Available online at www.sciencedirect.com
ScienceDirect
Journal of Network and
Computer Applications 31 (2008) 387–401
Journal of NETWORK and COMPUTER APPLICATIONS
www.elsevier.com/locate/jnca

Data aggregation in wireless sensor networks using ant colony algorithm

Wen-Hwa Liao*, Yucheng Kao, Chien-Ming Fan

Department of Information Management, Tatung University, No. 40, Sec. 3, Chungshan N. Road, Taipei 104, Taiwan

Received 24 August 2007; received in revised form 28 February 2008; accepted 28 February 2008

Abstract

Data aggregation is important in energy constraint wireless sensor networks which exploits correlated sensing data and aggregates at the intermediate nodes to reduce the number of messages exchanged network. This paper considers the problem of constructing data aggregation tree in a wireless sensor network for a group of source nodes to send sensory data to a single sink node. The ant colony system provides a natural and intrinsic way of exploring search space in determining data aggregation. Moreover, we propose an ant colony algorithm for data aggregation in wireless sensor networks. Every ant will explore all possible paths from the source node to the sink node. The data aggregation tree is constructed by the accumulated pheromone. Simulations have shown that our algorithm can reduce significant energy costs.

Keywords: Ant colony; Data aggregation; Wireless sensor networks

1. Introduction

Recent advances in wireless communications and electronics have enabled the development of low-cost, low-power, multifunctional sensor nodes that are small in size and communicate in short distances. Wireless sensor networks are becoming a rapidly developing area in both research and application. Although it was originally driven by military applications, wireless sensor networks are being investigated and applied in many

*Corresponding author. Tel.: +886 2 25925252x3608; fax: +886 2 25853966.
E-mail address: whliao@ttu.edu.tw (W.-H. Liao).

1084-8045/$ - see front matter
doi:10.1016/j.jnca.2008.02.006

different civilian applications. For example, the sensor network has been applied for vehicle tracking system, habitat monitoring, forest surveillance, earthquake observation, biomedical or health care applications and building surveillance. A wireless sensor network consists of a number of sensor nodes scattered in the region of interest in order to acquire some physical data. The sensor node should have the ability of sensing, processing and communicating (Akyildiz et al., 2002; Arampatzis et al., 2005; Culler et al., 2004). A wireless sensor network operates in an unattended environment, with limited computational and sensing capabilities, and capable of sensing, computing and communicating wirelessly. In order to effectively utilize wireless sensor nodes, we need to minimize energy consumption in the design of sensor network protocols and algorithms. A large number of sensor nodes have to be networked together. Direct transmission from any specified node to a distant sink node is not used since sensor nodes that are farther away from the sink node will drain their power sources much faster than those nodes that are closer to the sink node. On the other hand, a minimum energy multi-hop routing scheme will rapidly drain the energy resources of the nodes, since these nodes are engaged in the forwarding of a large number of data messages (on behalf of other nodes) to the sink node. Thus, one solution is to use multi-hop communication with in-network aggregation of correlated data (Luo et al., 2006). The application of an aggregation approach helps to reduce the amount of information that needs to be transmitted by performing data fusion at the aggregate points before forwarding the data to the end user (Li et al., 2006). The ant colony optimization is one of the most successfully proven swarm intelligence. It has been successfully applied in many difficult discrete optimization problems such as the traveling salesman problem (Dorigo and Gambardella, 1997), scheduling, vehicle routing, etc., as well as routing in wireless networks (Chen et al., 2006; Ducatelle et al., 2005; Okdem and Karaboga, 2006). In this paper, we proposed a data aggregation mechanism in wireless sensor networks using ant colony algorithm. The proposed mechanism assigned artificial ants to source nodes to establish low-latency paths between the source nodes and the sink node. Paths from different source nodes to a sink node form an aggregation tree rooted at the sink node. Data from different sources are opportunistically aggregated. Whenever similar data happen to meet at a branching node in the tree, the copies of similar data are replaced by a single message. Energy wise, opportunistic aggregation on a low-latency tree is not optimal because data may not be aggregated near the sources. Hence, our mechanism will extend the routing path to increase the probability of intersection of routing paths. If any node receives the data from a neighboring node, the node will select next node according to the random-proportional rule of our ant colony algorithm. After a short transitory period, the amount of pheromone on the aggregation nodes is sufficiently large to guide ants (the data packets from different sources) to meet together at these nodes for data aggregation.

The rest of the paper is organized as follows. Section 2 briefly describes the related work in data aggregation for wireless sensor networks. Our proposed ant colony algorithm for data aggregation in wireless sensor networks is presented in Section 3. Experimental results are summarized in Section 4, while Section 5 presents the conclusion of this paper.

2. Related work

A wireless sensor network consists of many small sensors with limited energy resources and thus, requires novel data dissemination paradigms to save on network energy.

Directed diffusion (DD) (Intanagonwiwat et al., 2003) is a typical data-centric routing paradigm for sensor networks. It consists of four basic elements: interests, data messages, gradients, and reinforcements. In DD, a task, which is a list of attribute-value pairs, is flooded into the whole network as an interest for named data. A gradient is a direction state created in each node that receives an interest. Events start flowing toward the originator of interest along the established shortest path. Data from different sources are opportunistically aggregated. However, opportunistic aggregation on a low-latency tree is not efficient because data may not be aggregated on nodes near the sources.

