Information Sciences xxx (2012) xxx–xxx

Contents lists available at SciVerse ScienceDirect

Information Sciences

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

Target tracking and boundary node selection algorithms of wireless sensor networks for internet services

Prasan Kumar Sahoo [a,*], Jang-Ping Sheu [b], Kun-Ying Hsieh [c]

[a] Department of Computer Science and Information Engineering, Chang Gung University, Kwei-Shan 33302, Taiwan
[b] Department of Computer Science, National Tsing Hua University, Hsinchu 30013, Taiwan
[c] Department of Computer Science and Information Engineering, National Central University, Chung-Li 32054, Taiwan

ARTICLE INFO

Article history:
Available online xxxx

Keywords:
Wireless sensor network
Ubiquitous network
Boundary node
Target tracking

ABSTRACT

Wireless sensor network (WSN) is an integral part of Internet of Things (IoT), in which sensors can be used to keep track with interesting targets under surveillance. Target tracking is one of the important research issues, where sensors are deployed in many applications such as campus security, surveillance, habitat and battle field monitoring. Information can be forwarded in an ad hoc multi-hop fashion via internet to monitor a specific region and can form a ubiquitous network for several internet services. In this paper, Sequential Boundary Node Selection (SBNS) and Distributed Boundary Node Selection (DBNS) algorithms are proposed to find out the boundary nodes of the wireless sensor network. Besides, a target tracking protocol is proposed to detect the entry and exit of the targets using those boundary nodes. Simulation results show that the selection of boundary nodes in our protocol is almost close to the optimal one and the time of selecting boundary nodes would not increase rapidly, with increase in the size of the deployed nodes.

1. Introduction

Wireless sensor network (WSN) is important for a number of strategic applications such as coordinated target detection, surveillance, and localization. Progress in miniaturization has allowed researchers to build networked sensors, increasingly compact devices that combine the functionality of sensors, radios, and processors. Their low cost and wireless communication capability makes it feasible to deploy them in large numbers, and without infrastructure. These sensor nodes are equipped with sensing, communicating, and data processing units, which allow sensor nodes to collect, exchange, and process information about the environments to detect the target. Several works on target detection and tracking [9,16] are found in recent years, which can be classified into four different categories. The first category is to find out the trajectory of the target. In [6], authors focus on finding the trajectory of the target via the detected data. They use the time information of entering or leaving of a target through the sensing range of the sensors to draw trajectory of the interesting target.

The second category, as described in [20] is to wake up the sensors by using predictive strategy in order to keep track with the target, when it moves into their sensing ranges. The third category [19] is to use the predictive strategy to reduce the transmitted data between the sink and each sensor node. The last category is to obtain more accurate information of the target. The authors in [8,21] have proposed a tree based structure that uses a lot of sensors to collaborate the detection mechanism and to collect precise data. However, above methods always need a lot of sensors to collaborate, and waste the

* Corresponding author. Tel.: +886 3 2118800; fax: +886 3 2118700.
E-mail address: pksahoo@mail.cgu.edu.tw (P.K. Sahoo).

0020-0255/$ - see front matter © 2012 Published by Elsevier Inc.
http://dx.doi.org/10.1016/j.ins.2012.07.034

resources, if a target moves to certain area iteratively. Recently, boundary node selection algorithms are proposed in [10,11], which can select boundary nodes if any coverage hole exists in the network and therefore cannot be used for target tracking. Moreover, the proposed methods heavily depend on the simulation work, instead of formal algorithms and theoretical analysis. It is to be noticed that in some applications may only need to record the information of a target entering or leaving a boundary of the specific regions. For example, zoologists want to know the wildlife migration or habitual behavior such as duration of a target that stays in the monitoring regions or when it enters into or exits from the region.

There are some related literatures about finding the boundary of the specific regions, which can be roughly classified into three categories: geometric, statistical and topological methods. The approaches in geometric methods relay on all the nodes know their geographical locations. The authors in [3] propose a simple and distributed algorithm to find the boundary of the hole by using the right-hand rule to mark the boundary of the holes. Unlike geometric method, the statistical method usually assumes the probability distribution, such as uniform distribution, of sensor deployment and without having location information. Base on these assumptions, the main idea of these related algorithms is applying some unique statistical properties that under certain network conditions they can probabilistically identify the boundary nodes. The authors in [2] found a characteristic can be used to identify the boundary node and define some corresponding thresholds for each node to determine whether it is a boundary node. In topological method, the nodes also assume not knowing location information and only use topological properties such as connectivity to identify the boundary nodes. The authors in [5] model the impact of sensor density on the accuracy of the position estimation in managing the sensor network for the target tracking. However, they do not consider the boundary nodes to track the target.

