Research Article

An Improved Trust Model Based on Interactive Ant Algorithms and Its Applications in Wireless Sensor Networks

Yun Pan, Yafang Yu, and Li Yan

School of Computer Science, Communication University of China, Beijing 100024, China

Correspondence should be addressed to Yun Pan; pany@cuc.edu.cn

Received 4 January 2013; Revised 29 March 2013; Accepted 3 May 2013

Academic Editor: Motoki Sakai

Copyright © 2013 Yun Pan et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Marmol et al.’s ant algorithm based trust model is improved from several aspects: new interactive ant algorithm, new node types, more reasonable meta-assumption on node behaviors, new trust evaluation function, new penalty mechanism, and so on. Simulations on identifying malicious nodes and electing cluster head show that the proposal is effective and can observably reduce the packet drop ratios.

1. Introduction

Wireless sensors are small and cheap devices powered by low-energy batteries, equipped with radio transceivers, and responsible for monitoring physical or environmental conditions, such as temperature, humidity, sound, pressure, motion, and anything we are interested in. They are featured with resource (e.g., power, storage, and computation capacity) constraints and low transmission rates [1]. Wireless sensor networks (WSNs) are networks based on such wireless sensors cooperation. They can be used in many industrial and civilian application areas, including industrial process monitoring and control, machine health monitoring, environment monitoring, healthcare applications, home automation, and traffic control.

However, the wireless sensor nodes are usually deployed in open environment where an adversary may easily capture sensor nodes and subsequently use these nodes to attack the whole network [1]. Therefore, it is desirable to identify the compromised nodes in time and then kick them out so as to avoid the whole network to be controlled by the adversary [2]. Traditional security system is usually based on cryptography, which requires complexity encryption and decryption operations. But some cryptographic countermeasures are not efficient and smart enough to be deployed in wireless sensor networks where sensors have limited communication bandwidth, memory, and computing power [3]. Although research on lightweight cryptography makes great progress on dealing with this problem, many currently available cryptographic components are far from lightweight. For example, the reported most lightweight implementation of elliptic curve cryptography (ECC) requires about 10,000 GEs (i.e., equivalent gates), but in general there is no more than 6000 GEs that are left for deploying security in typical sensors.

Therefore, it is interesting to probe effective non-cryptographic mechanism for identifying corrupted nodes, and then take the corresponding countermeasures to elevate the losses. Trust management technique is regarded a good complementary toward the system security protections based on cryptographic mechanism [3]. Since Marsh introduced the concept of computable trust model, many researchers proposed various trust models for different scenarios. Recently, a lot of trust models for WSN-oriented applications were proposed, such as for securing routing, data aggregation [4], cluster head selection [5], and synthesized trust management systems [6]. All these trust models are featured as classical computational patterns. In 2008, Mármol and Pérez proposed a novel trust model—BTRM-WSN that employs a smart-warm intelligent optimization algorithm—ant algorithm—into trust management model. Although at present any ant algorithm can merely be implemented by adopting classical computational pattern, its typical bionic computational pattern enables it powerful capability for
global optimization and this in turn introduces new promising properties on trust management. Although Már mol and Pérez [7] attested the advantages of BT RM-WSN model by using amount of simulations, this model suffers from too many specializations and constraints, resulting in a very narrow scope on its application. Considering that interactive ant algorithms are more effective than noninteractive ones, our main motivation is to, based on Marmol’s work, improve the efficiency with interactive multiple ant colony algorithm and extend the suitability of BT RM-WSN model by proposing new improvements from node types, node functionalities and trust value increasing, penalty function, and so forth.

The rest contents are organized as follows. In Section 2, we give an introduction on the new trust model based the BTRM-WSN model and our improvements; in Section 3, we conduct detailed simulations on our trust model; in Section 4, as a typical application of the proposed model, we, based on interactive ant colony algorithm, present a trust cluster head election framework for WSN environments. Finally, the concluding remarks are given in Section 5.

2. Trust Model

In [7], the authors presented a trust model for WSNs, called BTRM-WSN, based on ant colony systems aiming to help a node requesting a certain service to the network and find the most trustworthy route leading to a node providing the right requested service. Experiments and results demonstrated the accuracy and robustness of this model. Based on the original BTRM-WSN model, we introduce the following improvements to enhance its efficiency and scope.

