A Dynamic Routing Algorithm for Data-Aggregation Optimization in Event-Driven Wireless Sensor Networks

Yalin Nie[1] Sanyang Liu[2], Zhibin Chen[3], and Xiaogang Qi[2]
[1]School of Computer Science and Technology, Xidian University, Xi'an, Shanxi 710071, China
[2]School of Science, Xidian University, Xi'an, Shanxi 710071, China;
[3]School of Microelectronics, Xidian University, Xi'an, Shanxi 710071, China
Email: nieyalin111@163.com; {liusanyang, zhibinchen2013, qixiaogang2013}@126.com

***Abstract*—**In order to optimize the total energy consumption of data collection for wireless sensor networks, we study how to route data for improving data aggregation efficiency and propose a dynamic routing algorithm for data-aggregation optimization in event-driven wireless sensor networks. It clusters nodes within event areas in a distributed manner with low control overhead and establishes an approximate Steiner tree based on events aided by the sum of Euclidean distances from nodes to cluster heads. The algorithm does not only improve the data aggregation ratio but also decreases the control overhead for building and maintaining routing structure, achieving energy-efficient data routing and collection and promoting network performance eventually. Algorithm analysis and experiments show that the algorithm can effectively decrease the amount of transmissions for both data and control packet, prolonging network lifetime.

***Index Terms*—**wireless sensor networks, cluster, data aggregation, paths overlapping

I. Introduction

Wireless Sensor Networks (WSNs) comprise a large number of sensor nodes which can sense enviroment, process data and communicate through wireless telecommunication. With wide range of applications, WSNs can be used in many scenarios such as environmental monitoring, rescue/assistance systems (fight against forest fire, help disabled people and *et al*.), industrial process control, localization of services and users, traffic monitoring/control in urban/suburban areas, military/antiterrorism systems and so on.

The energy consumption for data transmission dominates in WSNs, dependent on the amount of in-network data transmission significantly. Data compression based in-network data aggregation [1], [2] can decrease the amount of in-network data transmission effectively and save energy, leading to longer network lifetime. For the applications of statistical query, like AVERAGE, SUM *et al*., no matter how much the amount of original data is, the size of aggregated data is fixed, which we call full-aggregation [3].

Manuscript received May 28, 2013; revised August 3, 2013.
This work was supported by the Youth Fund of Luoyang Institute of Science & Technology under Grant No. 2008QZ11.
Corresponding author email: liusanyang@126.com.
doi:10.12720/jcm.8.8.521-528

It is convenient to aggregate data to eliminate redundancy by clustering nodes into clusters in WSNs [4]-[7]. Clustering can also help extend the scalability of network. How to optimize routing structure through the combination of clustering and data aggregation is a popular topic in research field of WSNs.

There are two kinds of data collection in WSNs: periodic and event-driven [8]. In event-driven WSNs, if full-aggregation is applied while aggregating data, the routing optimization problem is equal to the Steiner Tree problem which connects all the nodes in event areas [9]. Studies have shown that searching a Steiner Tree connecting all the nodes in a subset from a graph is an NP-hard problem which can be solved approximately only by means of heuristic methods [10], [11], and several algorithms for building approximate Steiner Tree based on clusters for event-driven WSNs have been proposed recently [12]-[14].

For WSNs, the more approximate to the Steiner Tree the routing structure is, the better the data aggregation ratio is, and often the less the amount of in-network data transmission is. The overhead of building and maintaining an approximate Steiner Tree is usually significant. So, optimizing the approximation to the Steiner Tree under low control overhead is our purpose. Here we propose a novel Dynamic Routing Algorithm for data-aggregation optimization in event-driven WSNs (DRA). DRA clusters the nodes within event areas in a distributed manner, builds shortest paths from cluster heads to the Sink, and utilizes a simple heuristic strategy to increase the degree of path overlapping, leading to an approximate Steiner Tree constructed by low control overhead and efficient in-network data aggregation. Algorithm analysis and experiments show that DRA contributes low control overhead and better performance on data aggregation.

The rest of this paper is organized as follows. Section II presents the related works and our motivation. Section III elaborates the DRA algorithm. In section IV, we analyze the performance of DRA and present the results of experiments. Finally, Section V concludes the work.

II. Related Works

LEACH [4] is one of the early clustering algorithms. It divides the network time into rounds, chooses different p% nodes as cluster heads in each round, gets remaining nodes join the clusters according to the signal strength, and makes cluster heads send aggregated data to the Sink directly. LEACH can prolong network lifetime effectively, and many later clustering algorithms are proposed based on the basic framework of LEACH. Stanislava Soro *et al*. proposed CPCP [5] which is a clustering algorithm in consideration of network coverage. With the coverage as criterion, it chooses cluster heads and sleeps cluster members to get high coverage ratio while prolonging network lifetime. DACP [6] is an energy-efficient data aggregation protocol based on data prediction and clustering, which makes cluster heads decide when to send aggregated data according to the predicted data for improving the data aggregation ratio. Woo-Sung Jung *et al*. [7] proposed a state based clustering technique which can promote the data aggregation ratio while improving energy efficiency. DRINA [12] can find an approximate Steiner Tree based on clusters. Cluster heads build shortest paths to the existing Hop-Tree with greater overlap to improve data aggregation ratio. YEAST-CF、YEAST-FF and YEAST-BC [14] are strategies that can build approximate Steiner Trees regardless of the sequence of events for event-driven WSNs.

