators in $D$ in a greedy way as follows: Each time choose a node in $C_0$ that can join the most number of dominators, and then repeat this operation until all dominators are connected. The minimality of connector set $C$ will be used in the analysis of our convergecast algorithm. In the example of Fig. 1(a), $C_0$ consists of those four grey nodes while $C$ contains only three of them in Fig. 1(b).

Let $V_0 \equiv C \cup D$, then $V_0$ is a connected dominating set. In order to guarantee that data in all nodes could be sent to sink $s$, we set transmission ranges of all nodes in $V_0$ to be one, i.e., $r(v) = 1$ for every $v \in V_0$. So we just need to consider how to set the transmission ranges of other nodes in $V \backslash V_0$ and transmission time of all nodes in $V$. This is the major part of our convergecast algorithm which consists of two stages. In the first stage, data are sent from nodes in $V \backslash V_0$ to nodes in $V_0$. In the second stage, data are sent from nodes in $V_0$ to sink $s$. In the following two subsections, we will describe each stage in details.

### 3.1 The first stage

Denote by $\|uv\|$ the Euclidean distance between any two nodes $u$ and $v$ in $V$, and denote by $(u,v)$ the arc from $u$ to $v$ whose length is $\|uv\|$. In addition, denote by $D_X(u,r)$ the set of nodes in $X$ located in the disk of radius $r$ centered at $u$, i.e., $D_X(u,r) = \{v \in X\colon \|uv\| \le r\}$. For subset $X \subseteq V$, let $G[X] = (X,E)$, where $(u,v) \in E$ if the Euclidean distance between $u$ and $v$ is at most one.

The algorithm at this stage proceeds in phases (refer to the description below). Each phase corresponds to an iteration of the outermost while-loop (2–21). The purpose of this outer loop is to gradually reduce the number of nodes $v_i$ in $S$ that have not been set to send their data. Initially, $S$ contains all nodes in $V \backslash V_0$. Once a node is set to send its data in the current $t$-th round it will be put into $S_t$ at Steps 11–13; After that it will be discarded from $S$ at Step 20 since we do not need to consider it in the subsequent rounds (Fig. 2).

At the beginning of round $t$, denote by $F_t$ the set of the shortest arcs $(u,v)'$s in current $G[S]$ such that $u$ may be set to send its data to $v$, and denote by $E_t$ be the set of selected arcs $(u,v)'$s in $F_t$ such that $u$ is set to send its data to $v$ in the $t$-th round. The selection of arcs in $E_t$ from $F_t$ is done in a greedy way for energy-efficiency as follows: Start from the shortest arc $(u, v)$ in $F_t$ at Step 11. Note that at Step 11, a tie will occur if more than two arcs in $F_t$ have the minimal length. In this case we may choose the arc $(u,v)$ with minimal $x$-coordinate $x(u)$. If a node $w$ in the current $S$ has no neighbor in $S$, then $w$ is put into a set $X_0$. It is clearly that the distance between any two nodes in $X_0$ is more than one.

Whenever $(u,v)$ is selected, for every arc $(u',v') \in F_t$, if $u'$ sends its data to $v'$ but $v'$ can not receive it or $v$ can not receive data from $u$ because of transmission conflict, $(u', v')$ must be removed from $F_t$ at Steps 14–18.

