On Computing Multi-agent Itinerary Planning in Distributed Wireless Sensor Networks

Bo Liu[1(✉)], Jiuxin Cao[1], Jie Yin[1], Wei Yu[2], Benyuan Liu[3], and Xinwen Fu[3]

[1] School of Computer Science and Engineering, Southeast University, Nanjing, China
{bliu, jx.cao, jyin}@seu.edu.cn
[2] Department of Computer and Information Sciences, Towson University, Towson, USA
wyu@towson.edu
[3] Department of Computer Science, University of Massachusetts Lowell, Lowell, USA
{bliu, xinwenfu}@cs.uml.edu

Abstract. The agent technology has been widely used in wireless sensor networks to perform data fusion and energy balancing. Existing multi-agent itinerary algorithms are either time-consuming, or too complicated to realize in reality. In this paper, we design a routing itinerary planning scheme for the multi-agent itinerary problem by constructing the spanning tree of WSN nodes. First, we build a multi-agent based distributed WSN (DWSN) model. Second, we present a novel routing itinerary algorithm named DMAIP, which can group all the sensor nodes into multiple itineraries for agents. We also extend DMAIP and design DMAIP-E, which can avoid long distance transmission in DMAIP. Our evaluation results demonstrate that our algorithms are better in the aspect of life cycle and energy consumption than the existing DWSN data collecting schemes.

Keywords: Distributed wireless sensor network · Mobile agent · Itinerary planning · Energy efficiency

1 Introduction

Wireless Sensor Networks (WSNs) have been widely used for data collection and situation monitoring in various applications. The raw data collected by sensors can then be forwarded through a number of relay nodes to a remote base station, denoted as the sink node, [5, 6]. In a WSN, sensor nodes with limited energy are often randomly deployed in massive quantities, and each node may act both as a data collector and a traffic relay, as shown in existing research [1].

Agent-based technology has attracted growing attention to improve energy efficiency, scalability, and reliability of a WSN [2–4, 7, 21]. In an agent-based WSN, the main technical challenges include data fusion, energy efficiency, as well as energy balancing. For example, Lin et al. [8] leveraged the agent-based technology to balance energy consumption in data collection process of WSNs.

K. Xu and J. Zhu (Eds.): WASA 2015, LNCS 9204, pp. 366–376, 2015.
DOI: 10.1007/978-3-319-21837-3_36

A critical problem of an agent-based WSN is how to design the itinerary through the WSN for the mobile agent to collect data. Existing approaches for agent itinerary in WSNs can be classified into two categories: (i) single agent itinerary planning based on clustering [9–12], and (ii) multi-agent itinerary planning. For example, Xu et al. [9] investigated static, dynamic, and predictive dynamic schemes to solve the target tracking problem in a WSN. Nonetheless, their work assumes that there is only one target node in the field, and the itinerary is for only one agent. In reality, using a single agent is not practical because of large delay, unbalanced load, and insecurity with large accumulated size [14, 15]. So, multi-agent based WSN and the problem of multi-agent itinerary planning have attracted growing attention [13, 14]. Chen et al. [14] proposed a source-grouping scheme and an iterative algorithm for multi-agent based itinerary planning. They partition source nodes into several sets and use the least number of agents while achieving the required coverage of source nodes. Nonetheless, finding an optimal number of agents is a NP-hard problem and their algorithm is heuristic. Cai et al. [15] applied the genetic algorithm for multiple mobile agents traversing a WSN. Nonetheless, their proposed scheme is complicated and hard to be realized in reality.

Clustering is another important strategy to solve the data fusion and collection problem in WSNs and has been widely adopted. In the typical clustering algorithm called LEACH (Low Energy Adaptive Clustering Hierarchy) [16], the high-energy cluster-head positions are randomly rotated so that the energy consumption of sensors can be balanced. But LEACH assumes a homogeneous distribution of sensor nodes in the given area, which may not be realistic. We will compare our algorithm with clustering based algorithms [21–24].