SPT and CNS [15] are typical routing protocols for WSNs. By SPT, any node that has detected event reports data to the Sink along the shortest path with opportunistic data aggregation occurring at the crossover nodes. CNS just chooses the nearest node to the Sink that has detected the event as the aggregation node. DST [16] can build a routing tree regardless of the sequence of events too and decrease the number of working nodes according to the requirement of correlation for different applications.

Information-Fusion-based Role Assignment (InFRA) [13] is a cluster based routing algorithm for better information fusion. It can achieve the self-organization clustering for event nodes and build a shortest path tree based on cluster heads and the Sink with greater path overlapping as an approximation to the Steiner Tree, resulting in good data aggregation efficiency. However, InFRA has the following disadvantages:

1) The control overhead consumed by the cluster head election process is large, especially for large scale event.
2) In order to make all nodes obtain the distance to cluster heads, each cluster head should flood its information to the whole network.

It is easy to see that the overhead of building and maintaining routing structure is large for InFRA, with poor network scalability. InFRA is also not suitable when events occur frequently and/or the duration of events is short.

For the above shortcomings of InFRA, we propose a Dynamic Routing Algorithm for data-aggregation optimization in event-driven WSNs (DRA) consisting of 4 phases: network initialization phase, clustering phase, route update phase and data transmission phase. DRA has low control overhead during the clustering phase, avoids cluster heads flooding information across the whole network by the help of Sink broadcasting cluster information with a power strong enough, and makes the shortest paths from cluster heads to the Sink overlap as soon as possible in order to gain an approximate Steiner Tree. Algorithm analysis and experiments confirm that DRA has better data aggregation ratio with low overhead for building and maintaining routing structure, reducing energy consumption and prolonging network lifetime.

III. A Dynamic Routing Algorithm for Data-Aggregation Optimization

In event-driven WSNs, data aggregation within event areas can decrease the amount of in-network data, saving energy spent on data transmission. Clustering nodes that have detected events is a convenient and commonly used mechanism to help data aggregation. If full-aggregation happens both in and out of event areas, such applications as monitoring the maximum or minimum ambient temperature, the average intensity of light and so on, constructing a suitable routing structure to optimize the data aggregation is equal to finding an approximate Steiner Tree based on event nodes. The dynamic routing algorithm for data-aggregation optimization (DRA) is elaborated here, which can build an approximate Steiner Tree based on event nodes dynamically to achieve better data aggregation with low control overhead.

A. Network Initialization Phase

After deploying sensor nodes to the monitoring field, a shortest path tree measured by hops(we call it Hop-Tree) is built by the Sink flooding a *H*op *C*onfiguration *M*essage (*HCM*). An *HCM* is a 3-tuple as < Type, ID, HTS >, where Type specifies *HCM* message, ID is the identifier of the *HCM* forwarder, HTS (Hop-To-Sink) is the distance by which the *HCM* message has passed. Each node holds some fields accordingly: NH (the Next Hop in the routing structure for the whole network), ID (the IDentifier of the node), and HTS (the Hops from the node To the Sink).

TABLE I: Hop-Tree Building Algorithm

1. The Sink floods an *HCM* message with HTS=0
2. For each node *u* that received an *HCM* message
3. If HTS (*u*)> HTS (*HCM*)+1
4. NH (*u*)=ID(*HCM*);
5. HTS (*u*)=HTS (*HCM*)+1; ID(*HCM*)=ID(*u*); HTS (*HCM*)= HTS(*u*);
6. *u* retransmits the *HCM* message to its neighbors;

Hop-Tree building algorithm is shown in Table I. Initially, the HTS of the Sink is 0 and others ∞. On receiving an *HCM*, any node compares its HTS with the HTS in the *HCM*. If there is a shorter path to the Sink, the node will update the relevant information and retransmit the *HCM*, as shown in Lines 3-6. Otherwise, the received

HCM will be discarded. This process runs repeatedly until a Hop-Tree rooted at the Sink is built.

B. *Clustering Phase*

If an event occurs, a cluster based on the nodes detecting it will be formed in a distributed manner. A cluster head is responsible for its cluster members and aggregates data in the cluster. How to elect a proper cluster head is the key process in clustering phase. There are several metrics for head election, such as maximum node degree, maximum residual energy, minimum node identifier (ID) *et al*.. For ease of comparison, the paper adopts the head election metric used by InFRA: the node with the minimum ID will be the cluster head. Clustering algorithm is depicted in Table II.

