Computer Communications 35 (2012) 1132–1140

Contents lists available at SciVerse ScienceDirect

Computer Communications

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

Structure-free real-time data aggregation in wireless sensor networks

Hamed Yousefi [a,*], Mohammad Hossein Yeganeh [b], Naser Alinaghipour [b], Ali Movaghar [a]

[a] Department of Computer Engineering, Sharif University of Technology, Tehran, Iran
[b] Department of Computer Engineering, Science and Research Branch, Islamic Azad University, Tehran, Iran

ARTICLE INFO

Article history:
Available online 25 November 2011

Keywords:
Wireless Sensor Networks
Data Aggregation
Real-time Forwarding

ABSTRACT

Data aggregation is a very important method to conserve energy by eliminating the inherent redundancy of raw data in dense WSNs. Although structured approaches are particularly useful for data gathering applications, they incur high maintenance overhead in dynamic scenarios for event-based applications. Moreover, a WSN should be capable of timely fulfilling its mission without losing important information in event-critical applications. In this paper, we focus on designing a structure-free Real-time data AGgregation protocol, RAG, using two mechanisms for temporal and spatial convergence of packets – Judiciously Waiting policy and Real-time Data-aware Anycasting policy. Using extensive simulations in NS-2, we investigate the performance of RAG in terms of aggregation gain, miss ratio, energy consumption, and end-to-end delay for WSNs.

1. Introduction

Recent advancement in wireless communications, Micro Electro Mechanical Systems (MEMS), and also a tendency to use low cost, tiny, and autonomous high performance products have led to the emergence of Wireless Sensor Networks (WSNs) [1]. A WSN contains a set of thousands or more sensor nodes densely deployed in different environments to monitor temperature, pressure, light, or other physical information. The limited energy of nodes seeks to utilize energy efficient routing algorithms for achieving long-lived wireless multi-hop networks where communication costs (transmission power) are usually more expensive than computing costs [2].

Moreover, it is highly possible that nodes located at the monitored area sense repeated or redundant data. Therefore, much energy is dissipated if all these redundant data are forwarded over the network. Data aggregation is a promised technique aiming to reduce the number of packet transmissions and remove the inherent redundancy of raw data by exploiting in-network processing [2].

To aggregate data, every node combines all the received packets with its own packet into a single packet of fixed-size according to some aggregation function such as logical and/or, average, maximum, or minimum and then forwards it to upper nodes. A routing for data aggregation is a spanning inward architecture of the communication topology rooted at the sink of the aggregation [3].

The two necessary conditions for efficient data aggregation are spatial and temporal convergence [4,5]. Thus, the main challenges are building up proper routes and also applying efficient timing control mechanisms to the packets in such a way that they have more chance to meet at the same node at the same time. Actually, different streams are aggregated if they happen to intersect on their way to the sink. Approaching this goal, we have to hold packets in intermediate nodes to promote aggregation efficiency. More waiting time can lead to the collection of more data, the increase of aggregation gain, and vice versa. Therefore, data aggregation has a tradeoff relationship with the delay.

Although energy efficiency is usually the primary concern in WSNs, the requirement of real-time communication is becoming more and more important in emerging applications. A real-time sensor system has many applications, especially in intruder tracking, medical care, fire monitoring, and structural health diagnosis. However, its wireless nature, limited resources (power, processing, and memory), low node reliability, and dynamic network topology dramatically make it different from the traditional real-time systems. Here, outdated information would be irrelevant and even lead to negative effects on the system monitoring and control. However, aggregation extends the queuing delay at the intermediate nodes and can thus complicate the handling of delay-constrained data. Therefore, a key issue is how to effectively route and hold delay-constrained data so that real-time applications can be supported with a minimum power-consumption.

Besides, in the literature, most of the present routing schemes for data aggregation rely on a structured architecture, such as cluster-based [6,7] and tree-based [8–16]. However, in a dynamic environment where source nodes change with the situation, the benefit from structured data gathering may not compensate for the

* Corresponding author. Tel.: +98 2166166678.
E-mail addresses: hyousefi@ce.sharif.edu (H. Yousefi), m.yeganeh@srbiau.ac.ir (M.H. Yeganeh), n.alinaghipour@srbiau.ac.ir (N. Alinaghipour), movaghar@sharif.edu (A. Movaghar).

0140-3664/$ - see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.comcom.2011.11.007

construction and maintenance overhead. The two examples in Fig. 1 show the poor performance of cluster-based and tree-based architectures in event-based applications. While the nodes detecting the same event have the chance to spatially converge at the same point close to the event, their reports take partially (even completely) different routes to the sink, so this wastes the energy resources. Moreover, in many applications, several types of data are considered. Looking at Fig. 2, the static routes are not optimal in terms of conserving energy while a dynamic scheme can reduce the average number of transmissions by converging the same-type packets as much as possible along the way to the sink node. This improves the degree of aggregation. Furthermore, in real-time applications, it is necessary to dynamically route and schedule the delay-constrained packets to adapt to various environment changes. To achieve full benefits, the structure-free approaches perform dynamic data aggregation using local information, so they do not spend extra energy to build a structure [17,4].

In this paper, we propose an efficient structure-free algorithm supported by a real-time routing, RAG, for energy saving through data aggregation. Our protocol uses two mechanisms for temporal and spatial convergence of packets as follows:

- A Judiciously Waiting policy taking advantage of the available slack for efficient aggregation by delaying packets on their way to the sink as long as their deadlines are not missed.
- A Real-time Data-aware Anycasting policy at the MAC layer making the routing decisions on the fly for efficient aggregation of data as there is no pre-constructed structure.

The rest of the paper is organized as follows. Section 2 summarizes the related works. Section 3 explains the proposed scheme and presents its specifications. Simulation results are presented and discussed in Section 4. Finally, Section 5 concludes our work and discusses some future directions.

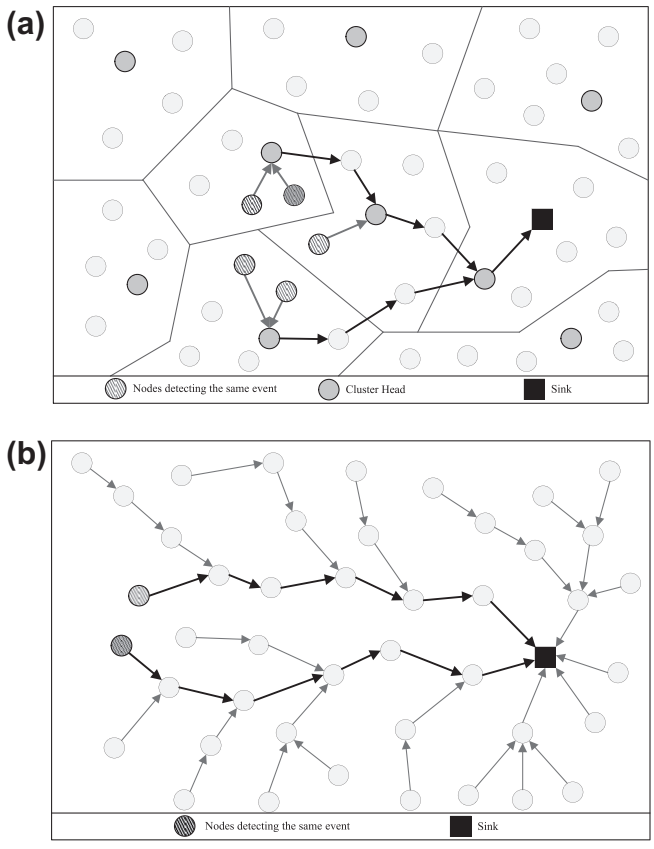

Fig. 1. Poor performance of structure-based data aggregation in event-based applications (a) cluster, (b) tree.