2.1. Interactive Multiple Ant Colony Algorithm (IMACA). Interacted Multiple Ant Colonies Optimization method just like Ant Colony System (ACS) is a bioinspired algorithm. In IMACA there are also two levels of interaction. One is the colony level and the other one is the population level [8].

The activities of a single colony in IMACA method are based on ACS. Each colony has its own pheromone that is used as an interaction between the ants of the same colony. The interaction between ant colonies using pheromone can be organized in different terms [8]. The IMACA algorithm is described as follows. M colonies of m ants each are working together to solve some combinatorial problem. Let ant k which belongs to the colony v be at node r at a certain moment. The probability of moving towards node s ∈ N^kv_r, where N^kv_r is the set of remaining neighbors not visited yet by ant kth ant of colony v, is computed as

\[
p_k^v(r, s) = \begin{cases} 
\frac{f(P_{ru})H_{ru}^β}{\sum_{u \in N^kv_r} f(P_{ru})H_{ru}^β} & \text{if } s \in N^kv_r \\
0 & \text{otherwise},
\end{cases}
\]

where \(f(P_{ru})\) is the evaluation function of pheromone on the edge \((r, s)\), \(H_{ru}\) is the problem dependent heuristic, \(β\) is a value used to determine the relative importance of pheromone versus heuristic, \(q_0 \in [0, 1]\) is a constant, and \(q \sim [0, 1]\) is a random number within the interval \([0, 1]\), while \(S\) is a random variable selected according to the following probabilistic formula:

\[
S = \begin{cases} 
f(P_{ru})H_{ru}^β & \text{if } s \in N^kv_r \\
0 & \text{otherwise}.
\end{cases}
\]

If \(q \leq q_0\), the most promising node is selected as the next step of ant \(k\) using expression (1); otherwise, that node is chosen using expression (2).

The pheromone evaluation function of IMACA evaluates the pheromone on an edge as a composition between the pheromone values of the ant own colony and the value of the pheromone evaluation function based on some pheromone evaluation rate [8]. An ant builds \(γ\) is the pheromone evaluation rate between \([0, 1]\) of its decision based on its own colony’s experience and the other based on others. The pheromone evaluation function is computed as

\[
f(P_{ru}) = γP_{rs}^v + (1 − γ)\frac{\sum_{v=1}^{M} P_{rs}^v}{M},
\]

where \(P_{rs}^v\) is the pheromone of colony \(v\) on the edge \((r, s)\).

Just like ACS, there are two kinds of pheromone updating: a local and a global one. Local pheromone update is then applied by each ant on the visited edges. The local pheromone update is defined as

\[
P_{rs}^v = (1 − ϕ)P_{rs}^v + ϕP_0,
\]

where \(ϕ\) is a pheromone evaporation parameter and \(P_0\) is the initial pheromone value.

Global pheromone updating includes that the best ant of each colony deposits an amount of pheromone on its own path. The ant can find the best solution according to the following rule:

\[
P_{rs}^v = (1 − ρ)P_{rs}^v + ρΔP_{rs}^{v,bs}
\]

\(ρ \in [0, 1]\) is the trail evaporation parameter and \((1 − ρ)\) represents the pheromone persistence. \(ΔP_{rs}^{v,bs}\) is the pheromone quantity added to the connection \((r, s)\) belonging to the best solution of vth colony \(L^{v,bs}\) and is computed as [8]

\[
ΔP_{rs}^{v,bs} = \begin{cases} 
\frac{1}{L^{v,bs}} & \text{if } (r, s) \text{ belongs to the best tour of colony } v \\
0 & \text{otherwise}.
\end{cases}
\]

2.2. New Trust Model Using IMACA. By adopting new interactive ant algorithm IMACA described above, we propose a new trust model that can be viewed as an improvement on Már mol and Pérez original model BTRM_WSN [7].

2.2.1. Node Types. In the original model, nodes are divided into two disjoint sets: the set of client nodes and the set of server nodes. This separation exerts many constraints for
WSNs by requiring that server nodes cannot request services and client nodes cannot provide services. However, in a distributed ad hoc network, each node could be server and client. In our improved model, the node types are aggregated such that a node can simultaneously request services from others and provide services for others. Apparently, this model is even more suitable for actual WSNs.