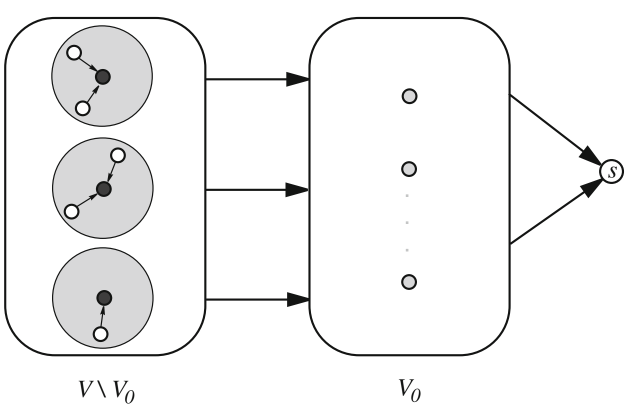


**Fig. 2** The basic idea of two-stage approximation algorithm where $X_0 \subset V \backslash V_0$ consists of those black nodes

```
1.  S := V \ V0; X0 := ∅; t := 1.
2.  while |S| ≥ 1 do begin
3.      Ft := ∅.
4.      for u ∈ S do begin
5.          Choose v ∈ S \ {u} with minimal ||uv||.
6.          if ||uv|| ≤ 1 then Ft := Ft ⋃{(u, v)}
7.                        otherwise S := S \ {u}; X0 := X0 ⋃{u};
8.      end-for                                  r(u) := 1.
9.      St := ∅; Rt := ∅; Et := ∅.
10.     while Ft ≠ ∅ do begin
11.         Choose the arc (u, v) of minimal length in Ft;
12.         Et := Et ⋃{(u, v)}; Ft := Ft \ {(u, v)};
13.         St := St ⋃{u}; Rt := Rt ⋃{v}; r(u) := ||uv||.
14.         for (u′, v′) ∈ Ft do begin
15.             if v′ ∈ DX(u, ||uv||) or v ∈ DX(u′, ||u′v′||) then
16.                 Ft := Ft \ {(u′, v′)}.
17.             end-if
18.         end-for
19.     end-while
20.     S := S \ St; t := t + 1.
21. end-while
22. return r(u) for each u ∈ {V \ V0} \ X0; St, Rt for each t; X0.
```

In the following we will prove the correctness of *Algorithm A* and estimate how many rounds are needed to finish data transmission from $V \backslash V_0$ to $V_0$. For this purpose, we need to show some lemmas first.

**Lemma 2** *For each time roundt, all nodes in $S_t$ can send their data to their intended receivers without causing collision.*

*Proof* We only need to show that all scheduled transmissions along arcs in $E_t$ could be received successfully by the intended receivers. Consider any two arcs $(u,v)$ and $(u',v')$ in $E_t$. Suppose that $(u,v)$ is added to $E_t$ prior to $(u',v')$. By the rule of Steps 14–18, we have $v' \notin D(u, ||uv||)$ and $v \notin D(u', ||u'v'||)$. Hence, the intended receivers can receive the data sent along these two arcs in the same time round. □

**Lemma 3** *When Algorithm A terminates, all nodes in $V \backslash V_0$ have sent their data to $X_0$, and $X_0$ is an independent set.*

*Proof* Suppose that a node $u$ is not set to send data at the $t$-th round. By Lemma 2, it will be considered at the beginning of the next round or it will be put into $X_0$. Hence, when *Algorithm A* terminates, every node in $V \backslash V_0$ has either sent its data to some node in $X_0$ or put into $X_0$. Moreover, a node $v$ is put into $X_0$ at Step 7 only when the distance between it and any other node in $S$ is more than one. Thus $X_0$ consists of nodes that can not reach each other with transmission range one, that means that $X_0$ is an independent set. □

**Lemma 4** *At Step 20, $|E_t| \ge |F_t|/22$ for each $t$.*

*Proof* To prove the lemma it suffices to show that once an arc $(u,v) \in F_t$ is selected at Step 11, at most 21 arcs $(u',v')$ could be removed from $F_t$ at Steps 14–18 (since $u'$ and $u$

can not send their data at the same time to $v'$ and $v$, respectively).

Recall that arc $(u,v) \in F_t$ is selected because it has the shortest length (and minimal $x$-coordinate if there are at least two arcs in $F_t$ that have the shortest length). Hence if there exists a node $w$ that satisfies $\|wu\| = \|uv\|$, then it must be on the boundary of the right half of $D(u, \|uv\|)$. This implies that there are at most two such nodes $w_1$ and $w_2$ since otherwise, there must exist an arc in $F_t$ whose length is shorter than $\|uv\|$, that contradicts the way of selecting $(u,v)$ at Step 11 (Fig. 3a).

Moreover, for each arc $(u,v) \in F_t$, because $v$ is the nearest neighbor of $u$ in $S$, there are at most six arcs in $F_t$ towards $v$ since otherwise, there are at least 7 nodes that are closer to $v$ than to each other, and there must be a triangle formed by two of them $u$, $w$ and $v$ in which the angle $\angle uvw$ is less than $\pi/3$. Thus the biggest angle in this triangle, say $\angle uwv$, is great than $\pi/3$. However, in this case the length of arc $(u,w) \in F_t$ is shorter than that of arc $(u,v) \in F_t$, contradicting the way of selecting $(u,v)$ at Step 5 (Fig. 3 b).