In-network data aggregation is an important in energy constraint sensor network which exploits correlated sensing data and aggregates at the intermediate nodes reducing the number of messages exchanged network. In data gathering application large amount of communication is reduced by in-network aggregation achieving maximum lifetime of network. Optimal aggregation tree problem is NP-hard (Al-Karaki et al., 2004) which is equivalent to Steiner tree (Krishnamachari et al., 2002), weighted set cover (Intanagonwiwat et al., 2002) problems. Many researchers have made efforts on data aggregation in wireless sensor networks (Bhattacharjee and Das, 2007; Intanagonwiwat et al., 2002; Krishnamachari et al., 2002; Li et al., 2006; Misra and Mandal, 2006; Motegi et al., 2006).

In Li et al. (2006), their greedy algorithm constructs a multicast tree by iteratively adding source nodes to the existing tree until all the source nodes and the sink node are included. Initially, the tree includes only the sink node. Each time the algorithm finds a source node among the remaining source nodes, which is closest to the existing tree, it adds the shortest path between that source node and the existing tree to the tree. This process continues until all the source nodes have been included in the tree. In addition, in order to further reduce the computational complexity and improve the quality of the output solution, they design another heuristic approximation algorithm based on minimum spanning tree to construct the data aggregation tree.

In Intanagonwiwat et al. (2002), they studied the energy efficiency of the greedy aggregation, which is different from the previous diffusion approach for opportunistic aggregation on the lowest latency tree. Greedy aggregation constructs a greedy incremental tree (GIT) as follows: the shortest path is established only for the first source to the sink node, while all the other sources is incrementally connected to the closest node in the existing tree. Simulation showed that greedy aggregation saved on energy cost considerably as compared to opportunistic aggregation without any adverse impact on latency or robustness.

In Misra and Mandal (2006), they applied the ant colony optimization algorithm to solve the data aggregation problem and referred to as the ant-aggregation algorithm. In a wireless sensor network with multiple sources and a single destination, artificial ants are assigned to source nodes to construct paths for transmitting data packets to the sink node. The shorter is the path between the source node and the aggregation node, the more pheromone ants will deposit on it. The algorithm uses the Euclidean distance formula to compute for the distance from a source node to an aggregation node and from an aggregation node to a sink node. However, the computed Euclidean distance may not be applicable in wireless sensor networks because the communication range of a node is limited. Since nodes can communicate only with their one-hop neighbor, the Euclidean distance between the source node and the sink node is not reliable. In Fig. 1, the employed Euclidean distance may not communicate directly with the aggregation node. In fact,

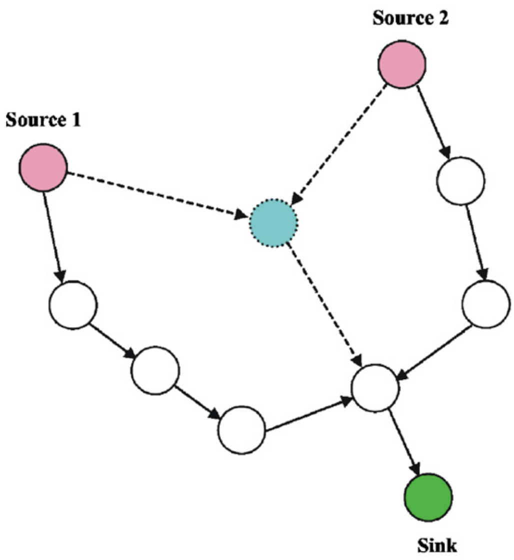

Fig. 1. The employed Euclidean distance may not communicate directly with the aggregation node.

sources must communicate indirectly with the sink node through the middle nodes. Hence, our algorithm uses the hop count instead of the Euclidean distance.

3. An ant colony algorithm for data aggregation

In the ant colony system, a colony of artificial ants is used to construct solutions guided by the pheromone trails and heuristic information (Dorigo and Gambardella, 1997). Ant colony system was inspired by the foraging behavior of real ants. This behavior enables ants to find the shortest paths between food sources and their nest. Initially, ants explore the area surrounding their nest in a random manner. As soon as an ant finds a source of food, it evaluates the quantity and quality of the food and carries some of it to the nest. During the return trip, the ant deposits a pheromone trail on the ground. The quantity of deposited pheromone, which may depend on the quantity and quality of the food, will guide the other ants to the food source. The indirect communication between the ants via the pheromone trails allows them to find the shortest path between their nest and the food sources. This functionality of real ant colonies is exploited in artificial ant colonies in order to solve optimization problems. In the ant colony system, the pheromone trails are simulated via a parameterized probabilistic model called the pheromone model. The pheromone model consists of a set of model parameters whose values are called the pheromone values. The basic ingredient of the ant colony system is a constructive heuristic that is used for the probabilistic construction of solutions using the pheromone values (Misra and Mandal, 2006).