Collaborative [14] event detection and target tracking algorithms are proposed for the heterogeneous wireless sensor networks to find the presence of targets. Though, the authors consider border nodes to detect the target, the number of nodes used for the target detection is high. Energy-efficient tracking algorithms [4,16,17] for the wireless sensor networks are proposed to accomplish the goal of target tracking. However, authors propose the power saving and routing mechanisms to minimize the energy consumption and to track the targets simultaneously. A novel distributed algorithm [18] is proposed that correctly detects the nodes on the boundaries and connects them into meaningful boundary. The authors in [12] have developed a boundary recognition algorithm without location using only local knowledge information. The authors use geometric constructions, called patterns, to recognize the inner nodes of the network and consider all other nodes to be part of the outer boundary or the boundary of a hole. The authors in [1] classify several military tracking system such as GPS based, RF based, and camera based. Though, they compare those tracking systems in terms of power consumption, cost, efficiency and so on, there is no new tracking mechanism proposed in the paper.

It is to be noted that wireless sensors are battery powered and therefore are energy constrained. As they are deployed in the harsh terrains, it is difficult to replace or recharge them. Therefore, we propose the power efficient boundary node selection algorithms to track the entry and exit of the targets and main contributions of our work can be summarized as follows:

- We propose three different types of boundary node selection algorithms, which can select boundary nodes either in distributed or centralized manner.
- We propose centralized, distributed and sequential boundary node selection algorithms in the same paper, which is not seen in any other work. The proposed algorithms can provide a comparative study between centralized and distributed protocols.
- Since, main goals of WSN is to track the targets, we propose also a target tracking protocol that can use few selected boundary nodes to check the entry and exit of a target.
- Unlike other existing target tracking protocols, limited number of boundary nodes are involved in our algorithm to detect a target, and therefore other nodes can go to the power saving mode. Hence, our proposed algorithms can give the complete solution of event monitoring in WSN and can improve the network lifetime.
- The proposed boundary selection method is based on the theoretical analysis of termination conditions, number of packets and boundary node selection time, which is unique.

Remainder of the paper is organized as follows. System model of our proposed protocols is presented in Section 2. Our boundary node selection protocols are described in Section 3 and target tracking protocol is given in Section 4 of the paper. Performance evaluation of our algorithms are done in Section 5 and concluding remarks are made in Section 6.

2. System model

It is assumed that the sensors are deployed densely over a rectangular monitoring region such that no sensor is left as unconnected. Besides, the whole monitoring region is fully covered and sink is placed along the boundary of the monitoring region. Though the sensing range of each node varies, communication range of the sensors is fixed. In other words, sensing range of a node may be larger or smaller than its communication range, which is fixed. Each node knows its location information via GPS or through some positioning methods [13,15] and has a unique ID for its identification.

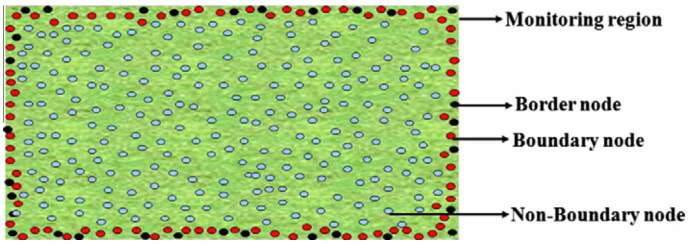


Fig. 1. Example of border, boundary and non-boundary nodes.

2.1. Definitions

Definition 1 (Connected nodes). Two nodes i and j are said to be connected, if their Euclidean distance $d_{ij} \leqslant R_c$, where R_c is communication range of those nodes. Throughout this paper, communication range of each node is fixed.