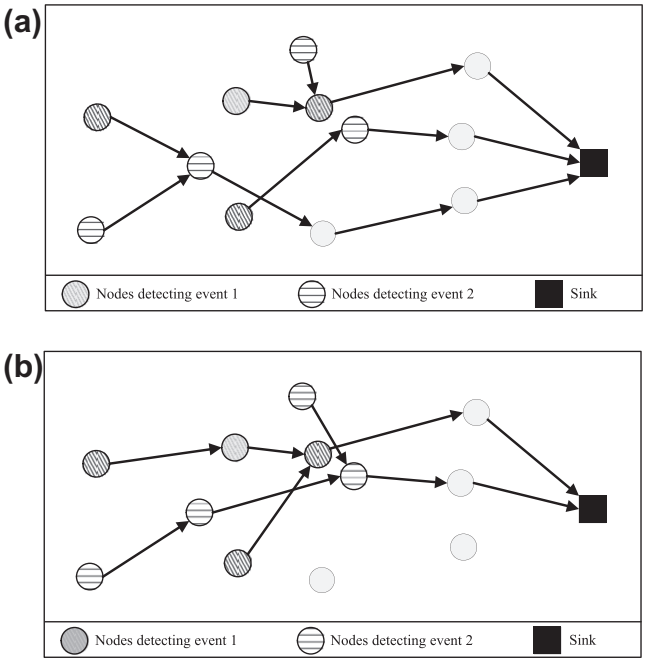

Fig. 2. Performace of data aggregation (a) static scenario, (b) dynamic scenario.

2. Related work

2.1. Real-time routing

Several routing algorithms have been developed with the aim of providing timeliness in WSNs [18–24]. Here, we briefly review some of the previous works in the field. SPEED [20] implements the end-to-end transmission delay control. It finds out the neighbors' information using a beaconing mechanism and chooses the next hop based on transmission velocity and local geographical information. Moreover, it utilizes a back pressure rerouting mechanism to avoid routing traps. MMSPEED [19], an extended version of SPEED, can provide different deadlines and packet reliabilities. Moreover, R2TP [21] is a real-time routing protocol, which utilizes multipath forwarding in such a distributed way to accomplish reliable transmission in WSNs. However, MMSPEED and R2TP are similar to SPEED in that they do not consider energy expenditure in data forwarding. This issue results in quick energy exhausting of some nodes and makes the real-time characteristic and the network lifetime worse and worse. RAP [23] prioritizes real-time traffic using velocity monotonic scheduling through a differentiated MAC layer. ARP [24] considers not only the real-time requirement but also the energy index synthetically. It computes the required transmission velocity of data packets in each hop and chooses the next node according to both transmission velocity and residual node energy. RPAR [18] tries to optimize power consumption by regulating the transmission power in real-time applications. This approach is, however, affected by anomalous behavior in heavy traffic conditions, which tends to favor network congestion. Hence, RPAR increases the transmission power that worsens the situation. In THVR [22], routing decisions are made based on two-hop neighborhood velocity integrated with the residual energy awareness mechanism. However, it might lead to high computing complexity

and heavy message exchange overhead to enhance the service quality of real-time packet delivery in WSNs.

2.2. Data aggregation

Data aggregation, by reducing the number of transmissions, is an effective approach to save energy. Temporal and spatial convergence during transmission are two necessary conditions for aggregation [4,5]. To achieve a high aggregation ratio, one category focuses on efficient timing control and the other focuses on establishing a proper routing scheme. However, in data aggregation protocols, most of the present schemes rely on the static structures, such as cluster-based or tree-based.

In the cluster-based approach sensor nodes are organized into clusters. There is a cluster head in each cluster which is responsible for aggregating data from all the sensors in the cluster and transmitting the concise digest to the sink node. LEACH [6] and HEED [7] are two typical examples.

Tree-based data aggregation protocols organize sensor nodes into a tree where data aggregation is performed at intermediate nodes along the way to the sink node. One of the main aspects of tree-based networks is the construction of an energy efficient data aggregation tree, such as Steiner Minimum Tree (SMT) for multicast algorithms which can be used in designing data aggregation protocols [9,14]. In EADAT [10], sensors with higher residual power have more chance to become a non-leaf tree node and thus extend the network lifetime in terms of the number of live nodes. The Power-Efficient Data gathering and Aggregation Protocol (PEDAP) [16] computes a minimum spanning tree over a WSN with transmission overhead as the link cost to minimize the total energy expended in each communication round. In cascading timeout [15], nodes schedule their timeout based on their own positions in the aggregation tree. This mechanism allocates a smaller waiting time for nodes farthest from the sink. A parent node's timeout happens after its children's timeout thus enabling a node to collect information from all its children. In [8], the authors introduce a dynamic aggregation time assignment for tree structure based on the number of children nodes of the root node. The complicated updating process of aggregation time causes the scheme to be very sensitive to little changes in the number of child nodes when the timeout happens. Literature [11] proposes a tree-based data aggregation method in real-time sensor networks. It constructs an energy efficient data aggregation tree with theoretically bounded energy cost under the latency constraint for data gathering. The data aggregation tree is constructed through the power level adjusting of sensor nodes in centralized methods. The scheme [12] proposes a heuristic algorithm for constructing data aggregation trees that minimize total energy cost under the latency bound. It develops an analytic model for IEEE Standard 802.15.4 CSMA-CA to compute the worst case delay for a sensor node in the unit of the number of time slots based on the parent node degree and the required success transmission probability. In Adaptive Time Control (ATC) [13] the locations of the nodes and the number of children in the data aggregation tree determine the aggregation timeout for a node. Thus, by ensuring sufficient time to process data from the children, it maximizes the opportunity for data aggregation.