Existing research efforts on multi-agent routing mainly focus on how to select source nodes to generate itineraries other than how to compute routes. Our work fills this gap. The rest of this paper is organized as follows: In Sect. 2, we present the problem. In Sect. 3, we present the agent routing approach considering energy balancing. In Sect. 4, we present performance evaluation results. Finally, we conclude the paper in Sect. 5.

2 Mobile-Agent Based Wireless Sensor Networks

2.1 Problem Definition

Given a WSN, we utilize mobile agent techniques to accomplish data collection and fusion with less energy consumption and longer life-cycle. A sink node (e.g., base station) is responsible for gathering data from sensor nodes. The objective of our research is to find multiple disjoint paths which cover all sensor nodes in the WSN. The disjoint paths will be assigned to mobile agents.

Our problem is defined as follows: Given an undirected connected graph $G = <V, E>$, where V is the set of nodes and E is the set of edges, the problem is to find k disjoint paths in G to cover all nodes in V, where every edge in these paths can be found in E. It is NP-hard to find optimal number of disjoint paths as we mentioned in Sect. 1. We focus on developing efficient heuristic and polynomial time algorithms to make agents work effectively in a WSN and produce near-optimal itineraries.

2.2 Multi-agent Based WSN

We propose a multi-agent based data collection scheme. As shown in Fig. 1, there are three parts: *remote user, sink node, sensor node* in our scheme. Remote user assigns tasks to sink node. All sensor nodes have operating environment installed for mobile agent. When sink node receives a task from remote user, sink node traversals network topology to generate a spanning tree, then assigns every path to a mobile agent. When a mobile agent moves along its own path to a sensor node, it will do data processing and carry the raw data to the next node. All agents will perform tasks in parallel until they visit all nodes of their paths. Sink node will manipulate data from every path and send back the final results to remote user. Figure 2 shows the detailed workflow.

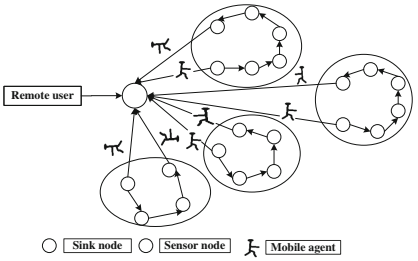

Fig. 1. Multi mobile agent data collection model

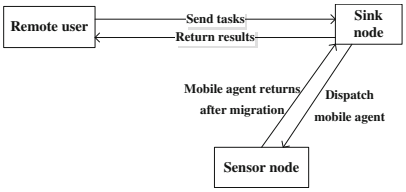

Fig. 2. Task flow model

3 Novel Algorithm for Multi-agent Itinerary

In this section, we present a novel DFS based Multi-Agent Itinerary Planning (DMAIP) algorithm in WSNs. Particularly, we first present the DMAIP algorithm that decreases the communication distance between sensor nodes in the agent's migration path. We then present the enhanced DMAIP algorithm, denoted as, DMAIP-E. DMAIP-E changes the process of traversing the network, considering distance between sink node and sensors when generating disjoint paths.

3.1 DMAIP Algorithm

We firstly build a network topology graph from WSN, then generate a spanning tree of the connected graph in the order of increasing weight and finally traverse the spanning tree recursively to find disjoint paths of all the subtrees.

Algorithm 1 introduces how we build the network topology G. We assume that the wireless sensor nodes can locate themselves. They will send position coordinates to the sink node. Accordingly, the sink node will collect position data and compute distances between all the pairs of nodes. If the distance is not larger than the radius, we leave it as what it is. Otherwise, we will mark it inaccessible explicitly by value −1.

Algorithm 1: Build a network topology graph G.

Input: Array c[n][2] and R //c includes n nodes' two-dimension coordinates. R is the communication radius.

Output: Weighted adjacent matrix $M = [d_{u,v}]_{n*n}$ //$d_{u,v}$ is the distance between node u and v.

