Mobile-Agent-Based Distributed Decision Tree Classification in Wireless Sensor Networks

SHUANG-QUAN WANG, XIN CHEN*, NING-JIANG CHEN* AND JIE YANG
Institute of Image Processing and Pattern Recognition
Shanghai Jiaotong University
Shanghai, 200240 P.R.C.
Philips Research East Asia
Shanghai, 200070 P.R.C.

Wireless sensor networks (WSNs) connect physical sensors that are distributed in the environment. In many applications, the statistical pattern recognition methods, such as decision tree (DT) algorithms, are used to recognize the patterns of the sensor readings. To enable the decision tree classification (DTC) in WSNs, a new distributed decision tree classification (DDTC) algorithm based on mobile agent is proposed in this paper. We organize the conjunctive sets of linear classifiers in DT into groups of operations on attributes. Each group of operations is allocated to a single sensor node. If a mobile agent visits these sensor nodes serially, the recognition result can be acquired step by step. Thus the sensor nodes do not need to transmit all the sensor data to a centralized node where all the data processing is carried out in traditional DTC. In DDTC, if not all the attributes are needed for operation, classification of one instance can finish halfway. Two public data sets are used to evaluate the performance of DTC and DDTC on energy consumption. The simulation results indicate that, compared with the centralized DTC, the DDTC algorithm decreases the number of transmissions, balances the power consumption and computation among sensor nodes, and prolongs the lifetime of the network.

Keywords: mobile agent, decision tree, distributed classification, wireless sensor networks, energy consumption

1. INTRODUCTION

A wireless sensor network (WSN) consists of sensor nodes with the capabilities of sensing, memorizing, processing and communicating in a power-efficient way. Each sensor node connects the physical world with the digital world by expressing the physical phenomena in sensor readings. It also processes and communicates the sensor readings to other sensor nodes to extract meaningful context information [1].

In many applications, the statistical pattern recognition methods, such as decision tree (DT) algorithms, are used to analyze the patterns of the sensor data [2]. In DT algorithm, the recognition model is initially built up from a labeled training data set. The second step is named as decision tree classification (DTC). To classify an unknown instance, it is routed down the tree according to the values of the attributes tested in successive nodes, and when a leaf is reached the instance is classified according to the class assigned to the leaf.

The DT algorithms are often designed to work in a centralized way, which does not

work well in many of the distributed and ubiquitous applications. In distributed systems like WSNs, the data are distributed among all sensor nodes. Transmission of all the sensor data to a centralized node for processing is neither desirable nor feasible because of limitations on power supply, bandwidth, storage and processing. The energy challenge is the major constraint to WSNs, for the lifetime of the node is mainly determined by the power supply and frequent battery replacement is not a feasible option [3, 4]. Distributed data processing can be used to decrease the data transmission, increase the efficiency of power consumption and balance the storage and computation among sensor nodes.

This paper proposes a new distributed decision tree classification (DDTC) method. The classification model of the decision tree consists of a conjunctive set of linear classifiers, each of which relates to one attribute of sensor data. The idea of DDTC is that the linear classifiers can be reordered and grouped according to the related attributes. If attributes in each group of linear classifiers can be calculated using the sensor data of one sensor node, the DTC can be executed in a distributed way, *i.e.* by a mobile agent serially visiting the sensor nodes, each of which is corresponding to one attribute group and related linear classifiers. Since the linear classifications done on one sensor node do not use the sensor data of other sensor nodes, the transmission of sensor data is not necessary any more. At the same time, the classification result obtained in this distributed way is the same as the one obtained from the traditional centralized method.

This paper is organized as follows. Section 2 introduces the DDTC algorithm in details. Performance evaluation of DDTC and comparison between DDTC and the centralized DTC are presented in section 3. Section 4 discusses the advantages and disadvantages of DDTC and its application environment. Section 5 concludes this paper.

2. DISTRIBUTED DECISION TREE CLASSIFICATION ALGORITHM

2.1 Decision Tree Classification (DTC)

DT algorithm builds pattern classifier from a labeled training data set using a divide-and-conquer approach. To build up a DT model, it recursively select the attribute that is used to partition the training data set into subsets until each leaf node in the tree has uniform class membership. At each partition node, one appropriate attribute is selected and the optimal threshold is determined based on the entropy measurement to produce the greatest information gain [5], which assures the training samples can be well separated. Each intermediate node of a DT can have multiple branches. In order to simplify the analysis, in the following discussions, we take the binary DT as an example, in which each non-leaf node has two branches as shown in Fig. 1. The characters in the rectangle represent which feature or attribute of samples, A_i, is used to classify the data. The number near the rectangle is the optimal threshold value for linear classification using the attribute A_i. The leaf nodes are the final classification results in predefined classes.