Therefore, incident to four nodes $u$,$v$, $w_1$ and $w_2$, there are at most 24 arcs in $F_t$ which could be removed from $F_t$ at Steps 14–18. Note, however, that among them three arcs $(u,w_1)$, $(u, w_2)$ and $(u,v)$ are counted twice. So in the total, at most 21 arcs $(u', v')$ could be removed from $F_t$ at Steps 14–18. The proof is then finished. □

**Lemma 5** *Data transmission scheduled by Algorithm A can finish in* $15\log_2|V\backslash V_0|$ *rounds.*

*Proof* Note that at the beginning of the algorithm, $S = V\backslash V_0$, and then at each iteration of the outmost while-loop (Steps 2–21), we have $|F_t| = |S\backslash X_0|$ at Step 9. By Lemma 4, we deduce that at least $|S\backslash X_0|/22$ nodes can send their data in the same round. Moreover, under the schedule of the algorithm, nodes in $X_0$ will not send their data, and $(1-1/22)^{15} < 1/2$. Thus by repeatedly applying Lemma 4, we deduce that after at most $15\log_2|V\backslash V_0|$ rounds, we will obtain $|S\backslash X_0| < 1$, which means that *Algorithm A* schedules all data transmission and terminates. □

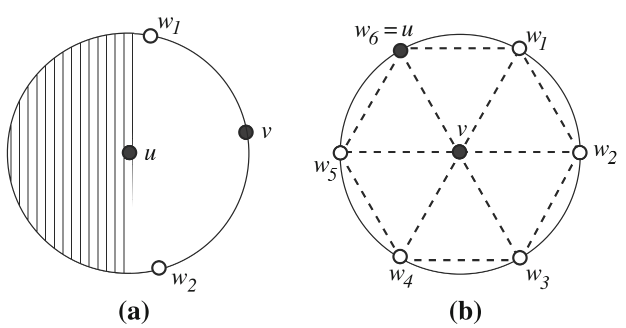


**Fig. 3** For the proof of Lemma 4

**Lemma 6** *Data transmission from all nodes in* $V\backslash V_0$ *can finish in*$(15\log_2|V\backslash V_0| + 5)$ *rounds*

*Proof* By Lemma 5, under the schedule of *Algorithm A* all nodes in $(V\backslash V_0)\backslash X_0$ can finish data transmission within $15\log_2|V\backslash V_0|$ rounds, so we just need to consider how to schedule data transmission from nodes in $X_0$. By Lemma 3, $X_0$ is an independent set and the transmission ranges of all nodes in $X_0$ are set to be 1 by *Algorithm A*. Thus each node in $V_0$ is adjacent with at most five nodes in $X_0$ (otherwise there must exist two nodes in $X_0$ such that the distance between them is less than 1). By Lemma 1, under $(X_0,V_0)$-schedule data transmission from all nodes in $X_0$ can finish in five rounds, that completes the proof. □

## 3.2 The second stage

The basic idea of our algorithm at the second stage is demonstrated in Fig.4. For each $0 \le i \le d(T_{\mathrm{BFS}})$, let $D_i$ and $C_i$ denote the sets of dominators and connectors in the $i$-th level, respectively, where $C_{d(T_{BFS})} = \emptyset$. For each level $i$, first schedule data transmissions from connectors in $C_i$ using $(C_i, D_i \bigcup D_{i-1})-$ schedule, and then schedule data transmissions from dominators in $D_i$ using $(D_i,C_{i-1})$-schedule.

**Lemma 7** *Each dominator in* $D_i \bigcup D_{i-1}$ *has at most 12 neighboring connectors in* $C_i$.

*Proof* For each node $v \in C_i$, let $p(v) = \{u$: $u \in D_{i+1}$ is adjacent with no nodes in $C_i$ except $v\}$. As $C$ is a minimal set of connectors, each node $v \in C_i$ has at least one neighbor $u \in p(v)$ otherwise $v$ should not be put into $C_i \subset C$. Now choose $u_v \in p(v)$, and let $P = \{u_v$: $v \in C_i\}$. Suppose that there is a dominator $x \in D_i \bigcup D_{i-1}$ and it has at least 13 neighboring connectors in $C_i$. Then there are at least 13 nodes in $P$ lying in the disk of radius 2 centered at $x$ (Fig. 5 a).