Such structured mechanisms perform well in a stable environment when nodes function properly all the time. However, in practical environments where nodes may fail unexpectedly and also, in event-based applications, the benefit from structured gathering may not compensate for the construction and maintenance overhead. On the contrary, the structure-free approaches do not spend extra energy to build any structure. Instead, they achieve data aggregation using local information. By identifying the limitations of the static routing schemes for the data aggregation, the scheme [25] proposes Dynamic Convoy Tree-based Collaboration (DCTC) to optimize the tree reconfiguration schemes in event-based applications. However, DCTC involves heavy message exchanges. Data-Aware Anycast (DAA) [4] is the first proposed structureless data aggregation protocol that can achieve high aggregation without incurring the overhead of structure approaches. DAA uses anycast to forward packets to one-hop neighbors that have packets for aggregation and also uses Randomized Waiting (RW) at the source nodes for each packet to introduce artificial delays and increase temporal convergence. In [26] a semistructured approach utilizes DAA in a dynamic forwarding algorithm to support network scalability on an implicitly constructed structure composed of multiple shortest path trees. In SFEB [17], Structure-Free and Energy-Balanced data aggregation protocol, the two-phase aggregation and dynamic aggregator selection enable both efficient data gathering and balanced energy consumption. In Phase One, using the concept of "gather before transmit", some data collecting nodes are selected first to gather their neighbors' sensing data as many packets as possible. Then, these aggregators send the collected packets back to the sink at Phase Two. The scheme [5] designs an effective Data Aggregation mechanism Supported by Dynamic Routing (DASDR) which can adapt itself to different scenarios without incurring much overhead. Enlightened by the concept of potential field in the discipline of physics, the dynamic routing in DASDR is designed based on two potential fields: the depth potential field which guarantees that packets will reach the sink at last and the queue potential field which makes packets more spatially convergent. Cooperating with a timing scheme similar as that in [15], this dynamic routing scheme can efficiently aggregate data.

3. RAG

In this section, we introduce the network model, our structure-free Real-time data AGgregation protocol (RAG), and energy consumption model, respectively.

3.1. Network model

Sensor nodes are distributed randomly into the two dimensional area in a static WSN and there exists one sink node that collects information from the sensors. Each node learns its own location and the geographic positions of the sink. The sink has no resource limitation and the sensors are battery-operated with limited energy and the same physical capabilities. Once their energy exhausts, the sensors cannot work anymore. Moreover, the source nodes may transmit different types of packets and the intermediate nodes are responsible for performing in-network aggregation of same-type packets. The nodes which are not adjacent conduct data communication through hop-by-hop.

3.2. Protocol overview and properties

RAG is a distributed algorithm that provides a structure-free transmission environment based on an aggregation-aware real-time routing. However, temporal convergence and spatial convergence are two necessary conditions for efficient data aggregation. Therefore, packets should meet at the same time at the same node while being transmitted to the sink. To achieve these objectives, our proposed protocol provides a two-fold contribution: (1) Judiciously Waiting policy and (2) Real-time Data-aware Anycasting policy. We describe them below.

3.2.1. Judiciously Waiting policy

As we mentioned before, there is a trade-off between energy and delay because aggregation requires that some data be delayed

at intermediate nodes, while waiting for more data packets to be received. Hence, a key issue in the context of real-time monitoring is the calculation of waiting timeout for each forwarding packet in such a way that it can be delivered to the sink within the stipulated time bound.

Aggregation scheduling is mostly classified into three different categories in structure-based approaches as presented below [15]:

- Simple periodic: in this method each node sends out the aggregated packet to the next hop in a pre-defined period of time.
- Periodic per-hop: in this scheme each node performs aggregation as soon as it hears from all its children and then sends out one single packet.
- Periodic per-hop adjusted: this method is the same as the periodic per-hop policy, but in each node, holding time is calculated based on the position of the node in the tree structure.

However, in structure-free approaches, exploiting an unalterable timeout policy such as the periodic scheme is not applicable. Particularly, this problem is highlighted even more in data aggregation applications with limited data packet delivery time because passing the packet through different paths can result in the reception of packets with different deadlines, thus requiring different timing policy. Therefore, the main question is how long a packet could be delayed in the intermediate nodes so that the aggregation gain and on-time end-to-end delivery ratio will be maximized.

Here, we propose Judiciously Waiting policy for each packet along the way to the sink node to introduce artificial delays and increase temporal convergence for effective aggregation while meeting the time constraint of the data.

In real-time applications, packet's TTD (Time-To-Deadline) is used to indicate how much time it remains for the packet before its deadline and has an important role in making routing decisions. We allocate the available packet's slack time, i.e., TTD–EED, proportionately to the remaining hop count to the sink node along the forwarding path to judiciously hold packets in the intermediate nodes while surmounting real-time constraints. The estimated End-to-End Delay (EED) is the time that takes to deliver the packet from current forwarding node to the sink node. To estimate EED, we measure EHD, Estimated one-Hop Delay, including channel contentions, packet transmissions, and queuing delay by employing a time-stamping mechanism. Besides, the overall effectiveness of aggregation is dependent on when and where it actually occurs. Based on [27], aggregating data close to the source nodes is the most efficient communication for perfect aggregation functions. Therefore, in our algorithm, the waiting timeout WT for a packet at an intermediate node R_h hops away from the sink is calculated as follows:

$$\mathrm{WT} = \frac{\mathrm{TTD}-\mathrm{EED}}{1+\left(\frac{R_h-1}{R_h}\right)}\cdot\alpha = \frac{\mathrm{TTD}-(R_h\times\mathrm{EHD})}{2-\left(\frac{1}{R_h}\right)}\cdot\alpha \quad (1)$$

where α is a constant factor used to leave some remaining time as a safety margin to ensure that the deadline would be met. Moreover, the remaining hop count to the sink node is formally calculated by $R_h = \frac{L}{L_{Next}}$ where L and L_{Next} are the distance from the current node to the sink and to the next hop forwarding node (see Section 3.2.2), respectively. Based on (1), by decreasing the remaining hop count a lower part of slack time is used for data aggregation as the packet moves closer to the sink. Finally, the packet uses its entire remaining deadline as a slack time in the current node if the next hop node is the sink node (i.e., $R_h = 1$).

Our waiting time policy, by taking advantage of the available slack if any, not only improves aggregation efficiency by judiciously delaying packets in intermediate nodes but also tolerates transient periods of high contentions without requiring any synchronization among sensor nodes.

3.2.2. Real-time Data-aware Anycasting policy

In this section, we present RDA, a Real-time dynamic routing supported by a Data-aware Anycasting policy for efficient data aggregation in WSNs. Considering the fact that all intermediate nodes delay the received packets to aggregate more data, it is important for a node to know, at now, which next hop neighbor nodes can achieve better aggregation performance while surmounting real-time necessity. On the other hand, to determine the next hop, we must satisfy real-time and data aggregation requirements. Hence, two questions are posed. First, what is the real-time policy to implement data aggregation for delay-constraint packets. Second, are there any nodes in the radio range of the current node that have homogeneous elements, i.e., data of the same type.

To make real-time decisions, before current node C forwards a packet, it computes the required velocity based on the progress made toward the sink node and the packet's TTD, as follows:

$$V_{req} = \frac{d(C,Sink)}{\mathrm{TTD}} \quad (2)$$

where $d(C,Sink)$ is the Euclidean distance between node C and the sink node. It is important to note that the deadline is met if the required velocity is met at each hop [18]. Hence, the problem of meeting end-to-end deadlines is mapped to the local problem of meeting the required velocity at each hop. This policy considers the current network conditions to adapt the packet's required velocity. If a packet is late in its way to the sink node, then its required velocity increases so that it may catch up. Conversely, its required velocity decreases if the packet is early.