Algorithm description: Calculate distance between all the pairs of nodes. Assume u and v are two nodes, then their distance is:

$$d_{u,v} = \sqrt{(c[u][0] - c[v][0])^2 + (c[u][1] - c[v][1])^2} \quad (1)$$

If $d_{u,v} \leq R$ *or* $u = sink$ *or* $v = sink$, *then leave* $d_{u,v}$ *as what it is, else* $d_{u,v} = -1$. *If* $u = v$, $d_{u,v} = -1$. *It is obvious that* $d_{u,v} = d_{v,u}$. *At last, we get matrix* $M = [d_{u,v}]_{n*n}$.

Algorithm 2 generates a spanning tree of the connected graph G. Basically, we traverse weighted adjacent matrix M (the graph) by DFS and generate a spanning tree.

Algorithm 2: Generate a spanning tree of connected graph.

Input: Weighted adjacent matrix $M = [d_{u,v}]_{n*n}$

Output: A connected spanning tree T

Algorithm description: Define an array storing visited states of nodes as S[i]($1 \leq i \leq n$). Initialize all values to zero except that $S[sink] = 1$.

Choose u_0 from all sensor nodes where $d_{u_0,sink}$ is minimum, as the initial start node or source node.

Visit node u_0 first. Regard $u_0$as u, set its visited state $S[u] = 1$, then find next node v where $d_{u,v}$ is minimum, $d_{u,v} \neq -1$ and $S[v] = 0$. Create an undirected edge from u to v, then set $S[u_1] = 1$ and regard node *v* as the next node *u*.

Repeat the process until we cannot find the next node *v*. At the time, a path is formed from node v to the current node *u*. Then we go back along the path to find a node from which we can find the nearest node *v* and we continue to do so using DFS. The visited nodes form a spanning tree. After all nodes are visited, a spanning tree is generated.

In Algorithm 3, we traverse the spanning tree recursively to find disjoint paths of all the subtrees.

Algorithm 3: Generate disjoint paths

Input: A connected spanning tree T

Output: Several disjoint paths $P_1 \ldots P_k$.

Algorithm description: We will achieve this algorithm by calling function GeneratePath defined as follows with its pseudo code.

```
GeneratePath(TreeNode &rootNode)
Begin
   TreeNode subtreeRootNodes[];
   find the path from rootNode to the left-most node;
   store the path as one disjoint path;
   delete the left-most child of all nodes in the path;
   store all nodes in the path that still have child node(s) in subtreeRootNodes [];
   For every subtreeRootNodes [i]
         GeneratePath(subtreeRootNodes [i]);
   End for;
End
```

3.2 DMAIP-E Algorithm

If the sensor node is far away from the sink node, energy consumption increases and this will reduce the life time of the WSN. Based on DMAIP, we propose an algorithm called DMAIP-E that considers the distance between the sink node and all sensor nodes, as shown in Algorithm 4. We first build network topology graph according to Algorithm 1 of DMAIP to get the distance matrix $M = [d_{u,v}]_{n*n}$. Secondly, we generate multiple paths for agents.

In Algorithm 4, we start from the nearest node A to the sink node, find the next node B that meets these conditions

$$d_{A,sink} < d_{B,sink} \text{ and } d_{A,B} < d_{B,sink} \tag{2}$$

We illustrate the process of choosing appropriate node B in Fig. 3. Node B must be satisfied with Formula (2). So nodes 1, 2 and 3 are being considered. Besides, we require the nearest node to node A. Therefore, node 2 is the node B. Assume node B as the next A and find the next node B. Once there is no more node B to be found, it will form a path from the first node A to the last node B. Then go back to the sink node and repeat the process above. It will stop when all nodes are included in all generated paths.

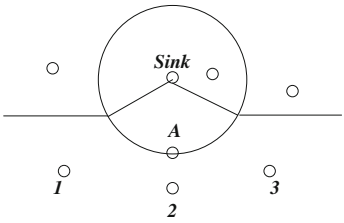

Fig. 3. A process of choosing appropriate node B

Algorithm 4: Generate multiple disjoint paths when traversing.