Let $u_v$ and $u_w$ be two nodes in $P$ satisfying $\angle u_v x u_w \le 2\pi/13$. Let $B(x)$, $B(u_v)$ and $B(u_w)$ be the unit disks centered at $x$, $u_v$ and $u_w$, respectively, $C(x)$ and $C(u_v)$ be the unit circles centered at $x$ and $u_v$, respectively (Fig. 5 b). Since $u_v$, $u_w \in D_{i+1}$, we have $\|u_v u_w\| > 1$. Let $y$ and $z$ be the two points where $C(x)$ meet $xu_v$ and $xu_w$ respectively. Assume $\|u_v z\| \le \|u_w y\|$, then we have $\|u_v z\| < 1$

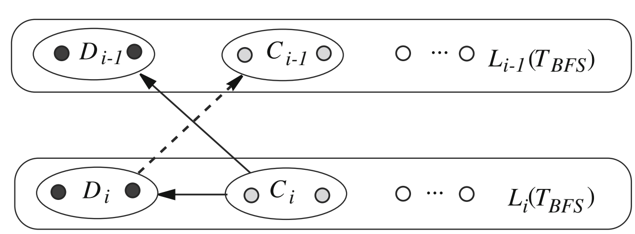


**Fig. 4** The basic idea of the algorithm at the second stage

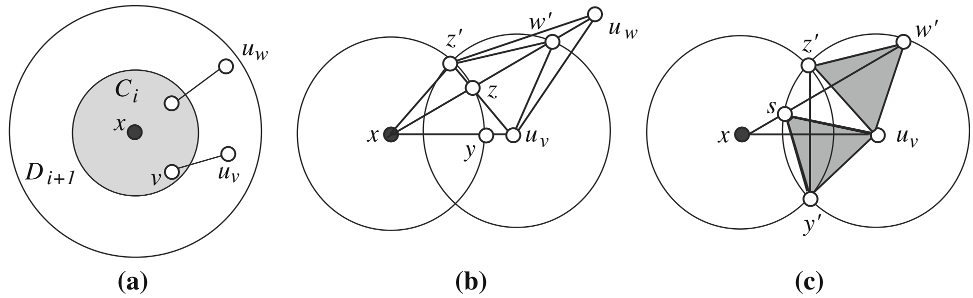


**Fig. 5** For the proof of Lemma 7

since $\angle u_v x u_w \leq 2\pi/13$ and $\|u_v u_w\| > 1$. Let $z'$ be the intersection point of the two circles $C(x)$ and $C(u_v)$ that lies in the same side of $xu_v$ as $u_w$. In the following we show that $\|u_w z'\| > 1$.

Note first that we can restrict $\angle u_v x u_w \leq \pi/6$ Since $\|u_w z'\|$ decreases and $\|u_w u_v\|$ increases while $xu_w$ is rotated away from $xu_v$. And then let $w'$ be the intersecting point of $u_w z$ and $C(u_v)$, $z'$ and $y'$ be two intersecting points of $C(x)$ and $C(u_v)$. In addition, let $s$ be the intersecting point of $C(u_v)$ and the line through $x$ that has a $\frac{\pi}{6}$-slope from line $xu_v$ and hits the segment $u_v z'$ (Fig. 5 c). We claim that both $\triangle u_v y' s$ and $\triangle u_v z' w'$ are equilateral. To see this, consider two isosceles triangles $\triangle z' u_v y'$ and $\triangle s u_v w'$. Note that their sides are one and their heights are $\|xu_v\|/2$, so they are identical. For any three points $a$,$b$,$c$ on a circle, denote by $\widehat{abc}$ the radian of arc with ends $a, c$ that traverses $b$. Then we have

$$\widehat{y'u_v s} = \widehat{xu_v y'} + \widehat{xu_v s} = \widehat{xu_v y'} + \widehat{u_v s w'} - \frac{\pi}{6}$$

$$= \widehat{xu_v y'} + \widehat{u_v y' z'} - \frac{\pi}{6} = \frac{\pi}{2} - \frac{\pi}{6} = \frac{\pi}{3}$$

Since $\widehat{z'u_v y'} = \widehat{su_v w'}$, we have $\widehat{y'u_v s} = \widehat{z'u_v w'}$, and then the claim follows.