Based on the velocity requirement and the information provided for the estimated delay EHD, node N in the neighbor set is an eligible forwarding choice if it is closer to the destination and the velocity it provides, $V_{relay}(C,N)$, is equal to or greater than the packet's required velocity V_{req} [28]. Relay velocity is calculated by dividing the advance in the distance to the next hop relay node by the estimated delay to forward the packet to that node [20]:

$$V_{relay}(C,N) = \frac{d(C,Sink)-d(N,Sink)}{\mathrm{EHD}} \quad (3)$$

where EHD, as we mentioned before, is the time it takes to forward a packet from a current node to the next hop relay node.

Moreover, to achieve a high aggregation ratio, RDA makes a dynamic forwarding decision by providing the freedom for the MAC layer to decide among a set of nodes (rather than a single next-hop). Therefore, it can achieve efficient spatial convergence by exploiting the information about the existence of same-type data in neighbor nodes. Our dynamic routing scheme uses only the local information of a node to make the routing decisions, therefore it is simple and scalable.

Anycasting [4] increases aggregation efficiency by determining the next-hop node for each transmission at the MAC layer. In anycasting, CTS responses are elicited from neighbor nodes using RTS packets before data transmission. Hence, the node which is the most benefic for data aggregation obtains the highest priority and sends CTS first. RTS contains the type and TTD of the transmitting packet. However, considering that the application is delay-constrained, a node in our routing scheme can respond with CTS provided that it is an eligible forwarding choice. On the other hand, from the nodes in the neighborhood, a node is eligible for CTS transmission if the real-time requirement has been satisfied.

Next hop relay nodes are prioritized based on real-time policy and aggregation efficiency. It potentially causes the receivers to randomly delay the CTS transmission to avoid CTS collision.

Choosing a shorter random delay before sending the CTS is a sign of higher priority. Moreover, nodes will cancel their CTS transmission if they overhear any CTS during the prioritized delay, so multiple CTS transmissions can be prevented because of the interference between the neighbors of the sender.

Eligible forwarding nodes are assigned three classes of priorities in responding to RTS as follows:

- Class A: The forwarder has some packets of the same type. Moreover, the minimum TTD of the packets in the aggregation queue is less than or equal to the TTD value as specified in RTS, so it has no effect on timing control of probably many packets for efficient aggregation.
- Class B: The forwarder has some packets of the same type and also, the minimum TTD of the packets is greater than the TTD value as specified in RTS.
- Class C: The forwarder does not have any packets of the same type. Here, we decide only based on the real-time routing policy in the sequel to assign the priority.

To avoid a collision between these three classes of neighbors three different slots are reserved in responder nodes for sending CTS packets. This policy is implemented as the priority of Class A is over Class B, and Class B is over Class C (Fig. 3). Moreover, to avoid a collision between nodes in the same class, they select a mini-slot for CTS transmission. As in [4], the main goal actually is to stagger the starting time of CTS transmissions based on the priorities using the slots and mini-slots. Note that the actual transmission time of the CTS could be larger than the mini-slot or slot time. However, considering the assumption of interference between neighbors, we expect only the first CTS transmission to succeed since the resulting interference will suppress the transmissions of the others.

Among the forwarding nodes in the same class, A or B, a higher priority (earlier mini-slot) is given to the node in which the minimum TTD is closer to the TTD value as specified in RTS. Moreover, node i with a higher fitness value, Fit_i, has a higher priority than the others in class C.

$$Fit_i = (w) \times \left(1 - \left(\frac{V_{req}}{V_{relay}(C,i)}\right)\right) + (1-w) \times \left(\frac{E_{Rem}(i)}{E_{Init}(i)}\right) \quad (4)$$

where $E_{Rem}(i)$ and $E_{Init}(i)$ are the residual and initial energy of eligible node i. Our algorithm uses factor $(1-w)$ to provide energy awareness in real-time routing as packets get closer to the sink

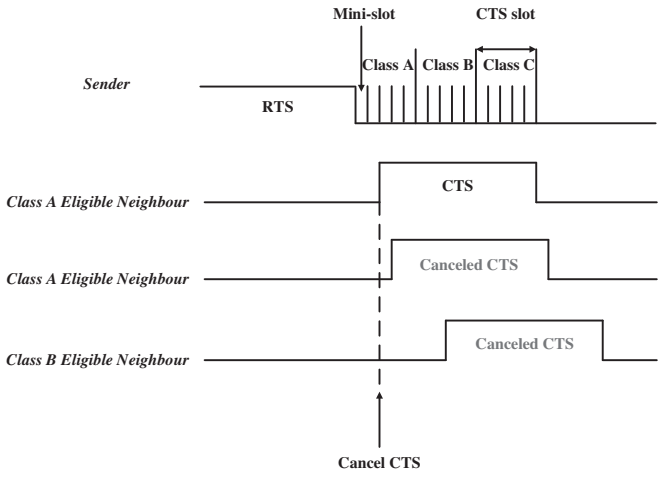

Fig. 3. Anycast based RTS/CTS [4]. Nodes select different slots or mini-slots for CTS transmission based on the priorities.

based on the bottleneck sphere theorem [29]. This bottleneck is placed near the sink node location where all nodes have the highest energy consumption. When all eligible nodes in the bottleneck sphere fail due to the depletion of energy, the sensing data outside this sphere will not reach the sink on time, which causes quality failure. Hence, we select the weight w, formally as:

$$w = \frac{\mathrm{TTD}}{\mathrm{TTD}_{\max}} \quad (5)$$

where $\mathrm{TTD}_{\max}$ is the packet end-to-end deadline. As packets get closer to the sink, the value of $(1-w)$ increases, so the effect of residual node energy is more highlighted in routing decisions. Using this factor enables the even balance of traffic between the eligible nodes along the path to the sink node, and especially in the bottleneck sphere.

At last, a node will be picked out of the eligible neighbor nodes while its priority is the highest. However, if there is no eligible node in the neighbor set, the back pressure rerouting mechanism is aimed [20].

Using the RDA approach can result in aggregation as early as possible on the routes to the sink. Moreover, RDA is tolerant to interference and node failures due to the dynamic routing in intermediate nodes; therefore, it is very robust even in unreliable WSNs. However, in the RDA approach, packets may not be aggregated if they are spatially separated, so we use the timing control policy for temporal convergence (see Section 3.2.1) to improve aggregation efficiency as the intermediate nodes delay the packets. Thereupon, using the above approach, packets will converge to the best aggregation points dynamically without explicit construction and maintenance of an aggregation structure while considering real-time necessity for efficient delay-constraint data forwarding.