In Misra and Mandal (2006), they proposed an ant colony algorithm for data aggregation. However, the probability of finding aggregation nodes based on any two routing paths is not high. To increase the probability, this paper suggests an extension of the search region around the routing paths. For example, in Fig. 2, there is no aggregation

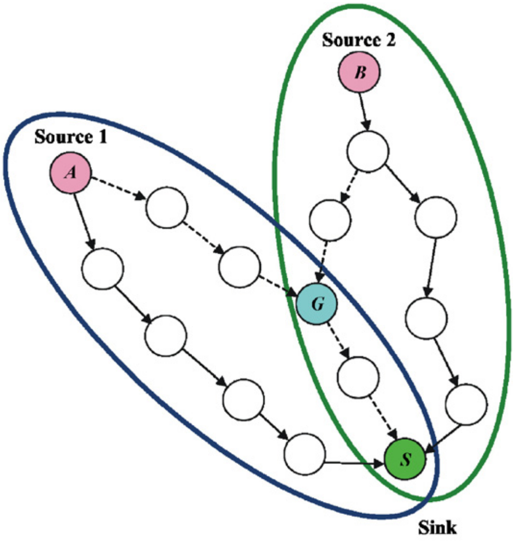

Fig. 2. The aggregation node G will be identified when the search regions are extended.

node between the path from the source node 1 to the sink node and the path from the source node 2 to the sink node (indicated in solid lines). If the search regions of the two paths are extended individually, the aggregation node G will be found and two new routing paths (indicated in dashed lines) will be formed.

In wireless sensor networks, each sensor node has a unique identity. Wireless sensor networks can be modeled by a unit disk graph. A random unit disk graph $G = (V, E)$ can be constructed as follows. Place V nodes at random in a square area and connect them with their neighbors at the Euclidean distance less than or equal to the transmission range R. Initially, every node has a pheromone table with its one-hop neighboring nodes. The pheromone of the path, which does not deliver data gets evaporated with time.

Ants are like agents placed on the cities of the traveling salesman problem graph (Dorigo and Gambardella, 1997). They move from city to city to construct a complete Hamiltonian circuit of the graph. The ants' movement is guided by the pheromone trail and *a priori* heuristic information. In the ant colony system, every arc between two cities is marked with a pheromone strength number, and the ant agents update this number during the algorithm. Ants modify the number after all the ants have finished their tours. Depending on the cost of the tour, the lower the cost, the higher is the number that will be added. Ant agents choose the next move probabilistically according to the pheromone strength. In this way, the arc within the lower cost tour receives a higher pheromone strength number and thus, has a higher probability to be chosen by other ants in the later runs.

In wireless sensor networks, the ant colony algorithm assigns ants to source nodes (Misra and Mandal, 2006). The ants search the routes and communicate with the others through pheromones. Each ant iterates to construct the aggregation tree where the internal nodes are aggregate points. The ants either try to find the shortest route to the destination and terminate, or finds the closest aggregation point of the route searched by previous ants and terminates. The algorithm converges to the local best aggregation tree. In order to find the global optimal aggregation nodes, the algorithm iterates on the different nodes located within the extended routing paths.

In the following paragraphs, we will describe how to find the aggregation points using an ant colony algorithm and illustrate how our proposed algorithm works. The ant colony algorithm includes three steps. Step 1 is how to select next hop node; Step 2 is to extend the routing path; Step 3 is to update the pheromone trails on the sensor nodes.

3.1. Next hop node selection

First, the sink node will flood its identity to all the nodes in the network. After the node receives this packet, it will compute the hop count to the sink node. When the source node wants to send data, it will select the next hop node by following the random-proportional rule (see Eq. (1)) where ant k in the data packet in node i choose to move to the node j until the sink node,

$$p_k(i,j) = \frac{\tau(i,j) \times \eta(i,j)^{\beta}}{\sum_{u \in N_i} \tau(i,u) \times \eta(i,u)^{\beta}} \tag{1}$$

where $\tau(i, j)$ is the pheromone level from node i to node j, $\eta(i, j)$ is the inverse of the hop count from node j to the sink node adding one, N_i is the number of neighbors of the node i, and β is a parameter which determines the relative influence of heuristic values $\eta(i, j)$ ($\beta > 0$).

3.2. Extending the routing path

In order to increase the probability of intersection of the routing paths, our mechanism will extend the routing path. After the node i selects its next hop node by Eq. (1), it sends a *Select* packet including the data packet to the next node. The *Select* packet consists of six parts:

- *SN*: the selected node.
- *S*: the source id.
- *P*: the previous hop node id.
- D_{is}: the hop count from the node i to the source node.
- *EHC*: the extended hop count.
- *TTL*: the time to live.

The field of D_{is} is computed by the hop count from the source node to the node i. The initial value of D_{is} is 0. As the node j receives the *Select* packet, it will build a reaction table. The reaction table consists of five parts:

- *S*: the source id.
- *P*: the previous hop node id.
- D_{js}: the hop count from the node j to the source node.
- *EHC*: the extended hop count.
- *TTL*: the time to live.

The *S*, *P*, *EHC*, and *TTL* of the reaction table are of the same values as the *S*, *P*, *EHC*, and *TTL* of the *Select* packet, respectively. The D_{js} of the reaction table is the D_{is} of the *Select* packet adding one. The node, whose reaction table already exists, will check the *S* value of the *Select* packet with the *S* value of its reaction table. If both *S* values are the same, node *j* will update its reaction table by selecting the lower value of D_{js}; otherwise, it will add a new record for the *Select* packet.

After node *j* builds or updates the reaction table, it will broadcast a new *Select* packet without the data packet to its neighbors. The *SN* of this packet is set to "*", which is the one-hop broadcasting packet. The *TTL* of the *Select* packet is the *TTL* of the reaction table subtracting one. When the *TTL* value is equal to zero, node *j* will stop extending its routing path.