Now the above claim yields $\|w'z'\| = 1$. As $\|u_w z'\| > \|w'z'\| = 1$, we have $B(x) \bigcap B(u_w) \subseteq B(u_v), w$ has two neighbors $u_v$ and $u_w$, contradicting the definition of $P$. The lemma is then proved. □

**Lemma 8** *Data transmission from all nodes in* $V_0$ to *sink node s can be finished in* $(16d(T_{\mathrm{BFS}})-12)$ *rounds.*

*Proof* For each level $i$ from $d(T_{\mathrm{BFS}})$ to 1, we first schedule data transmissions from connectors in $C_i$ using $(C_i, D_i \bigcup D_{i-1}) - schedule$, and then schedule data transmissions from dominators in $D_i$ using $(D_i,C_{i-1})$-schedule. By Lemma 7, we know that each dominator in $D_i \bigcup D_{i-1}$ has at most 12 neighboring connectors in $C_i$. Moreover, each connector in $C_{i-1}$ has at most four neighboring dominators in $D_i$ since each node is adjacent with at most 5 dominators (otherwise there exist two dominators $d_1$ and $d_2$ with $\|d_1 - d_2\| < 1$, this is impossible since each $D_i$ is an independent set), and at least one dominator is at $D_{i-1}$ or $D_{i-2}$. Hence the $(C_i, D_i \bigcup D_{i-1}) - schedule$ has latency at most 12, and $(D_i,C_{i-1})$-schedule has latency at most 4. Note also that there is no connector in the last level $d(T_{\mathrm{BFS}})$. Therefore, we obtain an upper bound of $(16d(T_{\mathrm{BFS}})-12)$ rounds on the latency of data transmission from all nodes in $V_0$ to sink node $s$, and the lemma is proved. □

Finally, we can prove the major result of this paper.

**Theorem 1** *There exists an approximation algorithm for minimal convergecast time problem with performance ratio no greater than 31.*

*Proof* It immediately follows from Lemma 6 and Lemma 8 that data transmission from all nodes in $V$ to sink node $s$ can finish in $(16d(T_{\mathrm{BFS}}) + 15\log_2|V|)$ rounds. Note that under the collision-free transmission model, each node can either send data or receive data but can not do both. Hence at each round at most half number of nodes that have not sent their data could finish data transmission, thus the latency of any convergecast schedule is at least $\log_2 |V|$. It is clear that $d(T_{\mathrm{BFS}})$ is also a lower bound. Therefore, we can schedule data transmission from all nodes in $V$ to sink node $s$ in such a way whose latency is at most 31 times that of the optimal schedule. □

## 4 Conclusion

In this paper we have proposed an approximation algorithm with a constant performance ratio for the minimal convergecast time problem in WSNs where sensors could adjust their transmission ranges. It is clear that the proposed approach could also be applied to the case of fixed transmission ranges studied by Chen et al. [3], who proposed an approximation algorithm with a nonconstant bounded performance ratio. Thus the obtained result in this paper makes a significant improvement on their work.

We also notice that the convergecast model adopted in this paper has its limitations since it is based on weather or environment monitoring networks. Thus some assumptions may not be satisfied in other applications. For examples, the availability of the global information of networks and the invariability of data sizes after aggregation. Thus to deal with those more complicated situations different approaches (e.g., distributed algorithms instead of centralized ones) need to be used and studied.

Although the proposed algorithm assigns each sensor node a proper transmission range instead of using the maximum transmission range in order to reduce the energy consumption, we do not use the total energy cost for convergecast as a metric to evaluate the performance of the proposed algorithm (we just use the latency). To design an algorithm for energy-efficient convergecast, we may first compute a convergecast tree of low energy cost (e.g., using the algorithm proposed in [1]). (In fact, a broadcast tree of low energy cost could also be used as a convergecast tree.) And then using the approach proposed in this paper we could schedule data transmission based on this tree (instead of BFS tree). However, it turns out to be very hard to design a convergecast algorithm that has a theoretically guaranteed performance in terms of both time-efficiency and energy-efficiency since the energy cost for data transmission between two nodes $u$ and $v$ is $\|uv\|^{\alpha}$ with $2 \leq \alpha \leq 4$, which is not a linear function of the distance between them (so it does not satisfy the triangle inequality). Hence this is worth further studying in future.

**Acknowledgments** The authors are deeply indebted to anonymous referees for their invaluable comments and suggestions which have greatly improved the presentation of this paper. This study was supported in part by the National Natural Science Foundation of China under Grant No. 10531070, 10771209, 10721101, and Chinese Academy of Sciences under Grant No. kjcx-yw-s7.