TABLE II: CLUSTERING ALGORITHM

```
1.  For each node u that detected the event
2.     u sends a DM to its neighbors and waits for a proper time to
receive DMs;
3.     If ID(u) is smaller than any ID(v)
                % v is a neighbor of u and has detected the same event
4.        Role(u)=CH; CH_ID1(u)=ID(u); CH_ID2(u)=NULL;
5.     Else          % w is the neighbor of u with the smallest ID;
6.        Role(u)=CM; CH_ID1(u)=NULL; CH_ID2(u)=w;
7.     If Role(u)==CH
8.        u broadcasts a CA within the event scope;
9.  While u receives a CA
10.    If CH_ID1(u)==NULL
11.       If CH_ID2(u)<CH_ID(CA)
12.          u discards the CA;
13.       Else
14.          CH_ID1(u)=CH_ID(CA);
15.          NH_C(u)=S_ID(CA);
16.          S_ID(CA)=u;
17.          u retransmits the CA;
18.    else
19.       If CH_ID1(u)>CH_ID(CA)
20.          Do the same operations as shown in Lines 14-17;
21.       Else
22.          u discards the CA;
```

In this phase, each node exchanges *D*etecting *M*essages (*DMs*) with its neighbors to figure out the event detection situation and the candidate cluster heads campaign for the formal cluster head by means of *C*luster-head *A*nnouncement message (*CA*). *DM* and *CA* are both 3-tuple as <Type, ID, E_ID> and <Type, CH_ID, S_ID> respectively, where Type specifies *DM*/*CA* message, ID is the identifier of the sender, E_ID identifies the event, CH_ID specifies the cluster head, and S_ID is the identifier of the *CA* forwarder. Correspondingly, each node holds 4 fields: Role(Cluster Head<CH> or Cluster Member<CM>), CH_ID1(the ID of formal cluster head), CH_ID2(the ID of temporary cluster head), and NH_C(next hop in the cluster).

On monitoring an event, any node figures out the situation of event monitoring and IDs of its neighbors by exchanging *DMs*. If its ID is the smallest, it will become candidate cluster head. Otherwise, it should be cluster member and set its temporary cluster head the neighbor who has the smallest ID, as shown in Lines 3-6. Then, the nodes whose Role is CH send *CAs*. By forwarding *CAs*, the node with the smallest ID will become the cluster head and intra-cluster routing structure will be built at the same time, which is detailed in Lines 9-22.

Fig. 1 shows the clustering process of 10 nodes deployed as Fig. 1(a). First, each node decides whether it becomes a candidate cluster head or not by exchanging *DMs*. As shown in Fig. 1(b), node 1, 2 and 3 become candidate cluster heads and others cluster members. Then, as shown in Fig. 1(c), node 1, 2 and 3 send out their *CAs* to campaign for cluster head. Node 4 and 5 receive a *CA* respectively from node 1, and modify their CH_ID1 to 1 and NH_C to 1. Node 6, 7 and 8 operate similarly, setting CH_ID1 to 2, 2, 3 and NH_C to 2, 2, 3 respectively. Next, node 4 and 5 retransmit a *CA* respectively, shown in Fig. 1(d). *CAs* received by node 1, 4 and 5 are discarded. Node 6 updates its CH_ID1 to 1 and NH_C to 4 due to the smaller ID of cluster head notified by the received *CA*. Node 7 and 8 act similarly, both updating CH_ID1 to 1 and NH_C to 5. Fig. 1(e)-(f) show the similar operations of the 10 nodes, not explained repeatedly here.

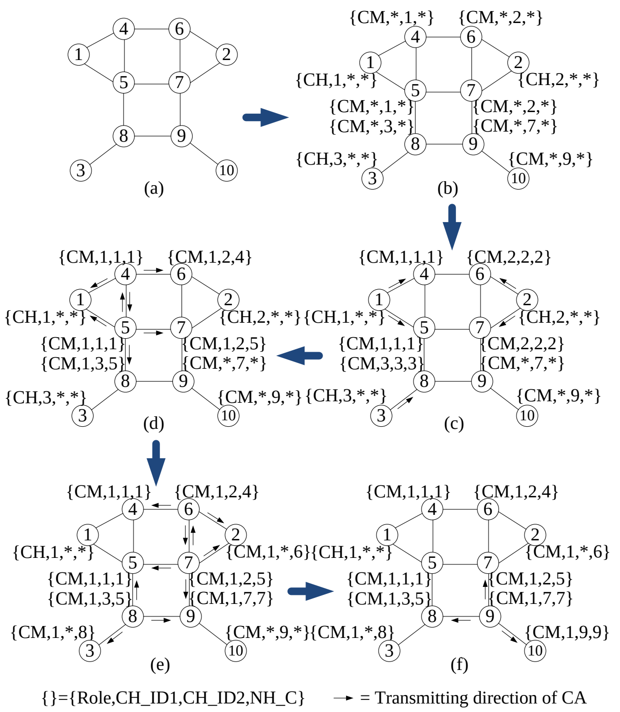

Figure 1. Example of clustering process

C. *Route Update Phase*

The cluster head should report its information to the Sink along the existing Hop-Tree when clustering finishes or event ends. After that, the Sink broadcasts the cluster head information with a power strong enough to let every node know the situation about cluster heads. Any node that gets cluster head information calculates the sum of distance to cluster heads according to formula (1).

$$\text{distance\_CHs}(v) = \sum_{CH \in CH\_Set} \text{distance}(v, CH) \quad (1)$$

where *CH_Set* is the set of cluster heads, distance(v, CH) is the Euclidean distance from node v to cluster head CH, and distance_CHs(v) is the sum of distances from node v to the cluster heads.

Each node should choose its neighbor nearer the Sink as the next hop. If there are several candidates, the one with the smallest *distance_CHs* will win, as shown in Table III. Fig. 2 shows the routing structure with three events, while (a) depicting the strategy of shortest path tree based on cluster heads and (b) showing our routing update strategy. The numbers beside nodes represent HTS and the arrows point out the next hops. From Fig. 2(b), we can see that our strategy can lead to a shortest path tree with greater path overlap for better data aggregation (an approximate Steiner Tree).