3.3. Pheromone updating rule

When the *EHC* is equal to the *TTL* of the node's reaction table, node *j* will send a *Pheromone_Update* packet to its parent, node *i*. The *Pheromone_Update* packet consists of four parts:

- *ID*: the node id.
- *P*: the previous hop node id.
- h_j: the hop count of the node *j* to sink node.
- $\Delta\omega_j$: the total cost of the sources reaching to the sink node through node *j*.

When node *i* receives a *Pheromone_Update* packet, it will update its pheromone table according to Eqs. (2)–(4):

$$\tau_{ij} = (1-\rho)\tau_{ij} + \rho\Delta\tau_{ij} \tag{2}$$

where

$$\Delta\tau_{ij} = [1 + (h_i - h_j)] \times \Delta\omega_j \tag{3}$$

where

$$\Delta\omega_j = \sum_{i\in R_j}(H_{ij})^{-1} + (h_j)^{-1} \tag{4}$$

where ρ is the pheromone evaporation parameter, h_i is the hop count between nodes *i* and the sink node, and h_j is the hop count between nodes *j* and the sink node.

In Eq. (2), the deposited pheromone is discounted by a factor of ρ; this results in the new pheromone level being a weighted average between the accumulated pheromone and the new added pheromone.

In Eq. (3), if the value of $(h_i–h_j)$ is greater than zero, then it can conclude that node *j* is closer to the sink node than node *i*. Hence, the algorithm will reward the path from node *i* to node *j* by depositing more pheromones. If the value of $(h_i–h_j)$ is equal to zero, then it means that both nodes *i* and *j* have the same hop count to the sink node. As a result, the algorithm will deposit the amount of pheromone $\Delta\omega_j$ on the path. If the value is less than zero, the algorithm will not deposit pheromone on this path.

In Eq. (4), R_j is the set of source nodes through node *j*, and $\sum(H_{ij})^{-1}$ is the inverse of the total hop counts of these source nodes reaching to node *j*. Thus, $\Delta\omega_j$ is the total hop count of the source nodes reaching to the sink node through node *j*. The lower the total hop count is, the greater is the amount of pheromone added on the path from node *i* to node *j*, as shown in Eq. (3). This means that more ants will be encouraged to follow this path.

For an aggregation node, it will update the pheromone levels of all its neighbors using Eq. (2) when an ant moves to it. If a node is not visited by ants within a threshold time, its pheromone will be evaporated according to:

$$\tau_{ij} = (1-\rho)\tau_{ij}. \tag{5}$$

After a short transitory period, the amount of pheromone of an aggregation node will be large enough to attract more ants carrying data packets from different sources to aggregate the data.

We will illustrate with an example shown in Fig. 3. Assume nodes A and B are source nodes and node D is a sink node. The sink node will flood its id to all the nodes in the network. All the nodes will compute the hop count to the sink node as h.

If a data packet on node A wants to transmit, then it will follow the random-proportional rule, Eq. (1), to select the next hop. In Fig. 4, node A sends a *Select* packet (C, A, A, 0, 1, 1) to node C. In Fig. 5, as the node C receives the *Select* packet, it will build a reaction table wherein the source id (S) is A, the previous hop node id (P) is A, the hop count from the node C to the source node A is 1 ($D_{js} = D_{js} + 1$), the extended hop count (EHC) is 1, and the time to live (TTL) is 1. Then node C broadcasts the *Select* packet (*, A, C, 1, 1, 0) to its one-hop neighbors. Node A (the current parent of node C) will ignore this packet. Afterwards, node C will check whether the EHC is equal to TTL or not. If EHC is equal to TTL, then node C will send the *Pheromone_Update* packet (C, A, h_c, $\Delta\omega_c$) to node A.

In Fig. 6, the algorithm builds the extended routing paths for sources A and B, and there are some nodes such as F, G, and L (aggregation nodes) which are overlapped by the two extended routing paths. For the aggregation nodes, they will send the *Pheromone_Update* packets to their own neighbors to reinforce their relationships. In Fig. 7, node G broadcasts the *Pheromone_Update* packet (G, *, 2, $\Delta\omega_G$) to its neighbors E, F, J, L, K,

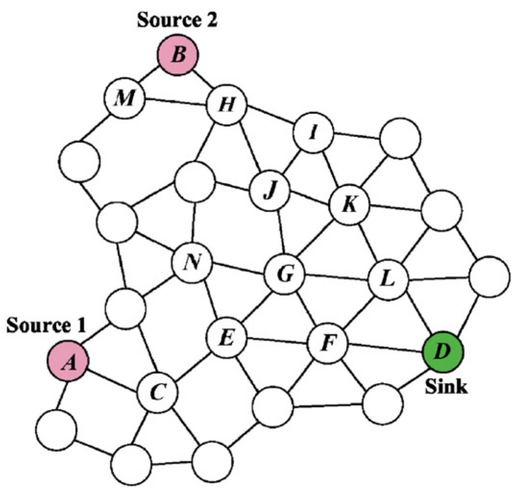

Fig. 3. There are two source nodes, A and B, and one sink node D in a wireless sensor network.