Input: Weighted adjacent matrix $M = [d_{u,v}]_{n*n}$
Output: A group of disjoint paths $P_1 \ldots P_k$
Algorithm description: Define an array storing visited states of nodes as $S[i](1 \leq i \leq n)$. Initialize all values to 0 except that $S[sink] = 1$.
step 1) Within M, find the nearest node A_0 to the sink node.
step 2) Regard node A_0 as node A, let $S[A] = 1$, and find appropriate node B.
step 3) Regard node B as node A, $S[B] = 1$ let find the next appropriate node B.
step 4) Repeat step 3) until we cannot find the next B. Store the path from the first A to the last B as a new path.
step 5) If there is at least one node u that $S[u] = 1$, go to step 1) else end.

The sink node will dispatch an agent to the farthest node of every disjoint path generated above. When an agent finished its task, it will return from the other end of its path to the sink node, and the sink node will manipulate all data from all paths. In fact, we let the sink node dispatch agents to the outermost location of the path to decrease the distance that the agent loaded with data migrates.

The time complexity of DMAIP algorithm is $O(n^3)$. The time complexity of DMAIP-E algorithm is $O(n^2)$.

4 Performance Evaluation

We implemented the DMAIP and DMAIP-E schemes in the Qualnet environment [20], and compared them with LEACH based algorithms [16, 22, 23]. At last, we present the experimental results of comparing DMAIP, DMAIP-E and LEACH.

4.1 Methodology

We deployed sensor nodes randomly in an area of 200 m × 200 m. Each node was deployed at a fixed position. Sink node is located at (100, 100). Table 1 gives the parameter settings.

We take the following assumptions: (i) wireless sensor network is of large-scale and nodes are distributed randomly, (ii) the initial energy batteries of all the sensor nodes are the same, (iii) The data packet is of the same size, and (iv) all sensor nodes stay in the fixed position.

Table 1. Simulation experimental parameter settings

Parameter Name	Value
Number of nodes	50, 75, 100, 125, 150
Size of the area/m^2	200*200
Location of the base station/m	(100, 100)
Initial energy E_{init}/J	0.5
Energy consumption of transmitting amplifier E_{fs}/(pJ*bit^{-1}*m^{-2})	10
Energy consumption of transmitting amplifier E_{amp}/(pJ*bit^{-1}*m^{-4})	0.0013
Energy consumption of transmitting and receiving circuit E_{elec}/nJ* bit^{-1}	50
Initial size of mobile agent/byte	1024
Length of data packet/byte	2048
IData fusion rate p	0.9
Communication radius/m	40

4.2 Results

We evaluate the performance of our proposed schemes in terms of energy efficiency and life cycle. In the following, we first compare three related algorithms and then show their performance results.

- **Results Generated by three Algorithms**

All LEACH based algorithms use a typically clustering mechanism for data collection, while our proposed DMAIP and DMAIP-E use multi-path methods. The WSN has 100

sensor nodes. In our experiments, the three algorithms run in the same environment. Initially, all sensor nodes are deployed as shown in Figs. 4, 5 and 6.

Itineraries of LEACH Algorithm: In LEACH algorithm, we choose 13 sensor nodes as cluster heads from 100 ones. Ordinary nodes can only communicate with cluster heads, and only cluster heads can communicate with sink nodes. As shown in Fig. 4, according to LEACH strategy, ordinary nodes will choose the nearest cluster head as theirs because of the randomness of deployed sensor nodes and cluster head nodes. Therefore, each cluster head node manages unbalanced number of ordinary nodes within its domain. A cluster head may communicates with many cluster members, and may have long communication distance with the sink node. This phenomenon leads to fast death of some cluster head due to the fast energy consumption. With the increasing number of the sensor nodes in large-scale WSNs, the distance between a cluster head, its cluster member nodes and sink node will be larger and consume much more energy.