During classification, a sample of unlabeled data propagates from the root node of the tree and is classified at each intermediate node (*i.e.* non-leaf node) using its corresponding attribute. When the sample reaches a leaf node, it is classified into a predefined class.
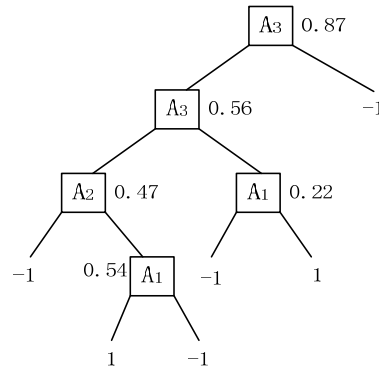
Fig. 1. An example of built decision tree.

2.2 Distributed Decision Tree Classification (DDTC)

A classification path from the root node to a leaf node consists of a conjunctive set of linear classifiers and can be represented as an if-then rule. Every rule consists of several conjunctive sub-conditions and one class value. Therefore, a DT can be decomposed into a set of if-then rules. We organize these rules according to the following two steps: (1) As some attributes may appear several times in the condition part of one rule, we combine these inequations for each attribute and number these rules; (2) Arranging the attribute inequations for all the rules in a specific order so that the attribute inequations related to more rules can be calculated earlier. The organized if-then rules of the DT in Fig. 1 are shown in Fig. 2.

1	If	$A_3 \leq 0.56$ and		$A_2 \leq 0.47$	Then	$class = -1;$
2	If	$A_3 \leq 0.56$ and	$A_1 \leq 0.56$ and	$A_2 > 0.47$	Then	$class = 1;$
3	If	$A_3 \leq 0.56$ and	$A_1 > 0.56$ and	$A_2 > 0.47$	Then	$class = -1;$
4	If	$0.56 < A_3 \leq 0.87$ and	$A_1 \leq 0.22$		Then	$class = -1;$
5	If	$0.56 < A_3 \leq 0.87$ and	$A_1 > 0.22$		Then	$class = 1;$
6	If	$A_3 > 0.87$ and			Then	$class = -1.$

Fig. 2. The organized if-then rules of the DT in Fig. 1.

Fig. 2 indicates that:

(1) All the sub-conditions of the rules can be operated one by one according to a specific order of the attributes. For instance, the operation order can be set as $A_3 \rightarrow A_1 \rightarrow A_2$ in above example.
(2) There is only one rule to be true for each test sample.
(3) If a rule does not need all the attributes for operation, the classification result can be obtained halfway and no further operation is needed.

The rules of a DT can be divided into groups of sub-conditions according to the related attributes. The DDTC can be realized by operating these groups of sub-conditions serially and each group operation is carried out on its corresponding sensor node in a WSN. Here it is assumed that each attribute in a DT only relates to the sensor data from a single sensor node in a WSN and vice versa.

After determining the itinerant order, some rules need to be modified accordingly because not every rule is related to the same attributes. If one rule does not include an attribute while the other rules do, the DDTC will still visit the sensor node related to that attribute according to the itinerant order. For example, in Fig. 2 the attribute A_1 does not exist in rule 1 but exists in other rules. A simple method is to add a dumb inequation of $-\infty < A_1 < +\infty$ in rule 1, which does not require any operation on the sensor node. However, if the missed attributes are not in the middle of the itinerant order, such as the attributes A_1 and A_2 in rule 6, the rule does not need to be modified.
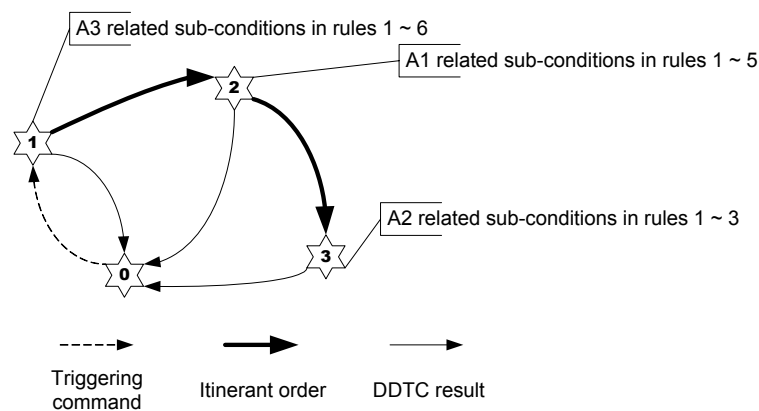

Fig. 3. The allocation of grouped sub-conditions to different sensor nodes.