TABLE III: ROUTE UPDATE ALGORITHM

1. If an event occurs or finishes
2. The cluster head of the event will report the case to the Sink;
3. The Sink broadcasts the cluster head information to the whole network;
4. Each node calculates its *distance_CHs*;
5. Node u finds a neighbor v who satisfies: distance_CHs(v) = $\min\{\text{distance\_CHs}(w) \mid w \in \text{Neighbor}(u), \text{HTS}(w) < \text{HTS}(u)\}$;
6. NH(u)=v;

D. Data Transmission Phase

The data transmission of DRA consists of intra-cluster and inter-cluster data transmission. Intra-cluster data transmission occurs within a cluster and data is transmitted from cluster members to the cluster head according to NH_C. Inter-cluster data transmission is responsible for transmitting data from cluster heads to the Sink with the help of NH. Due to the cluster head election based on ID, in some cases, the next hop of a cluster head might be its cluster member, leading to data back propagation and waste of energy. In order to avoid data back propagation, we adopt role migration similar to the one used by InFRA. When such case occurs, any related node just updates its NH_C with NH and modifies the Role accordingly to make sure that data can be transmitted within cluster normally and routed out of cluster correctly while avoiding data back propagation. During the data transmission, no matter inside or outside cluster, once several data meet at the same node, they will be aggregated fully.

IV. ALGORITHM ANALYSIS AND EXPERIMENTS

A. Algorithm Analysis

Since DRA is an improved algorithm based on InFRA, we analyze DRA compared with InFRA.

1) The clustering mechanism of DRA(DRA_C) leads to lower control overhead compared with that of InFRA (InFRA_C)

InFRA_C is a two-stage mechanism: (a) each event node compares its ID with neighbors' IDs to decide whether it is a candidate cluster head or not, (b) candidate cluster heads flood *CAs* to all cluster members. DRA_C is also a two-stage mechanism. The first stage of DRA_C is the same with that of InFRA_C, and the second stage is different. Suppose the number of nodes in the event area is N_C, the number of candidate cluster heads after the first stage is N_{CH}, and the average number of neighbors within the cluster is N_N. For InFRA_C, the amount of *CAs* sent and received is $N_C N_{CH}$ and $N_C N_N N_{CH}$ respectively. For DRA_C, the *CA* originated from the best candidate cluster head can be flooded to all cluster members and the other *CAs* might not be transmitted through the whole event area often. The worst case is when all the candidate cluster heads flood *CAs* in the ID descending order, with the amount of *CAs* sent and received being $N_C N_{CH}$ and $N_C N_N N_{CH}$ respectively. If the best candidate cluster head first floods *CA* while others keeping silent, the amount of *CAs* sent and received will be the least, N_C and $N_C N_N$ respectively. So, the average amount of *CAs* sent and received for DRA_C is $(N_{CH}+1)N_C/2$ and $(N_{CH}+1)N_C N_N/2$ respectively. It is easy to know that $(N_{CH}+1)N_C/2 \le N_C N_{CH}$ and $(N_{CH}+1)N_C N_N/2 \le N_C N_N N_{CH}$ with equality holding up if and only if $N_{CH}=1$. As $N_{CH}>1$ usually, DRA_C leads to lower control overhead compared with InFRA_C.

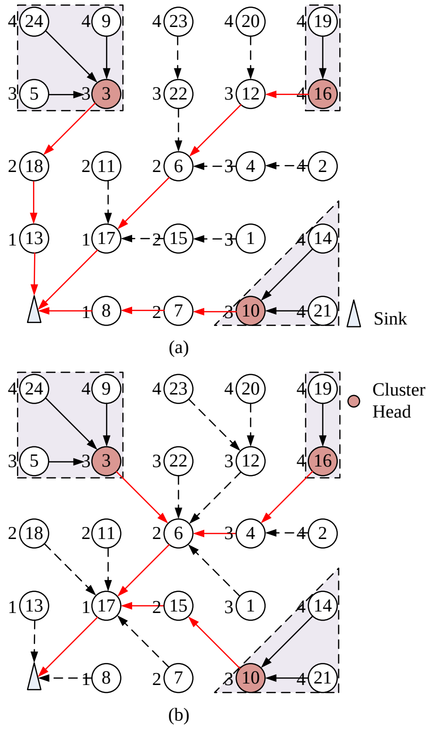

Figure 2. Routing structure (a) shortest path tree (b) our strategy

2) The routing update mechanism of DRA(DRA_R) is more energy-efficient than that of InFRA(InFRA_R)

InFRA_R requires that any node should know the sum of hop distances to the all cluster heads. After clustering, the cluster head floods its information through the whole network to build a shortest path tree (by hops) routed at itself. Then, any node gets the hop distance to the cluster head. Suppose the total number of nodes is N, the hops of the farthest node to the Sink is D, the number of events is N_E, the size of network is A, and the node communication radius is r. It is easy to know the average number of neighbors is $\pi r^2 N/A$. For InFRA_R, the number of control packets sent and received is NN_E and $\pi r^2 N^2 N_E/A$ at least respectively. After an event occurring, DRA_R gets the cluster head report its information to the Sink, having the number of control packets sent and received is DN_E (less than NN_E) and $(D\text{-}1)N_E$ at most respectively. Then, the Sink broadcasts the cluster head information in the monitoring area using a power strong enough, and each node receives the information. From the above, the total number of control packets received is $(N+D\text{-}1)N_E$ at most. In order to ensure network connectivity, $\pi r^2N/A>8$ should be hold. Since $N+D\text{-}1<N+N<2N$, we can get $(N+D\text{-}1)N_E<\pi r^2N^2N_E/A$.