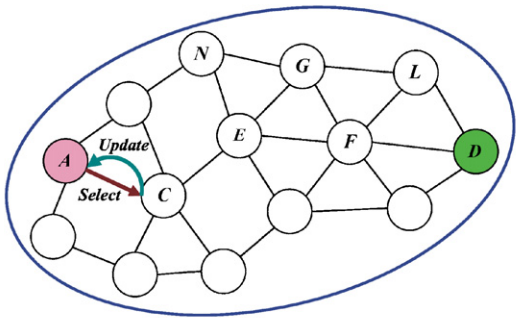

Fig. 4. Node A selects node C as the next hop, and node C will send the *Pheromone_Update* packet to node A.

a

Field	*S*	*P*	D_{js}	*EHC*	*TTL*
Value	*A*	*A*	1	1	1

b

Field	*S*	*P*	D_{js}	*EHC*	*TTL*
Value	*A*	*C*	2	1	0

Fig. 5. (a) The reaction table of node C. (b) The reaction table of the neighbors of node C.

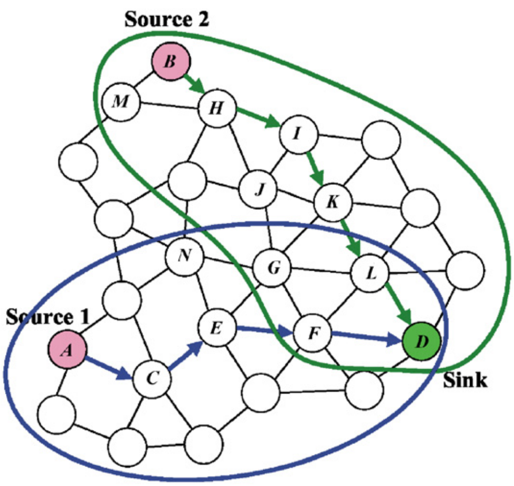

Fig. 6. The nodes F, G, L, and D are overlapped by the extended routing paths.

and N. The values of h_E h_F, h_G, h_J, h_L, h_K and h_N are 2, 1, 2, 3, 1, 2, and 3, respectively. The nodes E, F, J, L, K, and N will update their pheromone using Eq. (3). The increasing amount of pheromones of nodes J and G are $\Delta\tau_{JG} = [1 + (3-2)] \times \Delta\omega_G$.

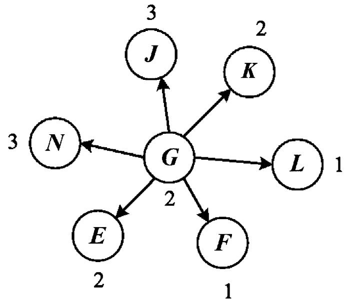

Fig. 7. The node G broadcasts the pheromone updating packets to its neighbors.

4. Simulation results

We first compared the performance of our proposed ant colony algorithm with the DD method. For these simulation experiments, we assumed that there are sensor nodes distributed randomly in a $100\,\text{m} \times 100\,\text{m}$ square region. All nodes have the same transmission range. There is a single sink node located at coordinates (10, 10) of the wireless sensor networks, which receives the data of all source nodes for all the simulations. We use an energy model (Heinzelman et al., 2000) to estimate the power consumption. We assume a simple model where the radio dissipates $E_{\text{elec}} = 50\,\text{nJ/bit}$ to run the transmitter or receiver circuitry and $\varepsilon_{\text{amp}} = 0.1\,\text{nJ/bit/m}^2$ for the transmit amplifier to achieve an acceptable $E_{\text{b}}/N_{\text{o}}$. Thus, to transmit a k-bit message a distance d using this radio model, the radio expends:

$$E_{Tx}(k, d) = E_{\text{elec}}k + \varepsilon_{\text{amp}}kd^2 \tag{6}$$

while to receive this message, the radio expends:

$$E_{Rx}(k) = E_{\text{elec}}k. \tag{7}$$

The simulation parameters are given in Table 1. The simulation results presented here are the averages of 30 simulation runs. The first set of experiments is carried out to investigate the total energy consumption with $N = 300$ and $R = 10$, 12, and 14. Fig. 8(a–c) shows the results of the DD method and our proposed ant colony algorithms Ant-0, Ant-1, and Ant-2 with different extended paths of 0, 1, and 2, respectively. We also vary the number of source nodes and observe the behavior. Since the DD method does not apply data aggregation, it has the highest energy consumption. The Ant-0 method generates fewer aggregation nodes than Ant-1 and Ant-2 and thus, requires also more energy. The Ant-1 and Ant-2 methods consume less energy than the other two methods because they can find more aggregation nodes to deliver the data.

The second set of experiments was carried out to measure the average energy consumption for each source node using the Ant-1 method with $R = 12$ and $N = 300$, 400, and 500. Fig. 9(a–c) shows the results by varying the number of source nodes in different runs. In the first run, the data aggregation does not happen and thus, the average energy consumption for one source node is almost equal for the different number of source nodes.