Definition 2 (*Sensing overlapping*). Sensing range of two nodes i and j is said to be overlapping, if sensing range of node i $\left(R_s^i\right)$ and sensing range of node $j\left(R_s^j\right)$ is not disjoint, i.e. $R_s^i \bigcap R_s^j \neq \Phi$. In this paper, sensing range (R_s) of a node is variable, i.e. $R_s \leqslant R_c$ or $R_s \geqslant R_c$.

Definition 3 (One-hop neighbor). A node A is said to be one-hop neighbor of node B, if their Euclidean distance $d(AB) \leqslant R_c$.

Definition 4 (Border Nodes (BoNs)). Border Node (BoN) is the set of nodes whose sensing range either touches or intersects the border of the monitoring region.Mathematically, let $Ax + By + C$ be the equation of the border of the rectangular monitoring region, and (x_1,y_1) be the location of a node within the monitoring region such that $d = \frac{Ax_1+By_1+C}{\sqrt{A^2+B^2}}$ be the perpendicular distance between the node with border of the monitoring region. Then, BoN = $\{\chi/d \text{ of } \chi \leqslant R_s\}$. As shown in Fig. 1, the red and black color nodes are the border nodes.[1]

Definition 5 (Extreme nodes). Any node A, which is located at (x,y) is said to be an extreme node among its one-hop neighbors, if it has either maximum or minimum value in its x or y or in both coordinates as compared to coordinates of its one-hop neighbors.

Definition 6 (Boundary Node (BN)). A border node that satisfies our boundary node selection algorithms as given in Section 3 is called a boundary node (BN). It is to be noted that set of boundary nodes $\subseteq$ of set of border nodes.

As shown in Fig. 1, only the red color nodes are the boundary nodes. A boundary is generated by the selected *Boundary Nodes (BNs)* and are linked together to form a loop, which are responsible for detecting the entry or exit of the target to or from the monitoring region.

Definition 7 (Non-Boundary Node (NBN)). A node, which is neither a border nor a boundary node is called a Non-Boundary Node (NBN). As shown in Fig. 1, nodes with blue color are the non-boundary nodes.

3. Boundary node selection protocols

In this section, we propose the Sequential Boundary Node Selection (SBNS), Distributed Boundary Node Selection (DBNS) and Centralized Boundary Node Selection (CBNS) algorithms to select the boundary nodes among all deployed nodes. It is to be noted that our SBNS algorithm can find the boundary nodes along the border area of the monitoring region in a sequential fashion, whereas the Distributed Boundary Node Selection (DBNS) algorithm can select the boundary nodes in a distributed manner. However, the Centralized Boundary Node Selection (CBNS) algorithm is used to analyze and compare with the performance of SBNS and DBNS algorithms.

3.1. Sequential Boundary Node Selection (SBNS) algorithm

SBNS algorithm is used to select the Boundary Nodes (*BNs*) among the border nodes of the monitoring region. According to the definition, sink must be a border node and therefore is assigned as an initiator of SBNS procedure along the border area of the monitoring region. It is assumed that location of the sink is taken to be (x_0, y_0). Being the initiator, the sink sets itself as

[1] For interpretation of color in Figs. 1–6, 8, 9, 11–18 the reader is referred to the web version of this article.

a *BN*, and broadcasts a *Request_Info* packet to its one-hop neighbors, which contains the location information, sensing range and ID of the sink and waits for the response. Upon receiving that packet, each of its one-hop neighbors responds with a *Reply_Info* packet, which includes their location information, sensing range and ID. Once, the sink receives its one-hop neighbor's information, it compares its location with location of those neighbors. Since, the sink (x_0, y_0) is located along the border of the monitoring region, it may have maximum or minimum value in its x and y coordinates. It is assumed that each node has a turning line to find a *BN* among its neighbors, in which length of the turning line is equal to the communication range of that node. For example, as shown in Fig. 2a, the turning line is assumed to be horizontal on the right or left side of the sink, if it has maximum $(x_0 = X_{max})$ or minimum $(x_0 = X_{min})$ value in its x-coordinate, respectively or vertical on its top or bottom side of the sink, if it has maximum $(y_0 = Y_{max})$ or minimum $(y_0 = Y_{min})$ value in its y-coordinate.