2.2.2. Node Behavior. In the original model, the behavior of malicious nodes is assumed to provide malicious services but does not specify what kind of malicious services is, and does not consider the infectious factors of packet loss rate of a node. In addition, in the original model, all client nodes are assumed to be trusted and without fault behavior. This makes the original model far from the real WSN environments. Therefore, according to node behaviors described in [3,9,10], a malicious is specified a very high packet loss rate (between 0% and 40%), while the packet loss rate of a trust node is specified between 0% and 10%.

2.2.3. Diagram of the New Trust Model. Based on the improvement of node types and node behavior, before a node requests to another node, it would perform the following steps.

(1) Let many ants of multiple ant colonies search path. These ants select the next step according to expressions (1) and (2).

(2) When ants select the next hop node, update pheromone value of the edge to the selected node. Every time an ant moves from one node to another, the pheromone local updating is carried out through the following expression:

\[ P_{rs}^v = (1 - \varphi) \cdot P_{s,t_2}^v + \varphi \cdot \Omega, \]

where

\[ \Omega = \left(1 + (1 - \varphi) \cdot \left(1 - P_{s,t_2}^v H_{s,t_2}\right)\right) \cdot P_{s,t_2}^v \cdot \cdots \cdot P_{s,t_2}^v. \]

After all ants find the best path, global updating is applied on those edges belonging to this path by using the following expression:

\[ P_{rs}^v = (1 - \varphi) \cdot P_{rs}^v + \varphi \cdot \left(1 + P_{rs}^v H_{rs} \Delta P_{rs}^{bs}\right) \cdot P_{rs}^v \]

At last, merge all the pheromone of all colonies using the following expression:

\[ P_{rs} = \frac{\sum_{\gamma=1}^{M} P_{rs}^\gamma}{M}. \]

(3) When ants find the node to meet the requirements, return to the initial node in the same way. And record the path information.

(4) Each time a launched ant returns to its node carrying a path or solution. Among all possible paths, that node would like to choose a path that has the best quality. The path quality computation can be done in the following way:

\[ Q(S_k) = \frac{\overline{P}_k}{\sqrt{\text{Length}(S_k)}} \cdot A_k\%, \]

where \( \overline{P}_k \) is the average pheromone of the path found by ant \( k \) and \( A_k \) represents the percentage of ants that have selected the same solution as ant \( k \).

After the node selects the best path, it requests service by the path. If the provided service cannot meet the requirements, the neighbor of the service node reduces the pheromone values with

\[ P_{rs} = (P_{rs} - \varphi) \cdot S_{at}, \]

where \( S_{at} \in [0, 1] \) means satisfaction that the node is about the service.

2.2.4. Trust Value Evaluation. In addition, there is no original trust value in BTRM-WSN. Every node maintains a set of pheromone trace \( s \) for all neighbors and these pheromone traces will determine the probability of ants choosing a certain route or another. Therefore, the pheromone trace can be treated as the amount of trust. However, in order to distinguish between pheromone and trust value, we defined \( T_i \) as the trust value of node \( i \), where \( 1 \leq i \leq k \). And trust value can be computed based on pheromone trace as follows:

\[ T = \frac{\sinh ((6.2 + \alpha) \cdot p - 3.1 - \alpha/2)}{\sinh (3.1 + \alpha/2)}, \]

where \( p \in [0, 1] \) is the pheromone value of the edge connected to a node, while \( \alpha \) means the degree of strictness.

In our model, each node maintains a trust table in which it records the trust value of its neighbors, as is shown in Figure 1. Node i has 3 neighbor nodes m, n, and o. Based on the pheromone trace of each edge, node i’s trust table is shown on the right side.

2.2.5. Punish and Reward. After using trust value, the punishment mechanism is also improved. In the original model, the punishments and rewards are designed in order to find a trusted service node, which involves the whole path. For example, the client node C wants to find the optimal path to the service node S: C \( \rightarrow \) A \( \rightarrow \) B \( \rightarrow \) D \( \rightarrow \) S. According to the service provided by S, C determines that the service does not meet its requirements so that this path would be punished by reducing the value of all the edges of the pheromone. If adopt our adding trust value (see Section 2.2.4), the trust value is a single increasing function of the value of the pheromone, and then reducing the pheromone value means to reduce the trust value of a node. If only S node is malicious node and other nodes are trusted node on this path, then the trust node’s trust value will be reduced due to the impact of malicious nodes, and such punishment is clearly unreasonable.

Therefore, the entire punishment mechanism will be modified to only reduce the trust value of the malicious
node without an impact on other nodes. In this way, we can guarantee the correctness of the trust value. For ease of description we modify the punishment, when we will use an example to illustrate. Suppose that node C sends the packet to node S, and the improved model throughout the process is as follows.