Fig. 3 shows the allocation of three attribute groups to three sensor nodes. Node 1 includes all the A_3 related sub-conditions. Node 2 includes all the A_1 related sub-conditions. All A_2 related sub-conditions are in node 3. Node 0 is used as a manager node to trigger the itinerancy and get the DDTC result. In some cases, the recognition needs to be carried out only when the application requires. That's why a specific trigger is sent by node 0 rather than node 1 starts automatically. The DDTC result can be obtained before finishing the whole itinerary and sent to node 0.

The itinerary planning, rule organization and allocation can be achieved after the training samples are collected and the DT model is built. This usually takes place on a central processing unit, for example, a PC.

2.3 Mobile Agent Based Realization of DDTC

The mobile agent technology [3] is used to realize the DDTC algorithm. A mobile agent has a data structure of (*Identification*, *Time stamp*, *Rule vector*, *Itinerary vector*). The *Identification* uniquely identifies each mobile agent. The *Time stamp* indicates the

time instances of starting the DDTC. All the attributes used for one classification on the sensor nodes must have the same time stamp as the mobile agent's. Each item in the *Rule vector* represents result of the condition part of corresponding rule: 0 represents the condition part of this rule is false, 1 represents it is true. The initial value of each item is 1. The *Itinerary vector* is the visiting order of sensor nodes.

Each sensor node stores a data structure as (*Rule number*, *Inequation*, *Class*). This is static after the initialization of DDTC. The *Rule number* identifies the rule. The *Inequation* is the operation condition of the attribute for the rule on this sensor node. The *Class* is the classification result if the rule terminates at this node. If the rule does not terminate, the classification result is null. Table 1 depicts the data structure on the three nodes for the above example.

Table 1. The data structure of three sensor nodes for the above example.

Node 1			Node 2			Node 3		
Rule No	Inequation for A_3	Class	Rule No	Inequation for A_1	Class	Rule No	Inequation for A_2	Class
1	$(-\infty, 0.56]$	Null	1	$(-\infty, +\infty)$	Null	1	$(-\infty, 0.47]$	-1
2	$(-\infty, 0.56]$	Null	2	$(-\infty, 0.54]$	Null	2	$(0.47, +\infty)$	1
3	$(-\infty, 0.56]$	Null	3	$(0.54, +\infty)$	Null	3	$(0.47, +\infty)$	-1
4	$(0.56, 0.87]$	Null	4	$(-\infty, 0.22]$	-1			
5	$(0.56, 0.87]$	Null	5	$(0.22, +\infty)$	1			
6	$(0.87, +\infty)$	-1						

When a DDTC process starts, the manager node dispatches a mobile agent and the mobile agent visits the first sensor node in the itinerary vector with a time stamp *t*. This node calculates the attribute value at time *t* and operates the inequations in all the sub-conditions. If an inequation result is false, the related item in the rule vector of the mobile agent is changed to 0. Otherwise the value remains 1. If the condition result is true and the class value is not null, the class value of this rule will be sent to the manager node directly and the classification is finished. Otherwise, the mobile agent with updated rule vector will be sent from this node to the next node in the itinerary vector. This process continues until the itinerancy terminates at a certain node or reaches the last node.

3. PERFORMANCE EVALUATION

We mainly provide the quantitative analysis and simulation results of energy consumption to evaluate the performances of DTC and DDTC. The performance of DDTC on bandwidth, storage and computation cost will be discussed in section 4.

3.1 Energy Consumption

Sensor nodes are composed of four basic units [6]: a sensing unit, a processing unit,

a communication unit and a power unit. The energy consumed in sensing is the same in DTC and DDTC so that this factor can be neglected in our following discussion.