Based on the information from the received *Reply_Info* packets, the sink uses the right-hand rule [2] and rotates the turning line along clockwise direction. Then the first node whose center intersects with the turning line is selected as the next *BN*. The unique ID of the selected *BN* is included in the *Ack_Info* packet and is broadcast by the sink. Upon receiving the *Ack_Info* packet, the selected *BN* replies the *Confirm_New_BN_Ack* packet back to the previous *BN* (i.e. the sink) for ensuring that the selected *BN* is informed. The *Confirm_New_BN_Ack* packet contains location information, sensing range and unique ID of the previous *BN* of the selected *BN* for removing the redundant *BN*. Next, each *BN* that wants to select a new *BN* assumes a turning line through the line connecting to the previous *BN* and itself, as shown in Fig. 2b. For example, if *BN A* wants to select a new *BN*, it assumes a turning line through the line connecting to *BN A* and the sink, which is the previous *BN* of *BN A*. The formal algorithm of SBNS procedure is given in Table 1 and an example of selecting *BN* is shown in Fig. 2b. Since, the sink has maximum value in its x-coordinate $(x_0 = Xmax)$ as compared to its one-hop neighbors, it assumes that there is a turning line on its right hand side and rotates it using right hand rule. Thus, node *A* is selected as the first *BN* by the sink, which selects node *B* as the next *BN* and *BN B* subsequently selects node *C* as the next *BN*. This procedure is repeated until the starting node (sink) is revisited.

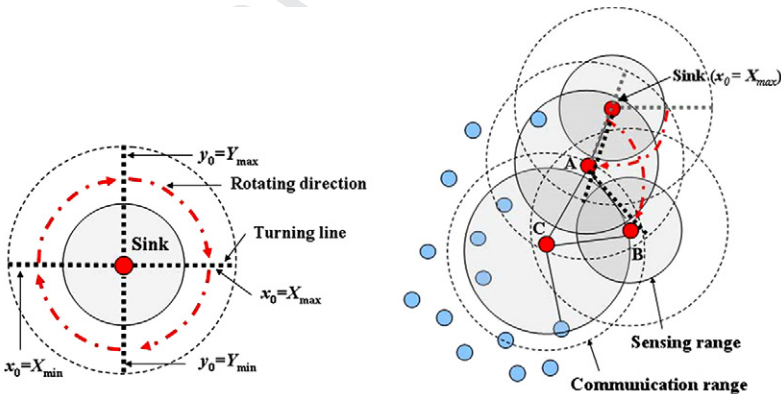
However, as shown in Fig. 2b, since sensing range of *BNs A* and *C* is overlapping with each other and *BN A* can directly communicate with *BN C*, *BNs A* and *C* can form the boundary without *BN B*. In other words, *BNs A* and *C* can still play the role of *BNs*, whereas *BN B* sets itself as a *Non-BN*. Thus, the nodes initially selected as *BNs* can switch their role to *Non-BNs*, thereby reducing the number of *BNs*. Before selecting the next *BN*, each selected *BN* (such as *BN C*), first checks its previous *BN* (such as *BN B*) applying the previous criteria based on the information of the *Confirm_New_BN_Ack* packet received from its previous *BN*. If its previous *BN* should be a *Non-BN*, the selected *BN* broadcasts that information to convey the related *BNs* (such as *BNs A* and *B*). On the other hand, to maintain the integrity among the *BNs*, each *BN* periodically sends a beacon packet to its previous *BN*. If any *BN* cannot receive the beacon packet from its related *BNs*, it executes the *SBNS* algorithm to select a new *BN* among the existing neighbors.

3.2. Distributed Boundary Nodes Selection (DBNS) algorithm

Normally, the Distributed Boundary Node Selection (DBNS) algorithm is used to select the boundary nodes along the border of the monitoring region in a distributed way. It has three phases: initial phase, selection phase, and pruning phase as described below.

3.2.1. Initial phase

Prior to this phase, it is assumed that sensors are deployed densely over the monitoring region. After deployment of nodes over the monitoring region, the sink broadcasts a *BN_Start* packet to the entire network with its location information, sensing



(a) The turning line of the sink. (b) The selection procedure of BN.

Fig. 2. Illustration of SBNS protocol.