(1) C sets $n$ ants carrying the control packet to search path. These ants select the next hop node in accordance with the state transition probability formula.

(2) When ants choose the next hop node N, detecting whether the node forward the control packet. If so, the pheromone values leading to the edge of the node will be updated according to pheromone update formula; otherwise, the ants backtrack a hope to node N-1; during the return the node is punished by reducing r edge pheromone value using formula (12). And continue searching the neighbor nodes of the node N-1, until it reaches the destination node S. If the ants have been returned to the node C, all the neighbors of node C which have also been probing finished, and then the ants stop detection.

(3) When the ants find the path leading to the destination node S, backtrack to the node C and record the entire path information (including node information as well as the edge of the pheromone value).

(4) Node C calculates $s$ the quality of the road according to the return path of the ants brought information, and to select the best quality path from all the ants chose the path $s$.

(5) Node C sends packets to the node S through the optimal path.

(6) After node S received successfully packets, this path will be rewarded that the pheromone values of all edge s of the path will be increased.

2.2.6. Packet Drop Ratio. The packet drop ratio of nodes is introduced in order to make this model more realistic; while in BTRM-WSN, this is never considered as one of the influencing factors. Now the drop ratio of good nodes is 0.001, and bad ones is 0.4. [5, 11].

3. Simulations

In Section 2, we added trust value in the model and set the packet loss rate for nodes. In this section we refer to the experimental method in [6] to evaluate the performance of our improved model from two aspects: the first is to identify malicious nodes and the second is to reduce network packet loss rate. In our model, there is no difference between client nodes and server nodes. So we compare our trust model to Sun et al.’s model [6] instead of BTRM-WSN. The experiment simulation tool is TRMSim-WSN that was developed by Márval and Pérez for the trust model specifically for wireless sensor network simulation [9].

3.1. Identifying Malicious Nodes. 40 nodes are randomly scattered in the area of 100 m $\times$ 100 m (see Figure 2). Each over a certain time, the node randomly selects a node as a destination node to send data packets, respectively, in accordance with this improved model and Sun model.

3.2. Capability of Dynamically Identifying Malicious Nodes. In this experiment, we initially set nodes 1, 2, 3, 4 as the malicious nodes and assume that other 4 nodes will become malicious over every given time segments. That is, after 1 segment, the nodes 5, 6, 7, 8 become malicious; after 2 segments, nodes 9, 10, 11, 12 become malicious. Figure 3 shows the average pheromone value of the nodes in the network run this improved model obtained in different time periods the running Sun et al.’s model obtained in different time segments. The average probability value is shown in Figure 4.
Figure 3: Capability of the improved model for detecting malicious nodes.

Figure 4: Capability of Sun’s model for detecting malicious nodes.
Comparing these two graphs, we can find that their pheromone value or probability value will have a very significant change whenever a new node becomes malicious. In particular, the node’s pheromone value change range in our improved model is much larger than that of in Sun et al’s model. The reason is that there is no punishment mechanism in the Sun’s model, the trust value of the node only positive feedback, while in our improved model, the node’s trust value not only positive feedback but also negative feedback, leading to enhancement on the capability of identifying malicious nodes. This also means that even if a malicious node in the network was initially regarded as trusted, but as long as it has a malicious behavior, it will be identified.

3.3. Node’s Trust Table. In this experiment, we set the nodes 1, 2, 3, 4, 5 as malicious nodes. Figure 5 shows the average trust value of all nodes, and wherein, x-axis represents a node in the network ID, and the y-axis indicates the average trust value that calculates according to their neighbor nodes keep.

From Figure 5, we can see that the neighbor nodes of nodes 1, 2, 3, 4, 5 get the low trust value about these 5 nodes. The three nodes trust value is the lowest. Therefore who can determine the node according to the trust value can easily identify a malicious node.

3.4. Reduce Packet Loss Rate. 100 nodes are randomly scattered in the area of 100 m * 100 m. Each over a certain time, the node randomly selects a node as a destination node, and sending data packets in accordance with the improved model. Trust node packet loss rate of 0.01. Malicious node gray hole attacks, and packet loss rate is set to 0.65–0.75.

Figure 6 shows the malicious node in the network by a packet loss rate becomes 10 when, from the figure, we can see that when the packet loss rate is still relatively low, about 0.05, at a time when network in the number of malicious nodes has accounted for 20%.