So, DRA_R is more energy-efficient than InFRA_R.

3) The routing structure constructed by DRA is an approximate Steiner Tree with good approximation ratio and DRA performs better than InFRA in the case of high network density.

In route update phase, both InFRA and DRA rule that each node should choose the nearer neighbor to the Sink as the next hop. If there are several candidates, the node with the minimum sum of hops to the cluster heads will win under InFRA while DRA chooses the node with the minimum sum of Euclidean distances to the cluster heads as the next hop. Both of them can ensure shortest paths with greater overlap. So, the route structure performance of DRA is similar to that of InFRA. Moreover, E.F. Nakamura *et al*. have proved that InFRA can produce an approximate Steiner Tree with good approximation ratio [13]. Hence, it is easy to deduce that DRA can also construct an approximate Steiner Tree with good approximation ratio.

As network becoming denser and denser, the number of nodes with the minimum sum of hops to the cluster heads increases accordingly. InFRA will choose the next hop randomly from those nodes, with more difficult to guarantee overlapping paths as soon as possible. Due to the rule based on the sum of Euclidean distances to the cluster heads, the next hop selection of DRA is less random, still resulting in greater path overlap. Under high density, DRA performs better than InFRA.

4) Shortcoming

For DRA, if events occur at some fixed places, the routing structure will nearly not change, leading to the nodes responsible for transmitting aggregated data being almost the same and with heavier load which causes energy consumption unbalanced. If full-aggregation is not adopted, the above drawback will be more obvious. In practice, the events do not occur at some fixed places usually, and an event may occur in any place within the sensor field. The routing structure is adaptive to the events accordingly and the nodes responsible for transmitting aggregated data will not always be the same, relieving the energy consumption imbalance for DRA. And if we adopt full-aggregation for both intra- and inter-cluster data process, there will be no problem about energy consumption imbalance. DRA is designed specifically for applications with full data aggregation, so the shortcoming about energy consumption imbalance can be ignored in realistic WSNs.

B. *Experiments*

In order to show the effectiveness of DRA, we compare it with InFRA, SPT and CNS. Suppose event area is circular in shape, and the position, time and duration of an event is random. The energy consumption model in [4] is employed here and the default simulation parameters are shown in Table IV.

TABLE IV: DEFAULT SIMULATION PARAMETERS

Parameter	Value
Network size	1000m×1000m
# of nodes	5184
Communication radius	35m
Data packet size	4000bits
Control packet size	200bits
Notification rate	40s
Event radius	60m
Event duration	2h-4h, uniform distribution
# of Simultaneously events	2
Inactivity time	0.5h
E_{elec} in energy model	50nJ/bit
E_{amp} in energy model	100pJ/bit/m^2
Network running time	12h

We evaluate the algorithms by the following network performance metrics:

- Data packets: the amount of data packet transmissions in the whole network.
- Energy consumption: the energy consumed by the whole network.
- Control overhead: the amount of control packet transmissions for building and maintaining network structure.
- Routing efficiency: the amount of packet transmissions used to process and deliver all data packets generated by source nodes.

(1) Clustering overhead

To illustrate the energy-efficiency of clustering for DRA, here we only compare DRA_C with InFRA_C by increasing event radius from 40m to 80m. The results are shown in Fig. 3. The number of *CAs* sent and received grows with increasing event radius for the both mechanisms. However, DRA_C has less *CAs* and more gradual growth in the number of *CAs* with increasing event radius compared with InFRA_C. The experimental results are consistent with the theoretical analysis and show that DRA_C is superior to InFRA_C.

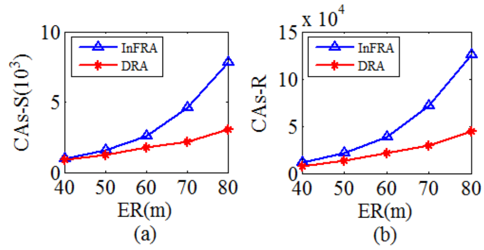

Figure 3. Clustering overhead (a) The number of *CAs* sent (b)The number of *CAs* received

(2) Event size

This section, we evaluate the influence of event size on the algorithms by changing event radius from 40m to 80m, with results shown in Fig. 4.

As event radius increases, the amount of in-network data packet transmissions for the four algorithms grows due to the increasing number of nodes that detect the event. Since DRA and InFRA cluster nodes to aggregate data within event and use some heuristic methods to increase the degree of path overlap for further data aggregation occurring as soon as possible, the amount of data packet transmission of them is less than that of SPT and CNS. In the cases of high density, InFRA_R leads to more "best" candidates for a node randomly choosing its next hop which decreases the degree of path overlap, while DRA_R makes the random of next hop selection of a node weaker by using the sum of Euclidean distances from the node to all cluster heads to keep greater path overlap. The amount of data packet transmission of DRA is slightly less than that of InFRA, as shown in Fig. 4(a).