The energy consumption depends on four components: energy consumed in transmitting, receiving, overhead processing and data processing. In DTC, the migration unit is the file of raw data or preprocessed data. The overhead processing refers to accessing or dispatching the data file. In DDTC, the migration unit is the mobile agent. The overhead processing refers to accessing or dispatching the mobile agent. For one migration unit, the energy consumed in once transmitting, receiving, overhead processing and data processing are charactered as E_{tx}, E_{rx}, E_{oh} and E_{proc}, respectively. In our simulation, we assume the migration units of these two paradigms have the same size of S. Then, we can roughly assume E_{tx}, E_{rx} and E_{oh} in DTC are equal to those in DDTC, and the energy needed to process on the center node in DTC is equal to that on each visited node in DDTC.

In DTC, for one classification, each client node needs to dispatch the data file and transmit it to the center node, which consumes $E_{oh} + E_{tx}$. n client nodes totally consumes $n(E_{oh} + E_{tx})$. The center node needs to receive and access all the n data files, then compute the classification result, which consumes $nE_{rx} + nE_{oh} + E_{proc}$. The energy consumption in DTC for one complete classification, E_{DTC}, is:

$$E_{DTC} = n(E_{oh} + E_{tx}) + nE_{rx} + nE_{oh} + E_{proc}$$
$$= nE_{tx} + nE_{rx} + 2nE_{oh} + E_{proc}. \qquad (1)$$

The maximum energy consumption on one node for one classification, $\max(e_{DTC})$, is the energy consumed on the center node. That is,

$$\max(e_{DTC}) = nE_{rx} + nE_{oh} + E_{proc}. \qquad (2)$$

In DDTC, for one classification, the manager node needs to dispatch the mobile agent and transmit it to the first visited node at the beginning of the classification, and receive and access the result at the end of classification. Thus, the manager node totally consumes $E_{oh} + E_{tx} + E_{rx} + E_{oh}$. Each visited node needs to receive and access the mobile agent and, after related processing, dispatch and transmit the updated mobile agent to the next node in the itinerary vector (if the classification result is obtained, it needs to dispatch and transmit the result to the manager node), which consumes $E_{rx} + E_{oh} + E_{proc} + E_{oh} + E_{tx}$. The energy consumption in DDTC for one complete classification, E_{DDTC}, is:

$$E_{DDTC} = E_{oh} + E_{tx} + m(E_{rx} + E_{oh} + E_{proc} + E_{oh} + E_{tx}) + E_{rx} + E_{oh}$$
$$= (m + 1)E_{tx} + (m + 1)E_{rx} + (2m + 2)E_{oh} + mE_{proc} \qquad (3)$$

where m is the number of visited sensor nodes in DDTC. The maximum energy consumption on one node for one classification, $\max(e_{DDTC})$, is the energy consumed on the visited node. That is,

$$\max(e_{DDTC}) = E_{tx} + E_{rx} + 2E_{oh} + E_{proc}. \qquad (4)$$

3.2 Simulation Results

Two public data sets are used to evaluate the energy consumption in DTC and DDTC. They are *Glass Identification* and *Pima Indians Diabetes* [7]. *Glass Identification* has 9 numeric attributes and 214 instances belong to 6 classes; *Pima Indians Diabetes* has 8 numeric attributes and 768 instances belong to 2 classes. Though these two data sets are not collected from sensor nodes in WSNs, we accommodate them to the distributed environment, *e.g.* a WSN, based on the following assumption. For each data set, we suppose the network has as many sensor nodes as attributes (each attribute relates only to one sensor node). There is one additional node served as the center node in DTC or the manager node in DDTC. Each instance in the data set represents the attribute values on these sensor nodes with the same time stamp. The class label of this instance is the classification result.

Table 2. The itinerary vector of these two data sets in DDTC.

Data set	Visiting order						
	1st	2nd	3rd	4th	5th	6th	7th
Glass Identification	1	8	2	5	7	3	4
Pima Indians Diabetes	2	5	8	6	4	7	1

Before classification, 100 instances are randomly selected from each data set as the training set to build up the DT model using the pruned C4.5 algorithm. According to the rationale of DDTC, the rules are organized and the itinerary vector is determined as shown in Table 2. We can find that the 6th and 9th attributes are vacant in the itinerary vector of *Glass Identification* data set and the 3rd attribute is vacant in the itinerary vector of *Pima Indians Diabetes* data set. That's because the DT algorithm has the capability to select only the related attributes from all attributes. The unused attributes have no contribution to the classification and, therefore, are excluded in both DTC and DDTC.