Figure 7 shows the network of malicious nodes gradually increasing to 10% packet loss rate can be seen from the figure the malicious nodes in the network to reach 30%, and the packet loss rate of the network remains a low state. Thus, our improved model to ensure that the node transmits the packet path is a secure path, thereby reducing the probability of packet loss of the node.

3.5. Average of Path Length. Next, let 100 nodes randomly scattered in the area of 100 m * 100 m and use 5 different topologies for conducting simulation. The sink node is also randomly selected and the data packets were sent and forwarded according to the improved model. After the simulation, the average of the path length is counted and depicted in Figure 8.

In Figure 8, the red line indicates the average of path length in the case without malicious nodes and we take this as the baseline. From this figure, we can see that the average of path length differs slightly from the baseline along the increasing of the number of malicious nodes. That is, the average of path length is controlled within 1.5 hops. This suggests that the improved model does not infer much resource consuming toward each node, and the additional overhead is acceptable.
4. Application Cluster Head Election

The cluster head is the basis for cluster formation. Once a malicious node has been a cluster head, the consequences would be disastrous. As far as we know, the cluster routing protocols are largely based on this assumption: all wireless sensor nodes are trustworthy [5]. This assumption may naturally lead the malicious or compromised node to be selected as the cluster head, which could be a threat to the network. Therefore, we need an effective mechanism to identify the captured nodes and ensure the cluster head is trustworthy.

4.1. Cluster Head Election Based on New Trust Model. The cluster head election can be divided into two parts. One is the nodes updating their trust values; the other is the cluster head election. In the first part, nodes use modified BTRM-WSN to find the most trusted paths leading to the cluster head. During this process, the trust value of each neighbor is updated as well. In the second part, we refer to Garth’s mechanism [5, 7] to elect trustworthy cluster head. In this section, our framework will be introduced in two processes: cluster head and member nodes.

The flow chart of cluster head is shown in Figure 9. The major blocks are explained in details as follows.

1. When the current cluster head’s battery power level falls below a predetermined threshold or serve for a predetermined period of time, it broadcasts (within the cluster) a new election message.

2. After cluster members vote, the current cluster head then tallies the votes and decides the winner based on simple majority. The node with the second highest number of votes is selected as the vice cluster head. The purpose of the vice cluster head is to assume cluster head function in the event that the newly elected cluster head fails before handing over to its successor.

3. The new winner and the vice cluster head have to pass a challenge response from the current cluster head before they are allowed to take up office.

4. If one or both of them fail, the incumbent cluster head informs the cluster members and initiates a new election.

5. If they success, the cluster head multicasts the winner and runner-up to all the members of the cluster.

The flow chart of cluster members is shown in Figure 10. The major blocks are explained in details as follows.

1. All the members use the improved BTRM-WSN to find the most trusted path leading to the cluster head and send their data to cluster head. When they receive a notification about cluster head election, all nodes vote for a new cluster head.

2. They select a candidate node with the highest trust value from their trust table.

3. Then, they send votes to current cluster head. For greater security, the vote is encrypted. Neighbors therefore have no idea of the vote content of each other.

4. After the members receive a broadcast about new cluster head, they now communicate with the new cluster head.
4.2. Evaluation. In this section, we evaluate the performance of our framework in two aspects: (1) the capability in preventing compromised nodes from being selected as the cluster head; and (2) the standard deviation shows the average probability of selecting compromised nodes. We use TRMSim-WSN [6] as our main simulation platform.

During our experiments, we use a flat, rectangular area of 100 m \( \times \) 100 m. 50 nodes are randomly deployed and formed as one cluster. The nodes transmission distance is 18 meters. We launched our model 300 times over 5 random WSNs. Every 10 times, we initiate a new cluster head election.

4.2.1. Probability of Selecting Compromised Nodes. We increase the number of malicious nodes from 10\% gradually to 90\% to test the robustness of our model. The good nodes run our cluster head election algorithm, while the malicious ones pick up a candidate randomly from their neighbors. We omit the challenge response procedure to avoid complicating this simple system.

Figure 11 shows the average probability. For clusters with less than 30\% of compromised nodes, our mechanism almost never selects a compromised node. However, the probability increases rapidly after 70\% of the nodes were compromised. This can be explained that malicious nodes send false votes, which greatly interferes with the result of election. Figure 12 shows the comparison with Crosby's model [5]. A conclusion that can be obtained is that the accuracy rate of these two models, on preventing the malicious or compromised nodes from being a cluster head, is basically the same. This also demonstrates the effectiveness of our framework in securing cluster.