In order to update the sum of hops to the cluster heads, InFRA depends on the cluster heads flooding their information through the whole network, so the control overhead for structure maintenance of InFRA is much higher than that of DRA, CNS and SPT. Moreover, as illustrated in the previous section, the clustering overhead of both DRA and InFRA increases with increasing event radius and DRA has less clustering overhead and more moderate rate of increasing clustering overhead. Consequently, compared with InFRA, DRA has less total control overhead and is more moderate in overhead increasing, shown in Fig. 4(b).

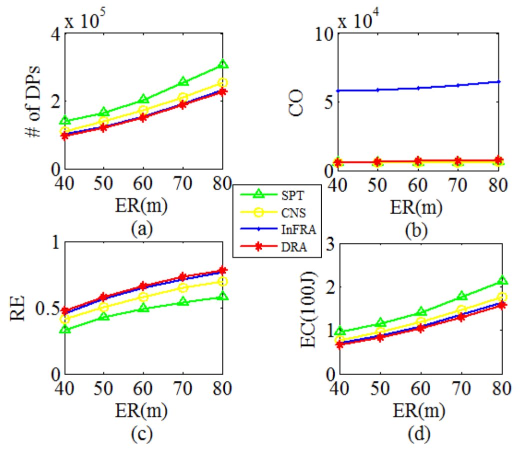

Figure 4. Event radius varying (a)Data packets (b)Control overhead (c)Routing efficiency (d)Energy consumption

Fig. 4(c) shows the comparison on routing efficiency of the algorithms. Due to full-aggregation, routing efficiency grows as event radius increases. DRA produces the least in-network data transmissions, so the routing efficiency of DRA is best, followed by InFRA. Since DRA can make the data aggregation occur as soon as possible to gain less in-network data transmissions and has low overhead on route construction and maintenance, the total energy consumption of DRA is the least of the four algorithms, as shown in Fig. 4(d).

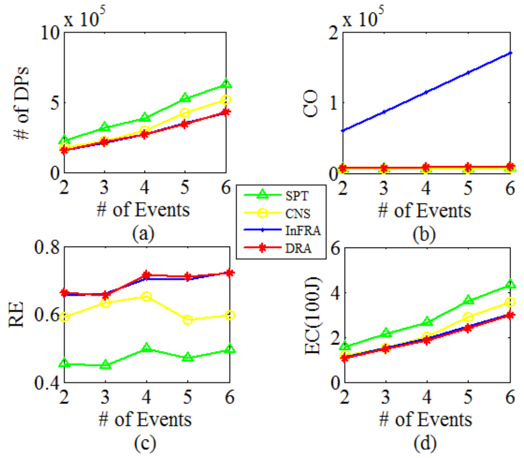

Figure 5. Number of simultaneous events varying (a)Data packets (b)Control overhead (c)Routing efficiency (d)Energy consumption

(3) Event Scalability

Fig. 5 shows the performance of the algorithms by increasing the number of simultaneous events from 2 to 6. The data aggregation of DRA is similar to that of InFRA when the number of simultaneous events increases, followed by CNS, and SPT worst, as shown in Fig. 5(a). For InFRA, since the cluster head of each event should flood control information through the whole network, the number of control packets increases with the number of events increasing, and for DRA, each cluster head unicasts its information to the Sink, so the number of control packets also increases as the number of events increases, but not much, which is shown in Fig. 5(b). Fig. 5(c) depicts that the routing efficiency of DRA, InFRA, CNS and SPT decreases sequentially. From Fig. 5(a)-(c), it is easy to know that DRA is the most energy-efficient, which is confirmed by Fig. 5(d).

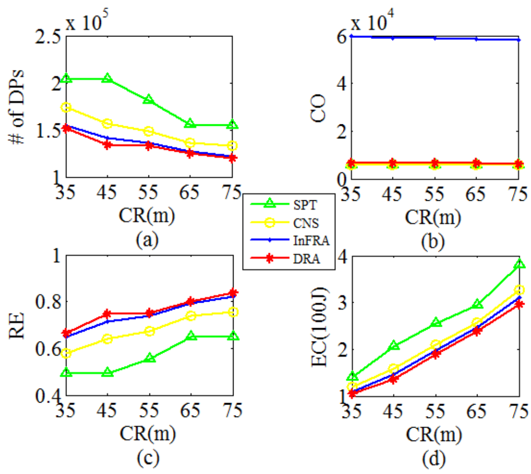

Figure 6. Communication radius varying (a)Data Packets (b)Control Overhead (c)Routing Efficiency (d)Energy Consumption

(4) Communication radius

To evaluate how the algorithms behave when communication radius varies, we simulate 5184-node networks, increasing communication radius from 35m to 85m, with results shown in Fig. 6. Because the number of hops from a node to the Sink and from a cluster member to a cluster head decreases respectively with increasing communication radius, the amount of both data and control packet transmissions decreases and routing efficiency increases for all the four algorithms, while DRA works the best due to better data aggregation, as shown in Fig. 6(a)-(c). Fig. 6(d) shows that the energy consumption of DRA is the best of the four algorithms.