Q1 Please cite this article in press as: P.K. Sahoo et al., Target tracking and boundary node selection algorithms of wireless sensor networks for internet services, Inform. Sci. (2012), http://dx.doi.org/10.1016/j.ins.2012.07.034

Table 1
Sequential Boundary Node Selection (SBNS) algorithm.

Algorithm 1: *Boundary node selection algorithm in SBNS*

Notation:
1. S: Sink;
2. A_i: Receiver node i, $\forall i = 1, 2, 3, \ldots, n$;
3. BN_i: Boundary node i;
4. NBN_i: Non-Boundary node i;
5. $Loc_i = (x_i, y_i)$: Location of node i;
6. L_i: Turning line of node i;
7. R_s^i: Sensing range of node i;
8. R_c: Communication range of a node;

Sequential Boundary Node Selection()
1. **while:** S is not visited
2. **do:** {
3. S sets itself as BN_i;
4. S broadcasts *Request_Info* message;
5. A_i unicasts *Reply_Info* message to S;
6. S compares Loc_S with Loc_{A_i};
7. S rotates its L_S;
8. **if** (Loc_{A_i} lies on L_S)
9. Set $BN_i \leftarrow A_i$;
10. A_i rotates its L_{A_i};
11. **if** (Loc_{A_j} lies on L_{A_i})
12. Set $BN_j \leftarrow A_j$;
13. A_j rotates its L_{A_j};
14. **if** (Loc_{A_k} lies on L_{A_j})
15. Set $BN_k \leftarrow A_k$;
16. **if** ($d(A_i, A_j) \leqslant R_c$ AND $R_s^i \bigcap R_s^j \neq \Phi$)
17. Update: $BN_k \leftarrow A_k$;
18. $NBN_j \leftarrow A_j$;
19. }
20. **End of while loop**

range and ID. Upon receiving the BN_Start packet, each node includes the same information and rebroadcasts it to their one-hop neighbors. Then each node determines whether it is an extreme node or not, if it has maximum or minimum value in its x or y or both coordinates as compared to its one-hop neighbors. Those extreme nodes declare themselves as BNs. Thus, each sensor node in the monitoring region could be classified into BN or Non-BN after the initial phase is executed.

3.2.2. Selection phase

In the initial phase, though we select few *BNs*, they may not be enough to form a complete boundary. This is because, some *BNs*' sensing range may not overlap with others' or they cannot communicate directly with each other, though their sensing range overlaps. Hence, we propose the selection phase, in which each *BN* collaborates with its neighbors to check the presence of a *BN* along its left and right side. Then, it selects a new *BNs* out of the *Non-BNs*, when one or both of these two *BNs* are absent. To achieve this, each *BN* broadcasts a *BN_Msg* packet to its one-hop neighbors, which contains the location information, sensing range and unique ID of the sink and waits for the response. However, there is possibility that the *BNs* and *Non-BNs* might have received zero to multiple number of *BN_Msg* packets from other *BNs*. It is to be noted that each *BN* can receive two *BN_Msg* packets, as maximum two other *BNs* can exist along its left and right hand side. Therefore, if a *BN* receives the *BN_Msg* packet, it selects the *BN* as a new *BN* along its left or right hand side unless it has already two *BNs* in those sides. However, based on the number of messages received by a *Non-BN*, we design the formal algorithm for DBNS as given in Table 2.

Q3 As given in Table 2, all possible cases of DBNS algorithm can be explained with examples as follows (see Tables 3 and 4).

Table 2
Distributed Boundary Nodes Selection (DBNS) algorithm.

Algorithm 2: *Boundary node selection algorithm in DBNS*

Notation:
1. S: Sink;
2. X: Any sensor node;
3. A_i: One-hop neighbor of a node;
4. BN_i: Boundary node i, $\forall i = 1, 2, 3, \ldots, n$;
5. NBN_i: Non-Boundary node i, $\forall i = 1, 2, 3, \ldots, n$;
6. BN Boundary node;

(*continued on next page*)

7. d_{ij} = Euclidean distance between BN_i and BN_j, for $i \neq j$;
8. R_c: Communication range of a node;
9. R_s^i: Sensing range of node i;
10. $Loc_i = (x_i, y_i)$: Location of node i;
11. L_i: Turning line of node i;
12. *BN_Msg*: Message that contains location,
sensing range and ID of a Boundary node;
13. *NBN_Msg*: Message that contains location,
sensing range and ID of a Non-Boundary node;
14. N: Number of messages received from the Boundary nodes;