The center node in DTC only receives the messages and needs not transfer the result to another node, while the manager node in DDTC needs not only dispatch the mobile agent, but also receive the result. DDTC need transmit one more time for each classification. Thus, for one classification, the number of transmissions is $m + 1$ if the mobile agent has visited m sensor nodes before it obtains the result in DDTC. In traditional DTC, the number of transmissions is always the number of used attributes, n. The correlation between the average number of transmissions for one classification and the number of instances in DTC and DDTC is shown in Fig. 4.

In Fig. 4 (a), the average number of transmissions in DDTC is a little larger than that in DTC at the beginning, and then it decreases gradually and is less than the number of transmissions in DTC when the number of instances exceeds 160. This indicates that the mobile agent need visit almost all the 7 sensor nodes for the prior classified instances in the *Glass Identification* data set, while classifying the posterior instances need visit fewer sensor nodes. In Fig. 4 (b), the average number of transmissions for one classification in DDTC is about 5, which is smaller than that in DTC. That is to say, averagely the mobile agent only need visit 4 sensor nodes to classify one instance in the *Pima Indians Diabetes* data set.

(a) Glass identification.   (b) Pima Indians diabetes.
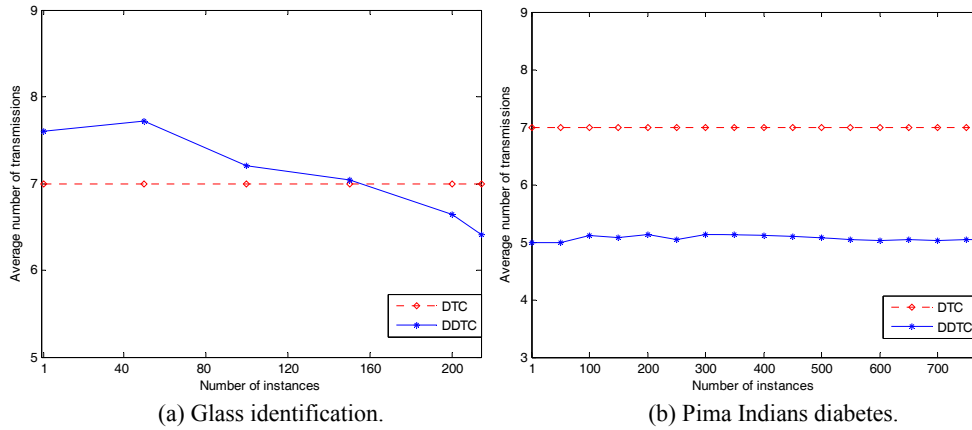
Fig. 4. The correlation between the average number of transmissions for one classification and the
number of instances in DTC and DDTC.

The above simulation indicates that, if the average number of visited sensor nodes $\bar{m}$ ($1 \leq \bar{m} \leq n$) for one classification is no larger than $n - 1$, DDTC can classify the instances with fewer transmissions than DTC.

With the average number of transmissions in Fig. 4, we can calculate the simulation results of average energy consumption for one classification in both DTC and DDTC. In our simulation, the wireless nodes in the network are based on the MicaZ platform with Atmel ATMega128L microcontroller and Chipcon CC2420 radio transceiver. The practical network transfer rate and clock frequency are 100 kbps and 7.37 MHz, respectively [8].

Suppose the migration units of these two models have the same size of 1 kbit (125 bytes), the time spent in transmitting or receiving one migration unit, t_{tx} or t_{rx}, is $t_{tx} = t_{rx} = 1kbit/100kbps = 0.01s$. According to [9], accessing one byte from the internal SRAM of ATMega128L is performed in two clock cycles. That is, accessing one byte needs $2/7.37 \times 10^6 Hz = 2.71 \times 10^{-7}s$. Thus, the overhead time for accessing one migration unit, t_{oh}, is $t_{oh} = 125 \times (2.71 \times 10^{-7}s) = 3.39 \times 10^{-5}s$. The processing time, t_{proc}, is set to be equal to the overhead time in these two models if we assume that it also takes two clock cycles to process one byte.

The energy consumption is calculated as the product of operating voltage, operating current and operating time. The operating voltage of the nodes is 3V. According to [8] which has the same configuration as our simulation, the operating current for transmitting, receiving and microcontroller operation are 17mA, 19.7mA and 12mA respectively. Then, the energy consumed in transmitting one migration unit, E_{tx}, is *Voltage × Current × Time* = $5.10 \times 10^{-4}J$. In the same way, $E_{rx} = 5.91 \times 10^{-4}J$, and $E_{oh} = E_{proc} = 1.22 \times 10^{-6}J$.