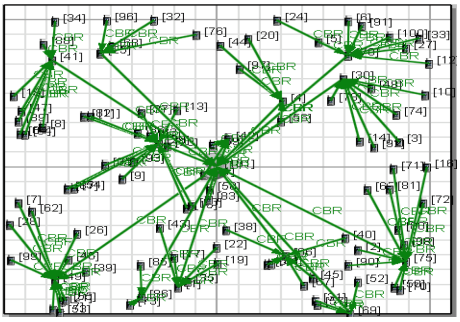

Fig. 4. The branches generated by LEACH

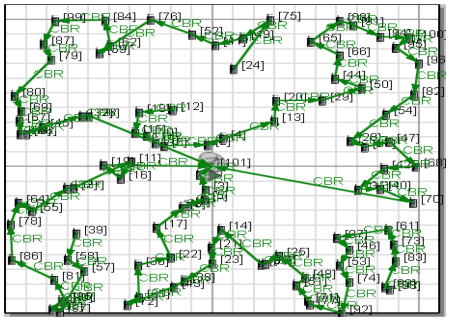

Fig. 5. The branches generated by DMAIP

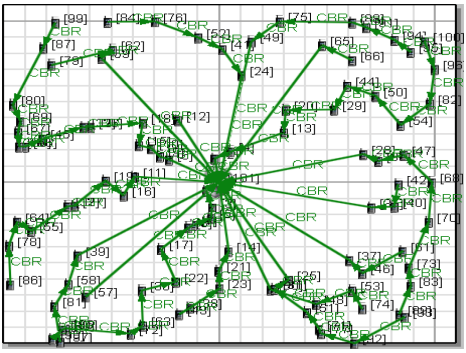

Fig. 6. The branches generated by DMAIP-E

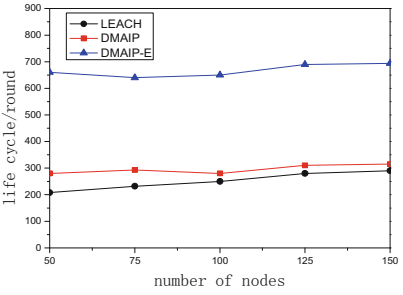

Fig. 7. Life cycle with different number of nodes

Itineraries of DMAIP and DMAIP-E Algorithms: Figure 5 illustrates the results for DMAIP. It has 4 paths and the last node along each path consumes the most energy. For example, in Fig. 5, the 70th node, the last node of a path, is far from the sink node and dies fast because of the fast energy consumption. Figure 6 shows the results for DMAIP-E algorithm. Recall that DMAIP-E is the enhanced version of the DMAIP algorithm and considers not only the distance between nodes, but also the distance between the last node of each path and sink node. The sink node dispatches mobile agents to the farther end of each path, and the agent returns to the other end of the path. This reduces the amount of energy consumption to a certain degree and extends the life cycle of the wireless sensor network.

- **Performance Results**

The following three performance metrics are considered in our simulation: life cycle, energy consumption, impact of the number of nodes. Table 2 gives the evaluation metrics.

Table 2. Performance metrics

Evaluation metrics	Descriptions
Round	The whole process from when the sink node dispatch agents for every path for data collection to when all agents return to the sink node
Energy consumption of network nodes	The total energy consumption by sensor nodes when sink node completes one round of data collection
Number of residual nodes	The number of sensor nodes alive in the network
Life cycle	The round number from the beginning of the network to the death of the first node

Life cycle: From Fig. 7, we can see that with different number of nodes, the life cycle of DMAIP -E is much longer than that of DMAIP, and the life time of DMAIP is a little longer than that of LEACH algorithm as the communication distance of DMAIP-E is the smallest, while the distance of LEACH is mostly very large.