4.2.2. Standard Deviation. Figure 11 actually shows the average probability of selecting compromised nodes in our framework over 5 random WSNs. But an average probability 0.8, for instance, could be reached because the model always selected a trustworthy cluster head on probability 0.8, or just because it selected on 1.0 in half of the tested wireless sensor networks and 0.6 in the other half.

This is the reason why we decided to measure and show the standard deviation related to that average as well. Figure 13 shows the results.

We can see the standard deviation also remains quite low. This means our framework is able to select a trustworthy cluster head with a quite high accuracy, regardless the topology of the WSNs.

5. Conclusion

In this paper, we propose a new trust model by making several improvements toward Marmol et al’s BTRM-WSN model: adopting an interactive multiple ant colony algorithm,
introducing new node types and more reasonable meta-assumption, new trust value evaluation function and penalty function, and so forth. The resulted model is more efficient and suited for general WSN-oriented application scenarios. In particular, the new model is very effective in identifying malicious nodes and decreasing packet loss rate according to simulations. As a typical application on our proposal, we also present a cluster head election framework based on interactive ant colony algorithm for WSNs. Further simulations show that our framework is feasible and has high accuracy in preventing compromised nodes from being a cluster head. In the future work, we will examine the scalability of our model through comprehensive simulations and try to integrate other trust models into our framework.

Acknowledgments

This work is partially supported by the National Natural Science Foundation of China (NSFC) (no. 61103198) and the Engineering Science Program of Communication University of China (no. 3132013XNG1319).

References

[1] F.-Y. Ren, H.-N. Huang, and C. Lin, "Wireless sensor networks," Journal of Software, vol. 14, no. 7, pp. 1282–1291, 2003 (Chinese).
[2] J. Hurt, Y. Lee, H. Yoont, D. Choi, and S. Jin, "Trust evaluation model for wireless sensor networks," in Proceedings of the 7th International Conference on Advanced Communication Technology (ICACT ’05), pp. 491–496, Phoenix Park, Republic of Korea, February 2005.
[3] Q. Jing, L. Y. Tang, and Z. Chen, "Trust management in wireless sensor networks," Journal of Software, vol. 19, no. 7, pp. 1716–1730, 2008.
[4] J. Hur, Y. Lee, S. Hong et al., "Trust-based secure aggregation wireless sensor networks," in Proceedings of the 3rd International Conference on Computing, Communications and Control Technologies, vol. 3, pp. 1–6, 2005.
[5] G. V. Crosby, N. Pissinou, and J. Gadze, "A framework for trust-based cluster head election in wireless sensor networks," in Proceedings of the 2nd IEEE Workshop on Dependability and Security in Sensor Networks and Systems (DSSNS ’06), pp. 13–22, IEEE Computer Society, Columbia, Md, USA, April 2006.
[6] Y. L. Sun, W. Yu, Z. Han, and K. J. R. Liu, "Information theoretic framework of trust modeling and evaluation for ad hoc networks," IEEE Journal on Selected Areas in Communications, vol. 24, no. 2, pp. 305–317, 2006.
[7] F. G. Mármol and G. M. Pérez, "Providing trusting wireless sensor networks using a bio-inspired technique," Telecommunication Systems Journal, vol. 46, no. 2, pp. 163–180, 2011.
[8] A. Aljanaby, K. R. Mahamud, and N. M. Norwawi, "Interacted multiple ant colonies optimization approach to enhance the performance of ant colony optimization algorithms," Computer and Information Science, vol. 3, no. 1, 2010.
[9] F. G. Mármol and G. M. Pérez, "TRMSim-WSN, trust and reputation models simulator for wireless sensor networks," in Proceedings of the IEEE International Conference on Communications, Communication and Information Systems Security Symposium, June 2009.
[10] T. K. Kim and H. S. Seo, "A Trust Model using fuzzy logic in wireless sensor networks," Proceedings of World Academy of Science, Engineering and Technology, vol. 32, 2008.
[11] B. Shen, S.-Y. Zhang, and Y.-P. Zhong, "Cluster-based routing protocols for wireless sensor networks," Journal of Software, vol. 17, no. 7, pp. 1588–1600, 2006.