Distributed Boundary Node Selection()
1. Set $BN_i \leftarrow X$;
2. Set $NBN_i \leftarrow A_i$;
3. **while:** X is not revisited
4. **do:** {
5. X broadcasts *BN_Msg* to its one-hop neighbors;
6. N = Number of *BN_Msg* received by A_i;
7. **switch** (N)
8. {
9. **Case** 1://If number of *BN_Msg* received by A_i is 1;
10. {
11. A_i unicasts a *NBN_Msg* to sender BN_i;
12. BN_i verifies whether it has another two BN_j
along its left and right side or not;
13. **if** (BN_i has two BNs, i.e. BN_j and BN_k)
14. Ignores *NBN_Msg*;
15. **else**
16. {
17. BN_i rotates its turning line L_i clockwise;
18. Selects NBN_i as a new BN (BN_k);
19.}
20. **Break**;
21.}
22. **Case** 2://If number of *BN_Msg* received by A_i is 2;
23. {
24. A_i verifies BN_Msg_i and BN_Msg_j;
25. **if** ($d(BN_i, BN_j) \leqslant R_c$ AND $R_s^i \bigcap R_s^j \neq \Phi$)
26. {
27. NBN_i does not change its role;
28. $BN \leftarrow BN_i$;
29. $BN \leftarrow BN_j$;
30.}
31. **else if**
32. ($d(BN_i, BN_j) \leqslant R_c$ AND $R_s^i \bigcap R_s^j = \Phi$)
33. NBN_i sets itself as a BN_i;
34. **else if**
35. ($d(BN_i, BN_j) > R_c$ AND $R_s^i \bigcap R_s^j \neq \Phi$)
36. {
37. NBN_i unicasts BN_Msg_i to BN_i and BN_j;
38. NBN_i sets itself as a BN_i;
39.}
40. **else**
41. NBN_i sets itself as a BN_i;
42. **Break**;
43.}
44. **Case** $N > 2$://If number of *BN_Msg* received by A_i is more than 2;
45. {
46. NBN_i selects one pair of BN_i corresponding to any pair of *BN_Msg*;
47. **Go to Case** 2;
48. **Break**;
49.}
50. **Default**
51. NBN_i goes to power saving mode;
52. **Break**;
53.}
54.}
55. **End of while loop**

Case 1: If a *Non-BN* cannot receive any *BN_Msg* packet from the *BNs* within certain predefined time, the *Non-BN* goes to power saving mode. This type of situation may happen for the nodes located within the central area of the monitoring region.

Q1 Please cite this article in press as: P.K. Sahoo et al., Target tracking and boundary node selection algorithms of wireless sensor networks for internet services, Inform. Sci. (2012), http://dx.doi.org/10.1016/j.ins.2012.07.034

Table 3
Execution of pruning phase in DBNS.

Algorithm 3: *Pruning phase in DBNS*

Notation:
1. BN_i: Boundary node i, $\forall i = 1, 2, 3, \ldots, n$;
2. BN_j: Boundary node that is left neighbor of BN_i;
3. BN_j: Boundary node that is right neighbor of BN_i;
4. NBN_i: Non-boundary node i;
5. R_s^i: Sensing range of node i;
6. *Mark_BN Msg*: Message that contains ID of a node;
7. *Go_Non − BN Msg*: Acknowledges others to change status to NBN_i;
8. $Mark\_BN_i$: Temporary status of a node;

Pruning()
1. BN_i checks $R_s^i \bigcap R_s^j$
2. **if** $(R_s^i \bigcap R_s^j \neq \Phi)$
3. {
4. $Mark\_BN_i \leftarrow BN_i$;
5. BN_i broadcasts *Mark_BN Msg*;
6. **if** $(R_s^{Mark\_BN_i} \geqslant R_s^{Mark\_BN_j})$
7. {
8. $Mark\_BN_i$ unicasts *Go_Non − BN Msg* to $Mark\_BN_j$;
9. $NBN_j \leftarrow Mark\_BN_j$;
10.}
11. **else**
12. $NBN_i \leftarrow Mark\_BN_i$;
13.}
14. **else**
15. Pruning phase is terminated;

Case 2: If a *Non-BN* receives the *BN_Msg* packets from two different *BNs*, the *Non-BN* chooses itself either to remain as a *Non-BN* or to be a new *BN* or becomes a forwarding node between those two *BNs*. These situations can be described in different sub cases as follows:

(1) If those two *BNs* can communicate and their sensing range overlaps with each other, the *Non-BN* does not change its role. As shown in Fig. 3a, the *Non-BN C* still remains as a *Non-BN*.