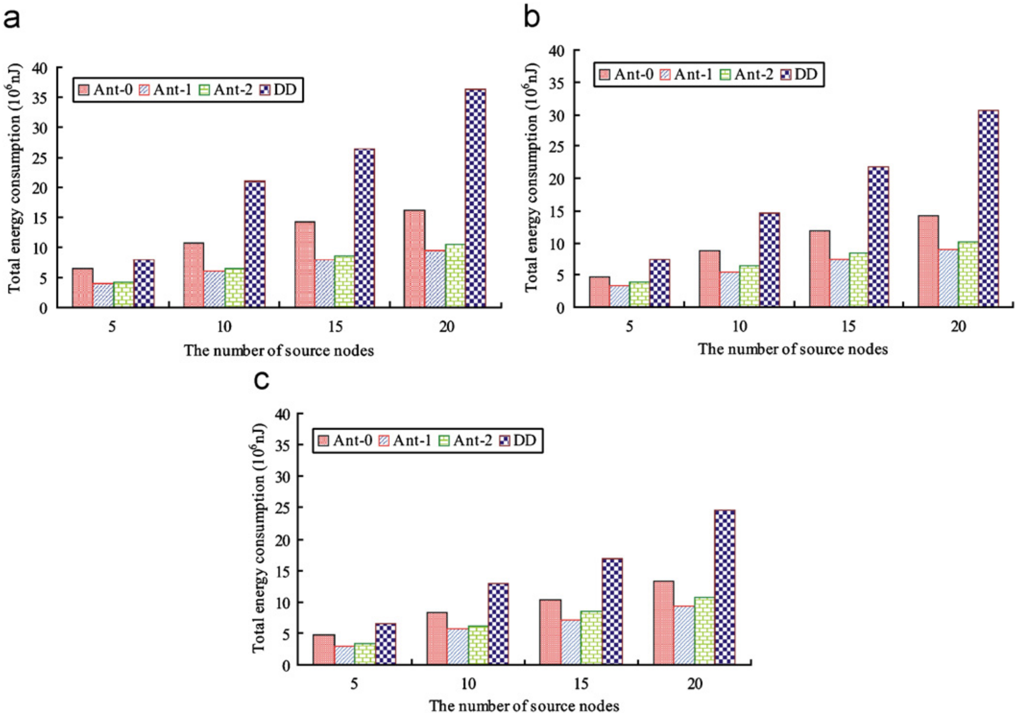

Fig. 8. Total energy consumption vs. the number of source nodes for methods Ant-0, Ant-1, Ant-2, and DD: (a) $R = 10$, (b) $R = 12$, (c) $R = 14$.

Table 1
Simulation parameters

Symbol	Definition	Values
N	Number of sensor nodes	300–500
S	Number of source nodes	5–20
R	Transmission range	10–14 m
E_{elec}	Radio dissipation	50 nJ/bit
E_{Tx}	Transmitter electronics	50 nJ/bit
E_{Rx}	Receiver electronics	50 nJ/bit
ε_{amp}	Transmit amplifier	0.1 nJ/bit/m^2
CP_{size}	Size of control packet	1 byte
DP_{size}	Size of data packet	64 bytes
S_{energy}	Initial energy of sensor nodes	0.25 J
ρ	Pheromone evaporation	0.3
β	Relative influence of heuristic values $\eta(i, j)$	20

In the 10th, 20th, and 30th runs, the higher the number of source nodes is, the higher the number of aggregation nodes and hence, the less is the average energy consumption.

The third set of experiments was carried out to measure the maximum run (a node finishes its lifetime) by comparing the DD method and our proposed ant colony algorithms with $R = 12$. Fig. 10 shows the results by varying the number of source nodes with the use

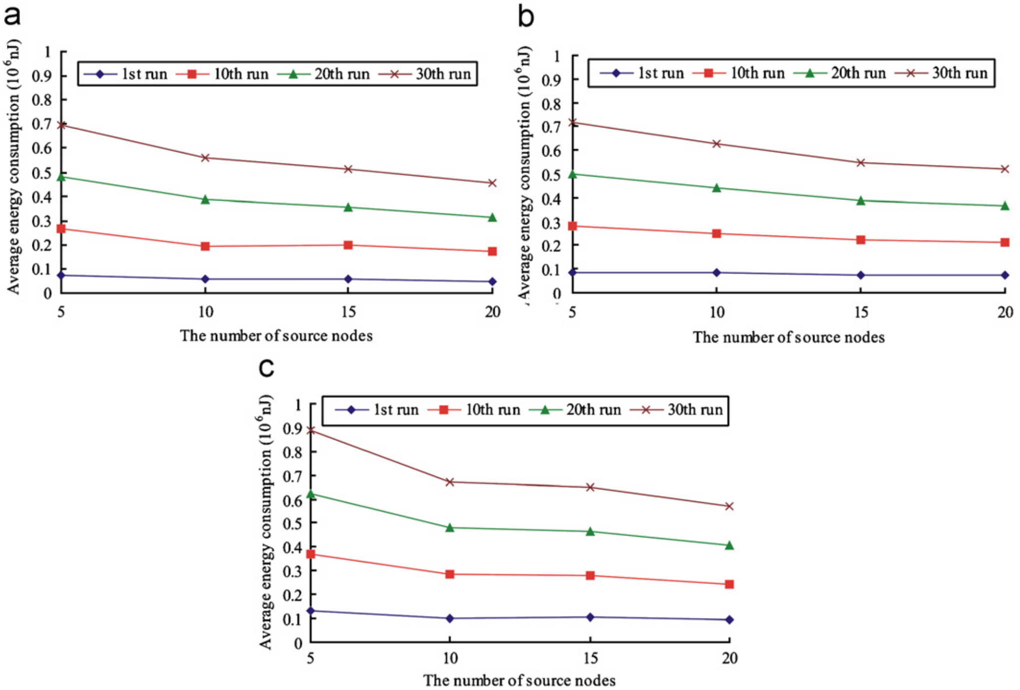

Fig. 9. Average energy consumption for each source node by varying the number of source nodes in the different runs: (a) $N = 300$, (b) $N = 400$, (c) $N = 500$.