3.3. Energy consumption model and aggregation policy

In this section, we evaluate the energy consumption ratio at node i, E_i, while comparing both the case when the node simply relays other nodes' packets and/or generating its own traffic as a forwarder, and the case when the node acts as an aggregator. In the first case, simple forwarding, the energy cost at node i is computed as follows:

$$E_i = (E_{TX} + E_{RX}) \times \sum_{l=1}^{N_i} \lambda_l + E_{TX} \times r_i + E_{sense} \times r_i \quad (6)$$

where λ_l is the packet reception rate at the forwarder node i imposed by each of the N_i one-hop neighbors. E_{sense} is the energy depleted for sensing and r_i is the rate of data originally generated by node i. Moreover, E_{TX} and E_{RX} are the packet transmission and reception energies, respectively. In the transmitting mode, energy is spent in the electronic components, E_{elec}, as well as in the front-end amplifier, E_{amp}, which supplies the power for the actual RF transmission. In the receiving mode, energy is consumed entirely by the transceiver electronics, E_{elec}.

However, packets could be effectively aggregated to reduce the energy consumption by eliminating the inherent redundancy of raw data. Hence, a FIFO (First In First Out) queue with a size of D is employed to hold packets for aggregation [30]. D actually represents the maximum number of packets whose information can be aggregated by node i into one packet. The packets wait at the queue until the number of accumulated packets is equal to the maximum aggregation limit D.

Moreover, according to Judiciously Waiting policy, different TTDs of the incoming packets make their waiting time, $\mathrm{WT}_{i,j}$, different where j is the packet position in the aggregation queue of node i. If $\mathrm{WT}_{i,j}$ for an incoming packet is less than the aggregation timeout value, the aggregation timer changes to $\mathrm{WT}_{i,j}$. Hence, this value

is continuously readjusted to the minimum waiting time of the incoming packets, $\min(WT_{i,j})$, if necessary, to meet their deadlines. Thus, the number of the aggregated packets depends on how soon the timer expires. Finally, the packets accumulated in the aggregation queue are flushed when the queue is full or the timer expires. After expiration, the aggregated result has a TTD equal to the minimum TTD of the packets and then is sent to the next hop forwarding node. Thus, in the case when node i is an aggregator, the energy consumption ratio can be written as follows:

$$E_i = (E_{TX} + E_{agg}) \times R_i + E_{RX} \times \sum_{l=1}^{N_i} \lambda_l + E_{sense} \times r_i \quad (7)$$

where E_{agg} is the energy employed for data aggregation and R_i is the forwarding rate of node i calculated via:

$$R_i = \frac{1}{\min\left\{WT_{i,j}\ (1 \leqslant j < D) \forall j, \frac{D}{\sum_{l=1}^{N_i} \lambda_l + r_i}\right\}} \quad (8)$$

where $\frac{D}{\sum_{l=1}^{N_i} \lambda_l + r_i}$ is the time it takes to meet aggregation limit D.

4. Performance evaluation

In this section, we evaluate the performance of our algorithm via simulation in NS-2 [31]. The goal of the simulation is to show that our proposed protocol, RAG, can outperform other important real-time routing and dynamic data aggregation protocols using two mechanisms for temporal and spatial convergence of delay-constrained packets – Judiciously Waiting policy and Real-time Data-aware Anycasting policy. The results are compared with SPEED [20], the most well-known real-time routing protocol, to show aggregation efficiency of RAG and also DAA + RW [4] and DASDR [5], two recent dynamic protocols which use both spatial and temporal convergence for efficient data aggregation, to show the efficiency of RAG for real-time applications in WSNs. DAA + RW uses Data-Aware Anycast for spatial data convergence and also Randomized Waiting for aggregation scheduling. However, DASDR designs a dynamic routing scheme based on the depth-queue potential field which makes packets more spatially convergent while cooperating with cascading timeout policy for temporal data convergence.

The simulation parameters for our model are mentioned in Table 1. To incorporate the anycasting capability, the RTS/CTS packet formats of 802.11 MAC are modified. The RTS contains two extra fields of packet type and TTD value and the CTS packet contains an extra field of the address of the CTS sender. The communication parameters are mostly chosen in reference to the Berkeley mote specifications [32].

We ran the simulation with several parameters, including data rate and aggregation limit, where 6 nodes randomly chosen from the left side of the terrain, send periodic data to the sink placed at the middle of the right side of the terrain. To create a dynamic event-based environment, each source node assigns a random type to its periodically generated data packet. We define 3 types of data and only same-type packets could be aggregated in the intermediate nodes along the way to the sink node. The simulation ends as soon as 5000 packets are received at the sink. Therefore, we add a field to each aggregated packet to specify how many effective pieces of information are contained in this packet. We derived different aggregation limits $(4, 8, \infty)$ from different values of aggregation ratio $(0, 0.5, 1)$ in [4]. Moreover, we set $\alpha = 0.7$, as in [28], and increase the packet generation rate step by step from 1 to 100 pps (packets/s) and then compare RAG with the existing protocols in the terms of average energy consumption, miss ratio, end-to-end delay, and aggregation gain where:

- Miss ratio, the most important metric in real-time systems, is defined as the percentage of packets that do not meet their end-to-end deadlines and hence be discarded during transmission to the sink node.
- End-to-end delay is defined as the duration from a packet (event) being generated in the source till it is received by the sink. Here, we derived this delay from reducing the TTD value of the deadline as soon as a packet is received at the sink node.
- Aggregation gain is defined as the measure of reduction in the communication traffic due to the aggregation. Thus, it is the ratio of traffic reduction due to aggregation to the total traffic without aggregation [8].

From Figs. 4 and 5 it can be seen that RAG outperforms SPEED in terms of energy consumption and miss ratio, especially in high data rates, where generated packets have the end-to-end deadline of 400 ms. In SPEED no packet gets aggregated along the way to the sink; thus, more packets are injected into the network. However, RAG uses a Judiciously Waiting policy to take advantage of the

Table 1
Simulation parameters

Terrain	200 m × 200 m
Node number	100
Topology	Grid
Initial node energy	0.6 J
Bandwidth	200 Kb/s
Radio range	40 m
Propagation model	Two ray
Payload size	50 bytes
E_{elec}	50 nJ/bit
E_{sense}	0.083 J/s
E_{amp}	10 pJ/bit/m^2
E_{agg}	5 nJ/bit/signal

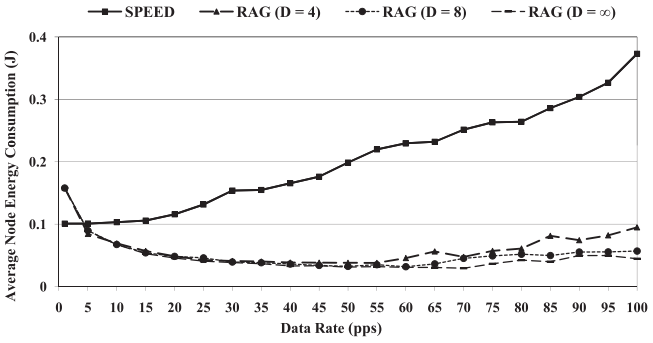

Fig. 4. Average node energy consumption vs. data rate.