(2) If two *BNs* are unable to communicate, as well as their sensing range does not overlap with each other, however the *Non-BN's* sensing range overlaps with both of those *BNs'* sensing range, the *Non-BN* directly sets itself as a new *BN*. It is to be noted that if more than one *Non-BN* satisfies this condition, those *Non-BNs* exchange their location information with their one-hop neighbors and the *Non-BN* having shortest vertical distance between its position and the virtual line (as shown in Fig. 3b, $\overline{AB}$ is the virtual line), connecting to both *BNs* sets itself as the new *BN*. For example, as shown in Fig. 3b, the *Non-BN D* has the shortest vertical distance than *C* and *E* with the virtual line connecting to both *BNs A* and *B*. Hence, *Non-BN D* becomes a new *BN* and broadcasts a *BN_Msg* packet to inform about its role to its neighbors.

(3) If those two *BNs* can communicate with each other without overlapping their sensing range and that *Non-BN's* sensing range can overlap with both of those *BNs*, then that *Non-BN* sets itself as a new *BN* and broadcasts a *BN_Msg* packet to its neighbors. If more than one *Non-BN* satisfies the above condition, the procedure introduced in subcase 2 is applied. As shown in Fig. 4a, *Non-BN C* becomes the new *BN*, as it has shortest vertical distance from the virtual line as compared to other *Non-BNs* and can connect to both *BNs A* and *B*.

(4) If two *BNs* cannot communicate with each other; whereas their sensing range overlaps with each other, the *Non-BN* having shortest distance from the virtual line joining those two *BNs*, broadcasts a *Forwarding_Node_Info* packet to inform them to be their forwarding node to exchange their message. To avoid the redundancy among forwarding nodes, the *Non-BNs*, which receives the *Forwarding_Node_Info* packet from another *Non-BN* does not transmit the same packet again. As shown in Fig. 4b, *BNs A* and *B* cannot communicate, though their sensing range overlaps with each other. Hence, the *Non-BN C* becomes a forwarding node between *A* and *B*.

Case 3: If a *Non-BN* receives more than two *BN_Msg* packets from the *BNs*, the *Non-BN* considers each pair of its nearby *BNs* to decide its own role. For example, as shown in Fig. 5a, *Non-BN D* considers the pair of *BNs* (*A*, *B*) and (*B*, *C*). For selecting one pair of *BNs* out of each pairs, the procedure of **Case 2** is applied to the *Non-BN*. After the *Non-BN* determines its new role for its two neighboring *BNs*, it can become a *BN* or forwarding node or still can remain as a *Non-BN*. It is to be noted that the *Non-BN* may play several roles simultaneously, such as it is selected as a forwarding node and *BN* by different pairs. In Fig. 5a, the *Non-BN D* receives three *BN_Msg* packets from *BNs A*, *B*, and *C*. *Non-BN D* considers its role among each pair of nearby *BNs A* and *B*, *BNs B* and *C*,

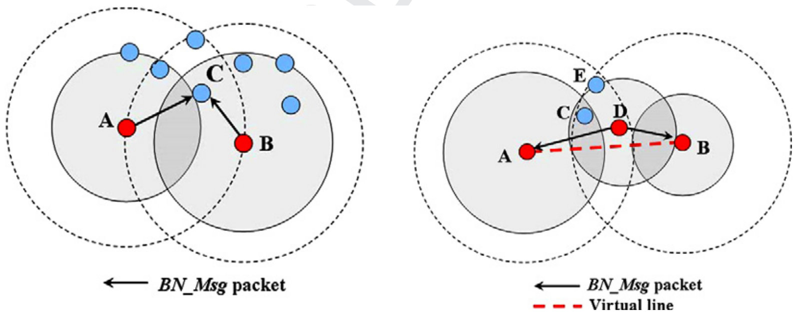
Q1 Please cite this article in press as: P.K. Sahoo et al., Target tracking and boundary node selection algorithms of wireless sensor networks for internet services, Inform. Sci. (2012), http://dx.doi.org/10.1016/j.ins.2012.07.034

Table 4
Centralized Boundary Node Selection (CBNS) algorithm.