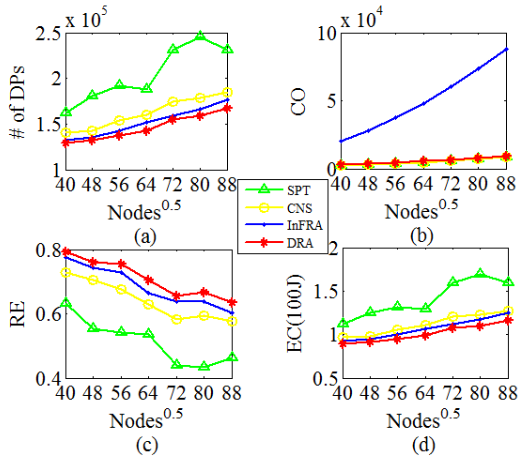

Figure 7. Network size varying (a)Data packets (b)Control overhead (c)Routing efficiency (d)Energy consumption

(5) Network scalability

To evaluate the impact of the network scalability on the algorithms, we increase the network size from 1600 to 7744 nodes and resize the sensor field to keep a constant network density of 20, with results shown in Fig. 7. The nodes that take part in information transmission increases as the network size increases, leading to data and control packet transmissions growing, routing efficiency decreasing and energy consumption increasing for the four algorithms, shown in Fig. 7(a)-(d) respectively. From Fig. 7(b), we can see that the control overhead of InFRA grows drastically due to flooding cluster head information over the whole network and the network scalability of InFRA is poor. Because DRA can built an approximate Steiner Tree under low control overhead, the performance of DRA is still the best while network size varying.

(6) Density

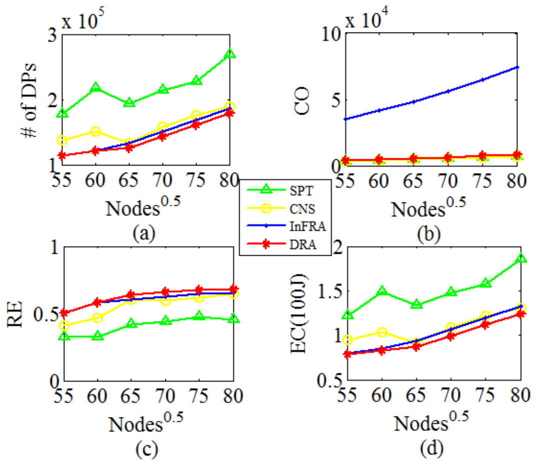

Figure 8. Density varying (a)Data packets (b)Control overhead (c)Routing efficiency (d)Energy consumption

We evaluate the density impact by keeping the sensor field (1000m×1000m) and communication radius (35m) constant and varying the number of sensor nodes from 3025 to 6400, with the results shown in Fig. 8. As the density increases, the number of nodes increases, leading to the amount of data and control packet transmissions growing for all the four algorithms, shown in Fig. 8(a)-(b). Compared with InFRA, CNS and SPT, the performance of DRA is still the best when density increases, which is depicted in Fig. 8.

V. Conclusion

In this paper, we study the routing problem in event-driven WSNs for better data aggregation and low control overhead, and propose a novel Dynamic Routing Algorithm for data-aggregation optimization (DRA). DRA can cluster nodes within event areas in a distributed manner by less control packets, which is energy-efficient confirmed by both algorithm analysis and experiments. Moreover, by DRA, cluster heads unicast their information to the Sink and then the Sink broadcasts them to the whole network using a power strong enough. Any node calculates its *distance_CHs* based on the received information of cluster heads and chooses the neighbor which is not only nearer to the Sink but also has the smallest *distance_CHs* as its next hop, resulting in an approximate Steiner Tree eventually. DRA can optimize the efficiency of data aggregation and decrease the control overhead over building and maintaining routes, leading to energy-efficient data routing and collection. Algorithm analysis and experiments prove that DRA is effective on data aggregation and control overhead compared with InFRA *et al*. and conducive to prolong the network lifetime.

DRA is based on full-aggregation and suitable for data collection applications under high correlation conditions. In practice, the correlation among events is different. How to exploit the data correlation among events and design routing algorithm for efficient data collection under different data correlation is one of our future works.

Acknowledgment

This work was supported by a grant from Youth Fund of Luoyang Institute of Science & Technology (2008QZ11).

References

[1] B. Krishnamachari, D. Estrin, and S. Wicker. “Modeling data-centric routing in wireless sensor networks,” in *Proc. IEEE INFOCOM, the 21st Annual Joint Conference of the IEEE Computer and Communications Societies*, New York, USA, 2002, pp. 1-18.

[2] R. Cristescu, B. Beferull-Lozano, M. Vetterli, and R. Wattenhofer, “Network correlated data gathering with explicit communication: Np completeness and algorithms,” *IEEE/ACM Transactions on Networking*, vol. 14, no. 1, pp. 41-54, February 2006.

[3] C. Buragohain C, D. Agrawal, and S. Suri, “Power aware routing for sensor databases,” in *Proc. IEEE INFOCOM, the 24th Annual*

Joint Conference of the IEEE Computer and Communications Societies, Miami, USA, 2005, pp. 1747-1757.

[4] W. B. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "An application specific protocol architecture for wireless microsensor networks," *IEEE Transactions on Wireless Communications*, vol. 1, no. 4, pp. 660-670, October 2002.