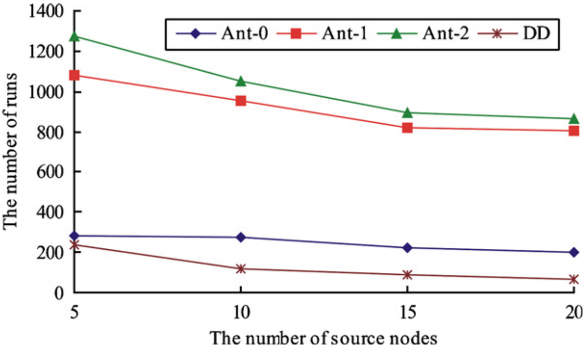

Fig. 10. Runs vs. the number of source nodes for methods Ant-0, Ant-1, Ant-2, and DD.

of different methods at $N = 300$. It was found that the maximum runs of Ant-1 and Ant-2 are greater than Ant-0 and DD. The reason is that Ant-1 and Ant-2 can find more aggregation nodes to share the same paths. Moreover, the neighboring nodes of the sink node will receive the aggregated data only and thus, will consume less energy.

The fourth experiment is to measure the total overhead and the total transmitted amount of data with $R = 12$ for 30 simulation runs. Besides the two previous methods, the GIT method was also considered for comparison. Figs. 11 and 12 show the results by varying the number of source nodes with $N = 400$. In Fig. 11, the GIT method broadcasts

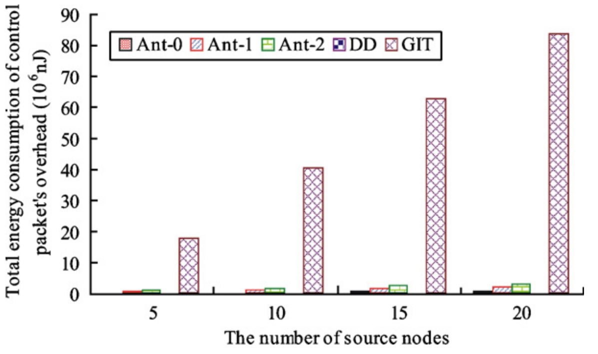

Fig. 11. Total energy consumption of the overhead for the different number of source nodes.

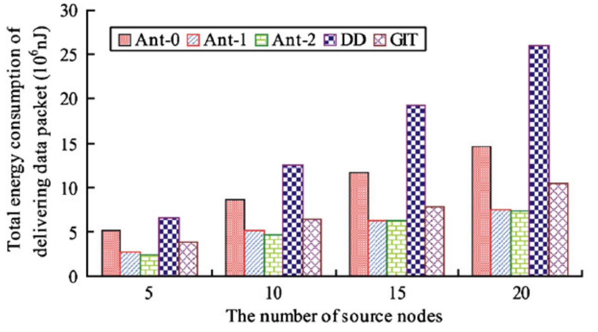

Fig. 12. Total energy consumption for data transmission of the different number of source nodes.

a lot of control packets to find the shortest path to connect with the established tree for each source at the first run. Although each source broadcasts only once, the total energy consumption is very high. In Fig. 12, the GIT method uses a greedy method to find the shortest path with the established tree. Thus, it consumes less energy for data transmission than the Ant-0, but still higher than that of Ant-1 and Ant-2.

Fig. 13 presents the final result of our proposed ant colony algorithm after 30 runs. It depicts the aggregation trees topology formation with $N = 400$, $S = 10$, and $R = 12$. When the source nodes send out data packets during the first run, the pheromone has not yet accumulated on any node. Hence, no data packet can be aggregated, and this is why the data packet is not aggregated in Fig. 13(a). On the 10th run, the network has source nodes to explore unknown paths in Fig. 13(b). On the 20th run, the source nodes can find relatively suitable paths in Fig. 13(c). After thirty runs, the aggregation tree is formed and shown in Fig. 13(d).

5. Conclusion

In this paper, we presented an ant colony algorithm for data aggregation in wireless sensor networks. Every ant will explore all possible paths from the source node to the sink node. In order to increase the probability of intersection of routing paths, our mechanism extends the routing paths. The data aggregation tree is constructed by the accumulation of