The average energy consumption for one classification can be calculated with the average number of transmissions from Fig. 4 based on Eqs. (1) and (3). Fig. 5 shows the correlation between the average energy consumption for one classification and the number of instances in both DTC and DDTC. As E_{tx} and E_{rx} are much larger than E_{oh} and E_{proc}, the average energy consumption for one classification is determined by the average number of transmissions. Therefore, Fig. 5 has the similar characters with that of Fig. 4.

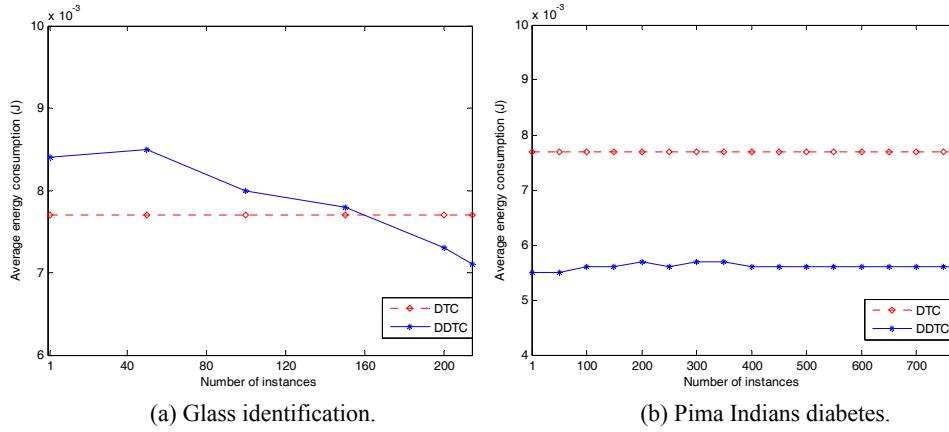(a) Glass identification.　　　　　　(b) Pima Indians diabetes.

Fig. 5. The correlation between the average energy consumption for one classification and the number of instances in both DTC and DDTC.

As defined in [10], the lifetime of a wireless sensor network refers to the time from node deployment to the time when the first node is out of operation due to energy depletion. In the homogeneous sensor network where all the nodes are identical, the lifetime of the network is determined by the sensor node who consumes the maximum energy.

With the attribute number of the data set and the simulation results of E_{tx}, E_{rx}, E_{oh} and E_{proc}, the maximum energy consumption on one sensor node for one classification in DTC, $\max(e_{DTC})$, and that in DDTC, $\max(e_{DDTC})$, can be calculated according to Eqs. (2) and (4), respectively. The above two data sets both have 7 nodes. In DTC, for one classification, the center node consumes the maximum energy and $\max(e_{DTC}) = 0.0041J$. In DDTC, for one classification, the visited node consumes the maximum energy and $\max(e_{DDTC}) = 0.0011J$.

From Eqs. (2) and (4) we can also see that $\max(e_{DTC})$ will go up with the increase of the number of sensor nodes (attributes), while $\max(e_{DDTC})$ is constant. When there are many nodes (attributes), $\max(e_{DTC})$ will be much larger than $\max(e_{DDTC})$. Thus, the center node becomes the bottleneck of the network in DTC. In DDTC, the energy consumption is balanced among the nodes and, compared with DTC, it can prolong the lifetime of the network.

4. DISCUSSION

Above simulation results indicate that the DDTC algorithm can decrease the number of transmissions, balance the power consumption among sensor nodes and, therefore, prolong the lifetime of the system in the homogeneous sensor network. In addition, DDTC also has another several advantages: (1) It needs lower bandwidth because raw data or preprocessed data don't need to be transmitted to the center node simultaneously; (2) DDTC divides the DT rules into a set of sub-conditions and each sensor node only stores its own attribute group, which balances the storage among the sensor nodes; (3) Each sensor node only calculates the inequations in its own attribute group, which also distributes the computation cost among the sensor nodes.

On the other hand, DDTC has several disadvantages: (1) DDTC is likely to increase the execution time because the mobile agent needs to visit the sensor nodes serially; (2) In DDTC, if one node in the itinerary vector breaks down, the classification task which needs to visit this failed node can not be finished; (3) DDTC needs to define the itinerary vector, organize and allocate the classification rules. Its initialization is a little more complex than that of DTC. If the DT model needs to be reconstructed, the new classification rules should be organized and allocated again.