[5] S. Soro and W. B. Heinzelman, "Cluster head election techniques for coverage preservation in wireless sensor networks," *Ad Hoc Networks*, vol. 7, no. 5, pp. 955-972, July 2009.

[6] L. J. Meng, H. Z. Zhang, and Y. Zou, "A data aggregation transfer protocol based on clustering and data prediction in wireless sensor networks," in *Proc. 7th International Conference on Wireless Communications, Networking and Mobile Computing*, Wuhan, China, 2011, pp. 1-5.

[7] W. S. Jung, K. W. Lim, Y. B. Ko, and S. J. Park, "Efficient clustering-based data aggregation techniques for wireless sensor networks," *Wireless Networks*, vol. 17, no. 5, pp. 1387-1400, July 2011.

[8] S. Kwon, J. H. Ko, J. Kim, and C. Kim. "Dynamic timeout for data aggregation in wireless sensor networks," *Computer Networks*, vol. 55, no. 3, pp. 650-664, February 2011.

[9] J. Al-Karaki, R. Ul-Mustafa, and A. Kamal, "Data aggregation in wireless sensor networks-exact and approximate algorithms," in *Proc. High Performance Switching and Routing Workshop*, IA, USA, 2004, pp. 241-245.

[10] S. Hougardy and H. J. Promel, "A 1.598 approximation algorithm for the steiner problem in Graphs," in *Proc. 10th Ann. ACM-SIAM Symp. Discrete Algorithms*, MD, USA, 1999, pp. 448-453.

[11] G. Robins and A. Zelikovsky, "Improved steiner tree approximation in Graphs," in *Proc. 11th Ann. ACM-SIAM Symp. Discrete Algorithms*, California, USA, 2000, pp. 770-779.

[12] L. A. Villas, A. Boukerche, H. S. Ramos, H. A. B. F. Oliveira, R. B. Araujo, and A. A. F. Loureiro, "DRINA: A lightweight and reliable routing approach for in-network aggregation in wireless sensor networks," *IEEE Transactions on Computers*, vol. 62, no. 4. pp. 676-689, April 2013.

[13] E. F. Nakamura, H. S. Ramos, L. A. Villas, H. A. B. F. Oliveira, A. L. L. Aquino, and A. A. F. Loureiro, "A reactive role assignment for data routing in event-based wireless sensor networks," *Computer Networks*, vol. 53, no. 12, pp. 1980-1996, August 2009.

[14] L. A. Villas, A. Boukerche, H. A. B. F. Oliveira, R. B. Araujo, and A. A. F. Loureiro, "A spatial correlation aware algorithm to perform efficient data collection in wireless sensor networks," *Ad Hoc Networks*, 2011.

[15] B. Krishnamachari, D. Estrin, and S. B. Wicker, "The impact of data aggregation in wireless sensor networks," in *Proc. 22nd Int'l Conf. Distributed Computing Systems*, New York, USA, 2002, pp. 575-578.

[16] L. A. Villas, D. L. Guidoni, R. B. Araujo, A. Boukerche, and A. A. F. Loureiro, "A scalable and dynamic data aggregation aware routing protocol for wireless sensor networks," in *Proc. 13th ACM International Conference on Modeling, Analysis, and Simulation of Wireless and Mobile Systems*, Bodrum, Turkey, 2010, pp. 110-117.

Yalin Nie was born in Yiyang, Hunan, China, August 1981. She received her B.S. in Computer Science and Technology in 2004 and M.S. in Computer application technology in 2007, both from School of Computer and Communication of Hunan University in Changsha, Hunan, China. Currently, she is working toward the Ph.D. in School of Computer Science and Technology of Xidian University in Xi'an, Shaanxi, China. She is also a teacher in Department of Computer and Information Engineering of Luoyang Institute of Science and Technology in Luoyang, Henan, China. Her research interest includes routing protocols, mobile computation and data aggregation in wireless sensor networks.

Prof. Sanyang Liu was born in Xi'an, Shaanxi, China, October 1951. He received his B.S. at Shaanxi Normal University in 1982, gained his M.S. at Xidian University in 1984, and received his Ph.D. from Xi'an Jiaotong Universty in 1989. At present, Prof. Liu is a PhD supervisor of Xidian University, the Dean of School of Sience of Xidian University and the Director of Institute of Industrial and Applied Mathematics. His research interest covers theory and application of optimization, network arithmetic.

Zhibin Chen was born in Zhangzhou, Fujian, China, December 1989. He received his B.S. from Xiamen University Tan Kah Kee College. Currently, he is working toward the M.S. in Microelectronics and Solid State Electronics at School of Microelectronics of Xidian University in Xi'an, Shaanxi, China. His research interest includes network modeling, algorithm analysis and design, and network optimization methods.

Prof. XiaoGang Qi was born in Baoji, Shaanxi, China, December 1973. He received his B.S. degree and M.S. degree in 1995 and 1996 from Institute of Equipment Command and Technology of Chinese People's Liberation Army, and gained his Ph.D. degree from School of Science of Xidian University in 2005. Now, Prof. Qi is an MS supervisor of Xidian University, the Director of the Society of Industrial and Applied Mathematics of Shaanxi, the leader and head coach of Mathematical Contest in Modeling of Xidian University. His research interest covers network optimization theory and methods, network algorithm analysis and design, and complex systems modeling and computer simulation.