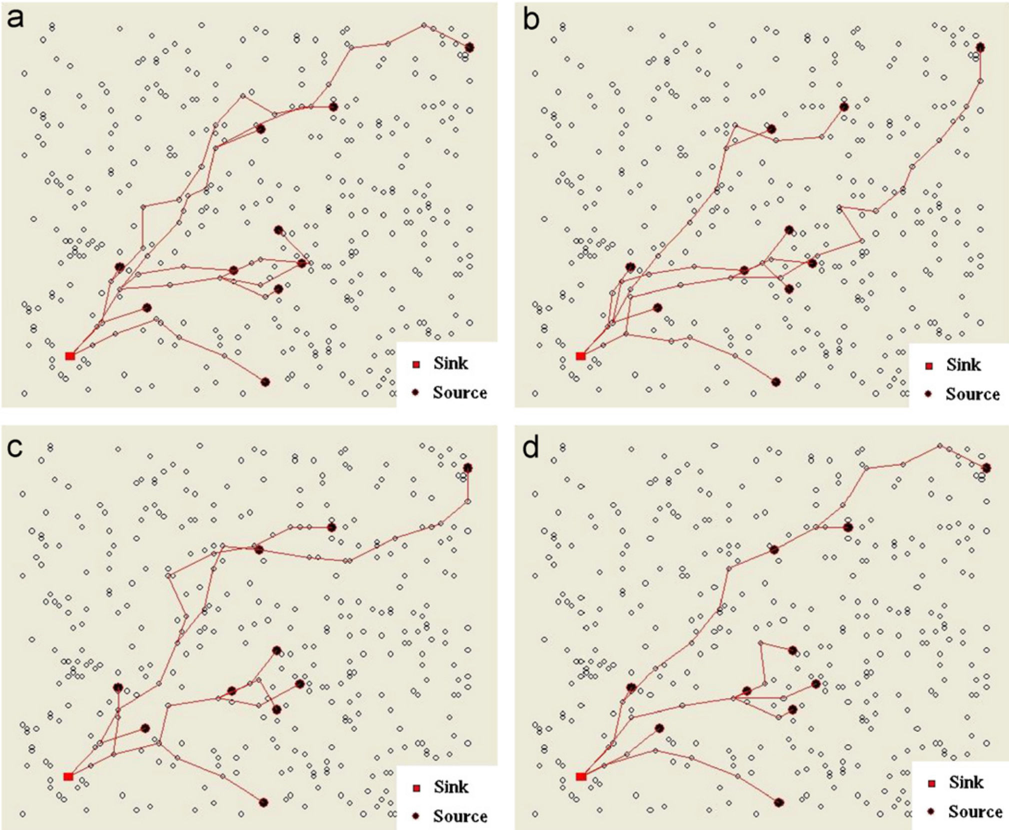

Fig. 13. The aggregation trees formation: (a) the first run, (b) the 10th run, (c) the 20th run, (d) the 30th run.

pheromone. After a short transitory period, the amount of pheromone on the aggregation nodes is sufficiently large to guide ants (the data packets from different sources) to meet together at these nodes for data aggregation. Simulation results show that our proposed algorithm actually consumes less data delivery energy than the other traditional methods such as the DD and GIT methods.

References

Akyildiz F, Su W, Sankarasubramaniam Y, Cayirici E. A survey on sensor network. IEEE Commun. Mag. 2002;40(8):102–14.

Al-Karaki JN, Ul-Mustafa R, Kamal AE. Data aggregation in wireless sensor networks—exact and approximate algorithms. In: International Workshop on High-Performance Switching and Routing, 2004.

Arampatzis T, Lygeros J, Manesis S. A survey of applications of wireless sensors and wireless sensor networks. In: Mediterranean Conference on Control and Automation, 2005.

Bhattacharjee S, Das N. Distributed data gathering scheduling in multihop wireless sensor networks for improved lifetime. In: International Conference on Computing: Theory and Applications (ICCTA), 2007.

Chen G, Guo T-D, Yang W-G, Zhao T. An improved ant-based routing protocol in wireless sensor networks. In: International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom), 2006.

Culler D, Estrin D, Srivastava M. Overview of sensor networks. IEEE Comput. 2004;37(8):41–9.

Dorigo M, Gambardella LM. Ant colony system: a cooperative learning approach to the traveling salesman problem. IEEE Trans. Evol. Comput. 1997;1(1):53–66.

Ducatelle F, Caro GD, Gambardella LM. Using ant agents to combine reactive and proactive strategies for routing in mobile ad hoc networks. Int. J. Comput. Intell. Appl. 2005;5(2):169–84.

Heinzelman WR, Chandrakasan A, Balakrishnan H. Energy-efficient communication protocol for wireless microsensor networks. In: Hawaii International Conference on System Sciences, 2000.

Intanagonwiwat C, Estrin D, Govindan R, Heidemann J. Impact of network density on data aggregation in wireless sensor networks. International Conference on Distributed Computing Systems (ICDCS), 2002.

Intanagonwiwat C, Govindan R, Estrin D, Heidemann J, Silva F. Directed diffusion for wireless sensor networking. IEEE/ACM Trans. Networking 2003;11(1):2–16.

Krishnamachari B, Estrin D, Wicker S. The impact of data aggregation in wireless sensor networks. In: International Conference on Distributed Computing Systems Workshops (ICDCSW), 2002.

Li D, Cao J, Liu M, Zheng Y. Construction of optimal data aggregation trees for wireless sensor networks. In: International Conference on Computer Communications and Networks (ICCCN), 2006.

Luo H, Liu Y, Das SK. Routing correlated data with fusion cost in wireless sensor networks. IEEE Trans. Mobile Comput. 2006;5(11):1620–32.

Misra R, Mandal C. Ant-aggregation: ant colony algorithm for optimal data aggregation in wireless sensor networks. In: International Conference on Wireless and Optical Communications Networks, 2006.

Motegi S, Yoshihara K, Horiuchi H. DAG based in-network aggregation for sensor network monitoring. In: International Symposium on Applications and the Internet, 2006.

Okdem S, Karaboga D. Routing in wireless sensor networks using ant colony optimization. In: First NASA/ESA Conference on Adaptive Hardware and Systems (AHS), 2006.