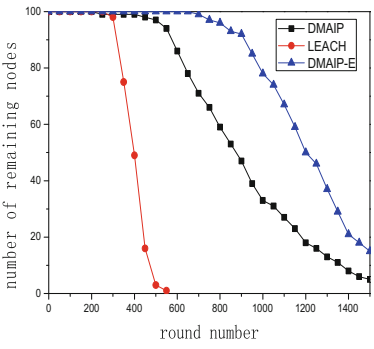

Fig. 8. Changes of residual nodes' number with rounds

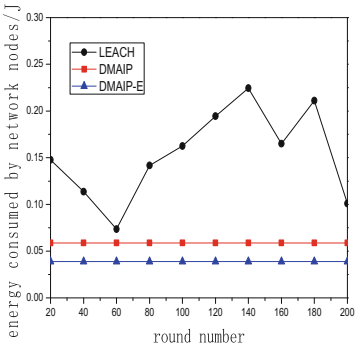

Fig. 9. Changes of total energy of nodes with rounds

The Number of Residual Nodes: Figure 8 illustrates results that present the number of remaining nodes versus the number of rounds. As we can see, as the number of rounds increases, some sensor nodes will die. The number of rounds for LEACH, DMAIP and DMAIP-E to have dead nodes are 250, 280 and 650, respectively. DMAIP-E has the largest number of rounds, i.e., the longest life cycle, more than twice rounds of the other two algorithms. The reason why nodes in DMAIP die very early is that it is very far from the end node of some paths to the sink node and therefore transmitting data from the end node to the sink node will consume energy heavily. In addition, all nodes for LEACH die in about 600th round, while for DMAIP and DMAIP-E, nodes will not be all dead until about 1500th round. This is because DMAIP and DMAIP-E adopt a mobile agent model that saves and balances the energy of sensor nodes. Since DMAIP-E outperforms the others.

Energy consumption: Energy consumption is another important performance metric for WSNs [17–19]. This experiment was performed in a network with 100 nodes and the results are shown in Fig. 9. As we can see, the energy consumption of DMAIP and DMAIP-E algorithm is almost the same while the energy consumption of LEACH is not stable. This is because DMAIP and DMAIP-E collect data according to preset paths while LEACH chooses cluster head randomly and changes it every few rounds. Therefore, energy consumption of LEACH varies with rounds. After every round, DMAIP-E algorithm consumes the least energy while LEACH uses the most.

5 Conclusion

The itinerary planning problem is a key issue in distributed WSN. In this paper, we present a multi-agent based paradigm for data collecting in WSN. Since the optimal multi-agent itinerary generation problem is NP-hard, we propose an approximate algorithm called DMAIP-E based on the global topology graph. Our simulation results show that DMAIP-E is appropriate for large-scale WSN and has better performance than the existing clustering based algorithms in terms of life cycle and energy consumption.

Acknowledgment. This work is supported by National Key Basic Research Program of China under Grants No. 2010CB328104, National Natural Science Foundation of China under Grants, No. 61272531, No. 61370208, No. 61320106007, No. 61472081, No. 61402104, China National Key Technology R&D Program under Grants No. 2010BAI88B03 and No. 2011BAK21B02, Jiangsu Provincial Natural Science Foundation of China under Grants No. BK20130634 and No. BK20140648, Jiangsu Provincial Key Laboratory of Network and Information Security under Grants No. BM2003201, and Key Laboratory of Computer Network and Information Integration of Ministry of Education of China under Grants No. 93K-9.