From above discussion we can see that DDTC algorithm is preferred in homogeneous sensor network. However, it should be pointed out that, in the heterogeneous sensor network which has a powerful center node with sufficient power supply, bandwidth, storage and processing capability, the DTC algorithm is better because it can overcome these three disadvantages of DDTC mentioned above.

5. CONCLUSION

Classification is widely used to recognize the patterns of sensor readings in sensor networks. Usually the data are transferred to a center node where the classification is carried out. We present a mobile-agent-based DDTC algorithm in this paper. It organizes the conjunctive sets of classification rules and allocates each group of sub-conditions to the corresponding sensor node. A mobile agent visits these sensor nodes serially and obtains the classification result when it reaches the last sub-condition of one rule.

DDTC can decrease the number of transmissions and increase the efficiency of power consumption when the itinerary terminates in the halfway. Compared with the centralized DTC, DDTC can balance the computation, storage and energy consumption among sensor nodes, and prolong the lifetime of the network.

ACKNOWLEDGMENTS

REFERENCES

1. N. Chen, W. Fontijn, M. Chen, and Q. Zhang, "A framework for ambient applications-context-based autonomous extension of applications," in *Proceedings of IEEE International Conference on Sensor Networks*, 2005, pp. 348-354.
2. A. K. Jain, R. P. Duin, and J. Mao, "Statistical pattern recognition: a review," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 22, 2000, pp. 4-37.
3. H. Qi, Y. Xu, and X. Wang, "Mobile-agent-based collaborative signal and information processing in sensor networks," *Proceedings of the IEEE*, Vol. 91, 2003, pp. 1172-1183.
4. C. Chong and S. P. Kumar, "Sensor networks: evolution, opportunities, and challenges," *Proceedings of the IEEE*, Vol. 91, 2003, pp. 1247-1256.
5. I. H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and*

Techniques, Morgan Kaufmann, San Francisco, 2005.

6. V. Raghunathan, C. Schurgers, S. Park, and M. B. Srivastava, "Energy-aware wireless microsensor networks," *IEEE Signal Processing Magazine*, Vol. 19, 2002, pp. 40-50.

7. C. Blake, E. Keogh, and C. J. Merz, "UCI repository of machine learning databases," http://www.ics.uci.edu/~mlearn/MLRepository.htm, Department of Information and Computer Science, University of California, Irvine, CA, 1998.

8. Crossbow Technology Inc., "MPR/MIB user's manual," Document 7430-0021-06, http://www.xbow.com/, 2004.

9. ATMEGA128L datasheet, http://www.atmel.com/.

10. Y. Xu, "Energy efficient designs for collaborative signal and information processing in wireless sensor networks," Ph.D. Dissertation, The University of Tennessee, 2005.

Shuang-Quan Wang (王雙全) received his M.A. degree from Wuhan University of Technology, China, in 2004. Currently, he is a Ph.D. candidate in the Institute of Image Processing and Pattern Recognition at Shanghai Jiaotong University, China. From 2004 to 2006 he was an intern in Philips Research East Asia (PREA) and was involved in the project of Intelligent Environment Controlling Based on Sensor Networks. His research interests include machine learning, data mining, and wireless sensor networks.

Xin Chen (陳鑫) received her M.A. degree in Electronic Engineering from Fudan University, China, in 2001. Since then she joined Philips Research East Asia (PREA). She has been working on projects of video code and streaming multimedia. Her current research interests include activity recognition, data fusion, and applications of wireless sensor networks.

Ning-Jiang Chen (陳寧江) received the B.S. degree in Automatic Control and the Ph.D. degree in Pattern Recognition and Intelligent System from Shanghai Jiaotong University in 1995 and 2001, respectively. In 2001, he joined Philips Research East Asia (PREA). His current research interests include data analysis in wireless sensor networks, healthcare applications, activity recognition, and data mining.

Jie Yang (楊杰) received his Ph.D. degree from Department of Computer Science, Hamburg University, Germany in 1994. Currently, he is a professor of Institute of Image Processing and Pattern Recognition, Shanghai Jiaotong University, China. He has taken charge of many research projects (*e.g.* National Science Foundation, 863 National High Tech. Plan) and published one book in Germany and more than 200 journal papers. His major research interests are object detection and recognition, data mining, and medical image processing.