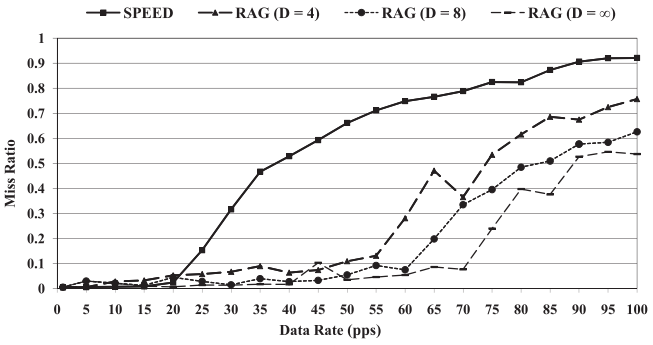

Fig. 5. Miss ratio vs. data rate.

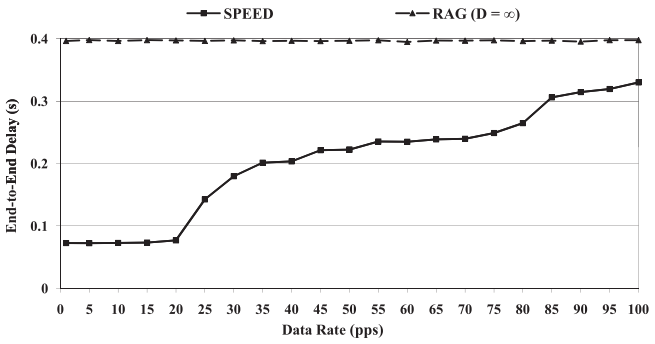

Fig. 6. End-to-end delay vs. data rate.

available slack for efficient aggregation, so it conserves energy by eliminating the inherent redundancy of raw data. As the traffic increases, miss ratio increases as well because more contention can result in longer queueing delays and congestion in intermediate nodes. A higher number of missed packets can lead to a higher number of retransmissions as well as generated packets in the source nodes, and finally, a higher energy consumption in heavy traffic loads. Considering that there is no aggregation opportunity in the low traffic, the energy consumption would decrease in our algorithm as the data rate increases. However, it calls to increase as the miss ratio increases in high data rates where heavy traffic are congested. Moreover, by increasing the aggregation limit in our protocol more number of same-type packets can be aggregated into one packet, so the injected traffic, the miss ratio, and the energy consumption decrease. The fluctuation in the curves of our proposed method is justifiable considering that RAG selects different routes in different traffic loads for data forwarding in a real-time aggregation-aware scheme, so there is a kind of randomness in routing.

Moreover, Fig. 6 compares SPPED and RAG protocols in terms of end-to-end delay in different traffic loads for an unlimited aggregation queue. As it is evident, RAG has a higher end-to-end delay than SPEED. Specifically, RAG takes advantage of the available slack to improve real-time performance and delays packets at every hop for a duration of time to efficiently aggregate data. Hence, in the proposed method, the packets are delivered to the sink with the maximum end-to-end delay while meeting the delay constraint.

Fig. 7 illustrates the aggregation ability of different dynamic aggregation protocols. Actually, it depicts the efficiency of waiting time policy in terms of aggregation gain for various aggregation limits under different traffic loads when the end-to-end deadline is 1 s. As expected, when the data rate increases, more packets have the chance to meet each other, so the aggregation ability increases. However, aggregation gain calls to be reduced after a threshold value where heavy traffic is congested, so the packets

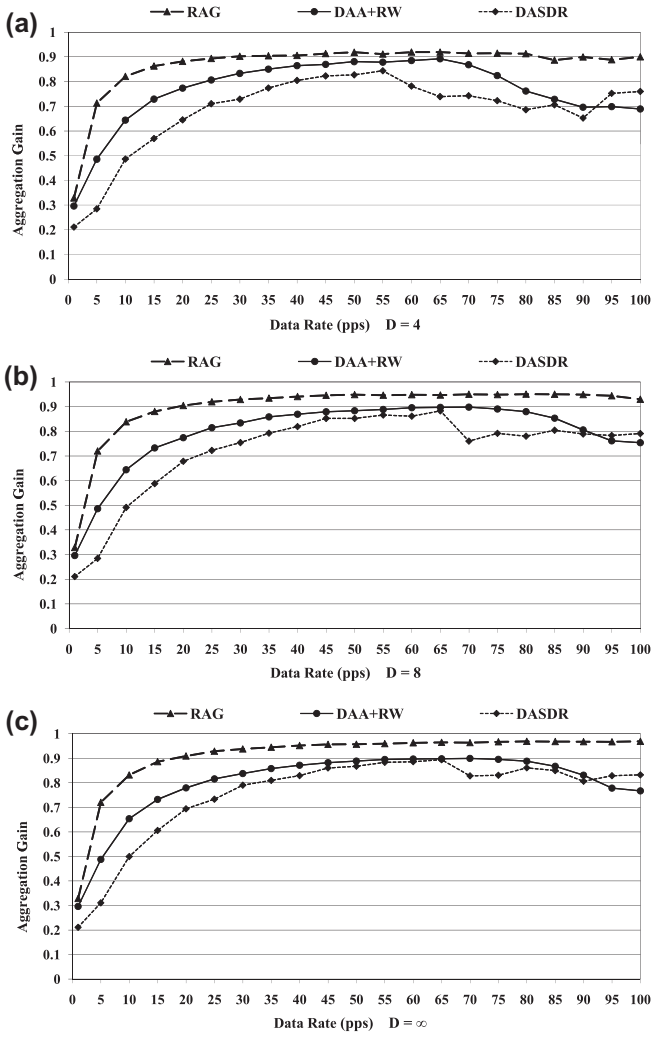

Fig. 7. Aggregation gain vs. data rate (a) $D = 4$, (b) $D = 8$, (c) $D = \infty$.