References

1. Wang, F., Wang, D., Liu, J.C.: Traffic-aware relay node deployment for data collection in wireless sensor networks. In: Proceedings of the 6th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad-HoC Communications and Networks, Rome, Italy, pp. 351–359 (2009)
2. Qi, H., Wang, X., Iyengar, S.S., Chakrabarty, K.: Multisensor data fusion in distributed sensor networks using mobile agents. In: Proceedings of the International Conference on Information Fusion, pp. 11–16 (2001)
3. Du, W., Deng, J., Han, Y.S., Varshney, P.K.: A pairwise key pre-distribution scheme for wireless sensor networks. In: Proceedings of the 10th ACM Conference on Computer and Communications Security (CCS 2003), pp. 42–51 (2003)
4. Eschenauer, L., Gligor, V.D.: A key-management scheme for distributed sensor networks. In: Proceedings of the Ninth ACM Conference on Computer and Communications Security (CCS 2002), pp. 41–47 (2002)
5. Wang, F., Wang, D., Liu, J.C.: Traffic-aware relay node deployment for data collection in wireless sensor networks. In: Proceedings of the 6th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad-HoC Communications and Networks, Rome, Italy, pp. 351–359 (2009)
6. Luo, H., Liu, Y., Das, S.K.: Distributed algorithm for en route aggregation decision in wireless sensor networks. IEEE Trans. Mob. Comput. **8**(1), 1–13 (2009)
7. Qi, H., Xu, Y., Wang, X.: Mobile-agent-based collaborative signal and information processing in sensor networks. Proc. IEEE **91**, 1172–1183 (2003)
8. Lin, K., Chen, M., Zeadally, S., Rodrigues, J.J.P.C.: Balancing energy consumption with mobile agents in wireless sensor networks. Future Gener. Comput. Syst. **28**(2), 446–456 (2012)
9. Xu, Y., Qi, H.: Mobile agent migration modeling and design for target tracking in wireless sensor networks. Ad Hoc Netw. **6**(1), 1–16 (2008)
10. Chen, M., Yang, L.T., Kwon, T., Zhou, L.: Itinerary planning for energy-efficient agent communications in wireless sensor networks. IEEE Trans. Veh. Technol. **60**(7), 3290–3299 (2011)
11. Chen, B., Liu, W.: Mobile agent computing paradigm for building a flexible structural health monitoring sensor network. Comput. Aided Civ. Infrastruct. Eng. **25**, 504–516 (2010)
12. Wu, Q., Rao, N.S.V., Barhen, J., et al.: On computing mobile agent routes for data fusion in distributed sensor networks. IEEE Trans. Knowl. Data Eng. **16**(6), 740–753 (2004)
13. Wang, X., Chen, M., Kwon, T., Chao, H.C.: Multiple mobile agents' itinerary planning in wireless sensor networks: survey and evaluation. IET Commun. **5**(12), 1769–1776 (2011). (Special issue on distributed intelligence and data fusion for sensor systems)
14. Chen, M., Gonzalez, S., Zhang, Y., Leung, V.C.M.: Multi-agent itinerary planning for wireless sensor networks. In: Bartolini, N., Nikoletseas, S., Sinha, P., Cardellini, V., Mahanti, A. (eds.) QShine 2009. LNICST, vol. 22, pp. 584–597. Springer, Heidelberg (2009)
15. Cai, W., Chen, M., Hara, T., Shu, L.: GA-MIP: genetic algorithm based multiple mobile agents itinerary planning in wireless sensor network. In: Proceeding of the 5th Annual International Wireless Internet Conference (WICON), Singapore, Mar 2010
16. Heinzelman, W.R., Chandrakasan, A., Balakrishnan, H.: Energy-efficient communication protocol for wireless microsensor networks. In: IEEE Proceedings of the 33rd Annual Hawaii International Conference on System Sciences (2000)

17. Xiao, Y.K., Shan, X.M., Ren, Y.: Game theory models for IEEE802.11DCF in wireless ad hoc networks. IEEE Radio Commun. **43**, S22–S26 (2005)
18. Tseng, C.-C., Liang, Y.J.: Organizing power efficient cluster-based network architectures for wireless ad hoc networks. In: IEEE Cat No. 07CH37784, pp. 114–118 (2007)
19. Zhi, L., Zhong, C., Wolisz, A.: An integrated data-link energy model for wireless sensor networks. In: IEEE Communication Society, pp. 3777–3783 (2004)
20. Simulator Q N. Scalable Network Technologies Inc (2011). www.qualnet.com
21. Amiri, E., Keshavarz, H., Fahleyani, A.S., Moradzadeh, H., Komaki, S.: New algorithm for leader election in distributed WSN with software agents. In: IEEE International Conference on Space Science and Communication (IconSpace) (2013)
22. Malhotra, R., Aggarwal, R., Monga, S.: Analyzing core migration protocol wireless ad hoc networks by Adopting Multiple Nodes Diverse Test-bed. Int. J. Comput. Appl. 9(3) 35–41 (2010)
23. Nugraheni, C.E.: Formal verification of ring-based leader election protocol using predicate diagrams. IJCSNS Int. J. Comput. Sci. Netw. Secur. **9**(8), 1 (2009)
24. Galstyan, A., Krishnamachari, B., Lerman, K.: Resource allocation and emergent coordination in wireless sensor networks. In: American Association for Artificial Intelligence (2004)