## References

1. Annamalai, V., Gupta, S.K.S., Schwiebert, L. (2003). On tree-based convergecasting in wireless sensor networks. *Proceedings of IEEE Wireless Communication and Networking Conference, 4*(1), 1942–1947.
2. Bulusu, N., Heidemann, J., Estrin, D. (2000). GPS-less low cost outdoor localization for very small devices. *IEEE Wireless Communications, 7*(5), 28–34.
3. Chen, X.-J., Hu, X.-D., Zhu, J.-M. (2005). Minimum data aggregation time problem in wireless sensor networks. *Lecture Notes in Computer Science ,3794*, 133–142.
4. Guo, S., Yang, O. (2006). Minimum-energy multicast in wireless ad hoc networks with adaptive antennas: MILP formulations and heuristic algorithms. *IEEE Transactions on Mobile Computing, 5*(4), 333–346.
5. Huang, C. H., Wan, P.-J., Vu, T., Li, Y., Yao, F. (2007). Nearly constant approximation for data aggregation scheduling in wireless sensor networks. In *Proceedings of the 26-rd IEEE conference on computer communications*.
6. Huang, Q., & Zhang, Y. (2004). Radial coordination for convergecast in wireless sensor networks. In *Proceedings of the 29-th annual IEEE international conference on local computer networks* (pp. 542–549).
7. Intanagonwiwat, C., Estrin, D., Govindan, R., Heidemann, J. (2002). Impact of network density on data aggregation in wireless sensor networks. In *Proceedings of 22nd international conference on distributed computing systems* (p. 457).
8. Kesselman, A., & Kowalski, D. (2005). Fast distributed algorithm for convergecast in ad hoc geometric radio networks. In *Proceedings of the 2-nd annual conference on wireless on demand network systems and serveices*.
9. Krishnamachari, B., Estrin, D., Wicker, S. B. (2002). The impact of data aggregation in wireless sensor networks. In *Proceedings of the 22-nd international conference on distributed computing systems* (pp. 575–578).
10. Nasipuri, A., Das, S.R. (2006). Performance of multichannel wireless ad hoc networks. *International Journal of Wireless and Mobile Computing, 1*(3/4), 191–203.
11. Upadhyayula, S., Annamalai, V., Gupta, S. K. S. (2003). A low-latency and energy-efficient algorithm for convergecast in wireless sensor networks. *Proceedings of IEEE Global Telecommunications Conference, 6*, 3523–3530.
12. Yu, Y., Krishnamachari, B., Prasanna, V. K. (2004). Energy-latency tradeoffs for data gathering in wireless sensor networks. In *Proceedings of the 23-rd IEEE conference on computer communications*.
13. Zhang, H., Arora, A., Choi, Y.-R., Gouda, M. G. (2005). Reliable bursty convergecast in wireless sensor networks. In *Proceedings of the 6-th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, 266–276.

## Author Biographies

**Weiping Shang** received the BS and MS degrees in Mathematics from Zhengzhou University, and her PhD degree in Operations Research and Cybernetics from Institute of Applied Mathematics, Chinese Academy of Sciences, in 2008.Currently, she is a lecture in the Department of Mathematics at Zhengzhou University. Her research interests include graph theory and wireless sensor networks.

**Pengjun Wan** received the BS degree from Tsinghua University, the MS degree from the Chinese Academy of Science, and the PhD degree from the University of Minnesota. He is currently a professor of computer science in the Department of Computer Science, Illinois Institute of Technology, Chicago. His research interests include wireless networks, and algorithm design and analysis.

**Xiaodong Hu** received his BS degree in Applied Mathematics from Tsinghua University, Beijing, China, in 1985, and his PhD degree in Operations Research and Cybernetics from Institute of Applied Mathematics, Chinese Academy of Sciences, in 1989. He was a postdoctor fellow at Rutgers Center for Operations Research, Rutgers University, in 1990–1991, a visiting associate professor at graduate school of information science, Japan Advanced Institute of Science and Technology, in 1993–1994, and a research fellow at Computer Science Department, City University of Hong Kong, in 1998–2000. Currently, he is a research professor at Institute of Applied Mathematics, Chinese Academy of Sciences. His research interests include combinatorial optimization and computer and communication networks. He is the member of American Mathematical Society (since 1992) and member of IEEE Computer Society (since 1997).