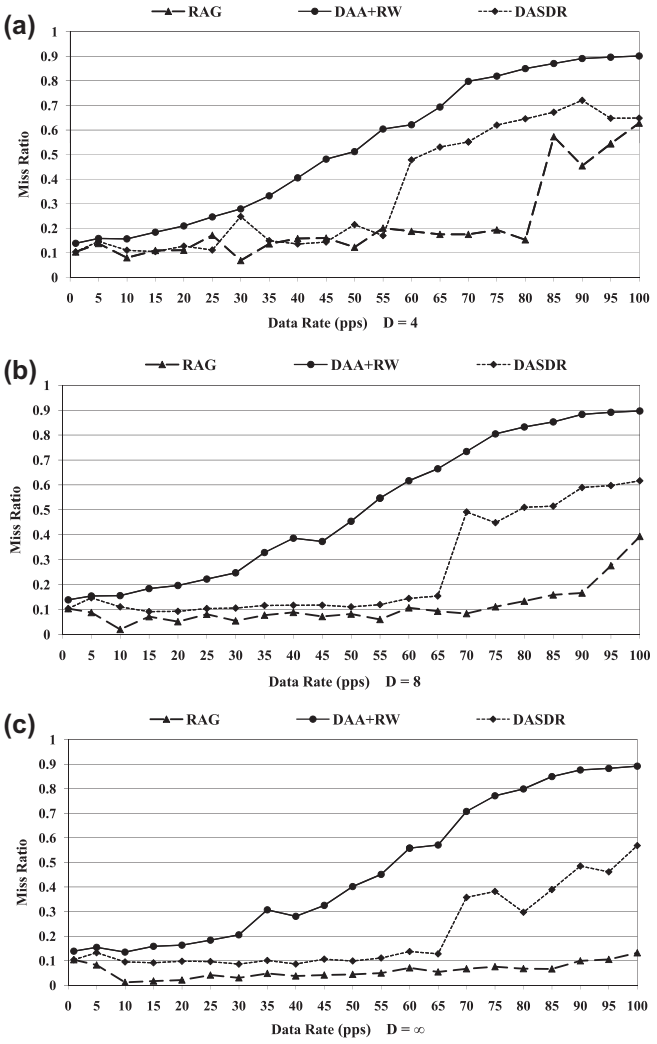

Fig. 8. Miss ratio vs. data rate (a) $D = 4$, (b) $D = 8$, (c) $D = \infty$.

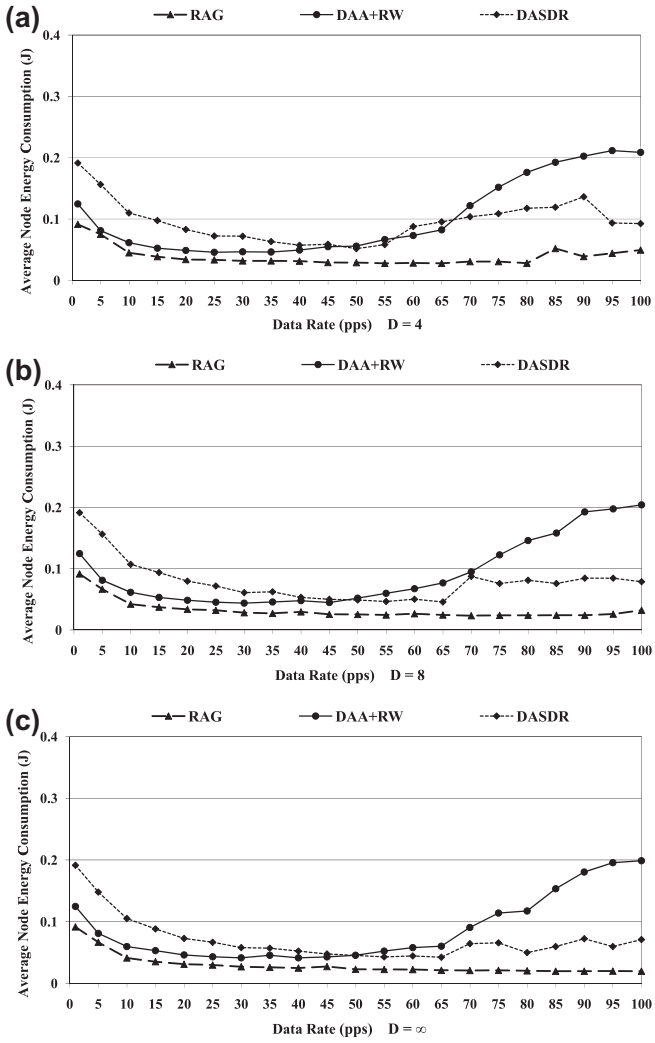

Fig. 9. Average node energy consumption vs. data rate (a) $D = 4$, (b) $D = 8$, (c) $D = \infty$.

do not have any considerable slack time to use for data aggregation. Moreover, our proposed algorithm outperforms DASDR and DAA + RW in terms of aggregation gain (on average, by 27% and 15% in the unlimited aggregation queue scenario, respectively) since in our method, each aggregator node has an accurate estimation of delays along the ways to the sink node, hence it could collect more data from its neighbors in a judicious manner. However, in DAA + RW and DASDR the scheduling is based on random waiting and cascading timeout policies which cannot appropriately delay packets in intermediate nodes, thus they have less aggregation gain. Moreover, DAA + RW routes the packets only based on queues potential, so it gives a better aggregation gain compared to DASDR.

Fig. 8 shows the packet miss ratio for various aggregation limits under different data rates. As it can be seen, by increasing the data rate, the miss ratio increases as well for all different schemes. However, RAG outperforms the others by combining both real-time routing and timing control for delay-constrained data aggregation. Moreover, our scheme reduces the number of packets injected into the network in intermediate nodes from sources to the sink by providing a higher data aggregation at the nodes closer to the source nodes. Thus, RAG can handle the number of transmissions and the network traffic more effectively as the data rate increases. Actually, RAG, on average, gives an overall improvement of 64% and 83% over DASDR and DAA + RW, respectively, in terms of miss ratio in the unlimited aggregation queue scenario.

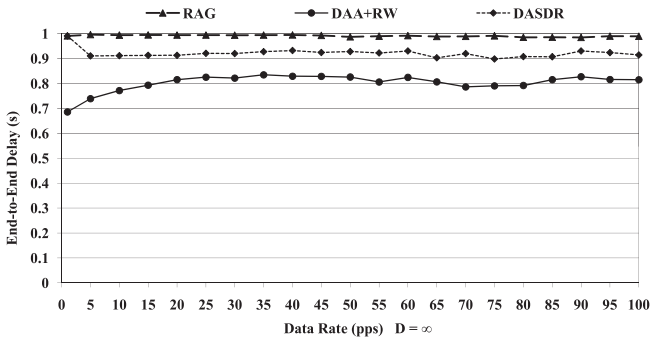

Fig. 10. End-to-end delay vs. data rate ($D = \infty$).

Moreover, as Fig. 9 shows, RAG has a higher energy efficiency than DAA + RW and DASDR (on average, by 55% and 58% in the unlimited aggregation queue scenario, respectively), especially in the high traffic loads due to using an efficient real-time data aggregation policy which decreases the miss ratio, and increases the aggregation gain as well as the network lifetime.

Moreover, we evaluate the performance of RAG for achieving temporal convergence. Fig. 10 shows average end-to-end delay of protocols in different traffic loads for an unlimited aggregation queue. As it is evident, RAG is taking advantage of available slacks for efficient aggregation, so the packets are delivered to the sink with the maximum end-to-end delay while meeting the delay constraint.

Thus, considering the delay constraint, RAG uses a Judiciously Waiting policy for temporal convergence and a Real-time Data-aware Anycasting policy for spatial convergence to increase the aggregation gain and energy efficiency and decrease the miss ratio as much as possible.

5. Conclusion

Data aggregation is a promising technique to reduce energy consumption and prevent congestion in WSNs. In this paper, we considered the problem of real-time data aggregation in WSNs and proposed a new method combining temporal and spatial convergence of packets using Judiciously Waiting policy and Real-time Data-aware Anycasting policy, respectively, without explicit maintenance of a structure. Results evaluated in simulation demonstrated a significant performance improvement in terms of energy consumption, miss ratio, and aggregation gain. The algorithm could be modified to take into account some aspects that have not been addressed in this work, which can be an interesting subject of future research. For instance, studying an aggregation-aware real-time routing protocol in mobile WSNs can be considered in future studies.

References

[1] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci, A survey on sensor networks, IEEE Communications Magazine 40 (2002) 102–114.
[2] E. Fasolo, M. Rossi, J. Widmer, M. Zorzi, In-network aggregation techniques for wireless sensor networks: a survey, IEEE Wireless Communications 14 (2007) 70–87.
[3] P. Wan, S.C. Huang, L. Wang, Z. Wan, X. Jia, Minimum-latency aggregation scheduling in multihop wireless networks, in: Proceedings of the 10th ACM Interational Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc'09), pp. 185–194.
[4] K.W. Fan, S. Liu, P. Sinha, Structure-free data aggregation in sensor networks, IEEE Transactions on Mobile Computing 6 (2007) 929–942.
[5] J. Zhang, Q. Wu, F. Ren, T. He, C. Lin, Effective data aggregation supported by dynamic routing in wireless sensor networks, in: Proceedings of the IEEE International Conference on Communications (ICC'10), pp. 1–6.

[6] W.R. Heinzelman, A. Chandrakasan, H. Balakrishnan, Energy-efficient communication protocol for wireless microsensor networks, in: Proceedings of the 33rd Annual Hawaii International Conference on System Sciences.
[7] O. Younis, S. Fahmy, HEED: A hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks, IEEE Transactions on Mobile Computing 3 (2004) 366–379.
[8] J.Y. Choi, J.W. Lee, K. Lee, S. Choi, W.H. Kwon, H.S. Park, Aggregation time control algorithm for time constrained data delivery in wireless sensor networks, in: Proceedings of the 63rd IEEE Vehicular Technology Conference, pp. 563–567.
[9] R. Cristescu, B. Beferull-Lozano, M. Vetterli, On network correlated data gathering, in: Proceedings of the 23rd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'04), pp. 2571–2582.
[10] M. Ding, X. Cheng, G. Xue, Aggregation tree construction in sensor networks, in: Proceedings of the 58th IEEE Vehicular Technology Conference, pp. 2168–2172.
[11] H. Du, X. Hu, X. Jia, Energy efficient routing and scheduling for real-time data aggregation in WSNS, Computer Communications 29 (2006) 3527–3535.
[12] Y. Hu, N. Yu, X. Jia, Energy efficient real-time data aggregation in wireless sensor networks, in: Proceedings of the International Conference on Wireless Communications and Mobile Computing (IWCMC'06), pp. 803–808.
[13] H. Li, H. Yu, B. Yang, A. Liu, Timing control for delay-constrained data aggregation in wireless sensor networks, Communication Systems 20 (2007) 875–887.
[14] H.F. Salama, D.S. Reeves, Y. Viniotis, Evaluation of multicast routing algorithms for real-time communication on high-speed networks, IEEE Journal on Selected Areas in Communications 15 (1997) 332–345.
[15] I. Solis, K. Obraczka, The impact of timing in data aggregation for sensor networks, in: Proceedings of the IEEE International Conference on Communications (ICC' 04), pp. 3640–3645.
[16] H.O. Tan, I. Korpeoglu, Power efficient data gathering and aggregation in wireless sensor networks, SIGMOD Record 32 (2003) 66–71.
[17] C.M. Chao, T.Y. Hsiao, Design of structure-free and energy-balanced data aggregation in wireless sensor networks, in: Proceedings of the 11th IEEE International Conference on High Performance Computing and Communications (HPCC'09), pp. 222–229.
[18] O. Chipara, Z. He, G. Xing, Q. Chen, X. Wang, C. Lu, J. Stankovic, T. Abdelzaher, Real-time power-aware routing in sensor networks, in: Proceedings of the 14th IEEE International Workshop on Quality of Service (IWQoS'06), pp. 83–92.
[19] E. Felemban, C. Lee, E. Ekici, MMSPEED: Multipath multi-speed protocol for QoS guarantee of reliability and timeliness in wireless sensor networks, IEEE Transactions on Mobile Computing 5 (2006) 738–754.
[20] T. He, J.A. Stankovic, C. Lu, T. Abdelzaher, SPEED: A stateless protocol for real-time communication in sensor networks, in: Proceedings of the 23rd International Conference on Distributed Computing Systems, pp. 46–55.
[21] K. Kim, S. Park, H. Park, Y.H. Ham, Reliable and real-time data dissemination in wireless sensor networks, in: Proceedings of Military Communications Conference (MILCOM'08), pp. 1–5.
[22] Y. Li, C.S. Chen, Y. Song, Z. Wang, Y. Sun, Enhancing real-time delivery in wireless sensor networks with two-hop information, IEEE Transactions on Industrial Informatics 5 (2009) 113–122.
[23] C. Lu, B.M. Blum, T.F. Abdelzaher, J.A. Stankovic, T. He, RAP: A real-time communication architecture for large-scale wireless sensor networks, in: Proceedings of the 8th IEEE Real-time and Embedded Technology and Applications Symposium.
[24] H. Peng, Z. Xi, L. Ying, C. Xun, G. Chuanshan, An adaptive real-time routing scheme for wireless sensor networks, in: Proceedings of the 21st International Conference on Advanced Information Networking and Applications Workshops (AINAW'07), pp. 918–922.
[25] W. Zhang, G. Cao, DCTC: Dynamic convoy tree-based collaboration for target tracking in sensor networks, IEEE Transactions on Wireless Communications 3 (2004) 1689–1701.
[26] K.W. Fan, S. Liu, P. Sinha, Dynamic forwarding over tree-on-dag for scalable data aggregation in sensor networks, IEEE Transactions on Mobile Computing 7 (2008) 1271–1284.
[27] A.F. Harris, R. Kravets, I. Gupta, Building trees based on aggregation efficiency in sensor networks, Ad Hoc Networks 5 (2007) 1317–1328.
[28] K. Liu, N. Abu-Ghazaleh, K. Kang, JiTS: Just-in-time scheduling for real-time sensor data dissemination, in: Proceedings of the 4th Annual IEEE International Conference on Pervasive Computing and Communications (PerCom'06).
[29] P. Zeng, H. Yu, An ant-based routing algorithm to achieve the lifetime bound for target tracking sensor networks, in: Proceedings of the INFOCOM'06.
[30] L. Galluccio, S. Palazzo, A.T. Campbell, Modeling and designing efficient data aggregation in wireless sensor networks under entropy and energy bounds, IJWIN 16 (2009) 175–183.
[31] The network simulator, <http://www.isi.edu/nsnam/ns>.
[32] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, K. Pister, System architecture directions for network sensors, in: Proceedings of ASPLOS'00, pp. 93–104.