Algorithm 4: *Boundary node selection algorithm in CBNS*

Notation:
1. S: Sink;
2. A_i: Receiver node i, $\forall i = 1, 2, 3, \ldots, n$;
3. A_j: Neighbor of A_i, $\forall i, j = 1, 2, 3, \ldots, n$;
4. t: Random waiting delay duration; 5. BN_i: Boundary node i;
6. NBN_i: Non-Boundary node i;
7. $Loc_i = (x_i, y_i)$: Location of node i;
8. L_i: Turning line of node i;
9. R_s^i: Sensing range of node i;
10. R_c: Communication range of a node;

Centralized Boundary Node Selection()
1. **while:** S is not visited
2. **do:** {
4. S broadcasts *Collect_Info* message to all nodes;
5. Upon receiving *Collect_Info* message, A_i waits for t units;
6. A_i rebroadcasts *Collect_Info_Ack* message to A_j;
7. A_j waits for t units;
8. Rebroadcasts *Collect_Info_Ack* message to S;
9. S compares Loc_S with Loc_{A_j};
10. S rotates its L_S;
11. **if** (Loc_{A_j} lies on L_S)
12. Set $BN_j \leftarrow A_j$;
13. A_j rotates its L_{A_j};
14. **if** (Loc_{A_i} lies on L_{A_j})
15. Set $BN_i \leftarrow A_i$;
16. A_i rotates its L_{A_i};
17. **if** (Loc_{A_i} lies on L_{A_k})
18. Set $BN_k \leftarrow A_k$;
19. **if** ($d(A_j, A_k) \leqslant R_c$ AND $R_s^j \bigcap R_s^k \neq \Phi$)
20. Update: $BN_k \leftarrow A_k$;
21. $NBN_i \leftarrow A_i$;
22. S checks its neighboring NBN_i;
23. if ($d(S, NBN_i) \leqslant R_c$ AND $R_s^S \bigcap R_s^i \neq \Phi$)
24. S is replaced by NBN_i;
25. else
26. S checks its next selected BN_j;
27. if ($d(BN_j, NBN_i) \leqslant R_c$ AND $R_j^S \bigcap R_s^i \neq \Phi$)
28. BN_j is replaced by NBN_i;
29.}
30. **End of while loop**



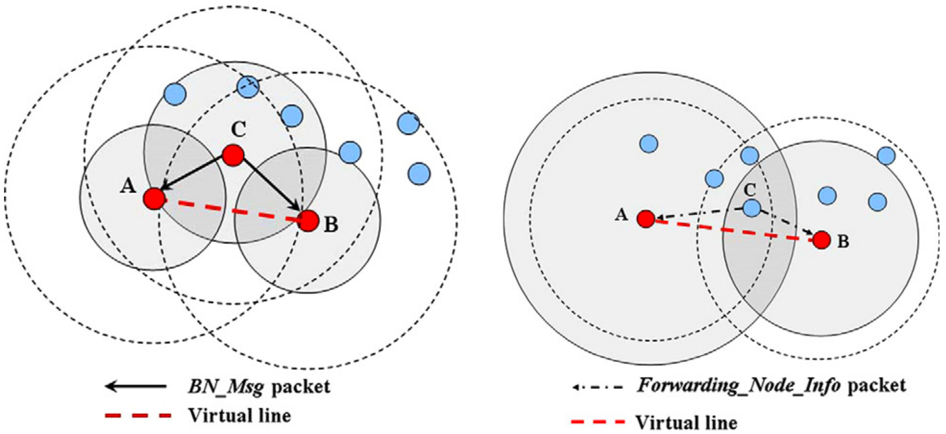
(a) *Non-BN C* still remains as a *Non-BN*.

(b) *Non-BN D* is selected to be a new *BN* to connect *BNs A* and *B*.

Fig. 3. Example pertaining to **Case 1** of selection phase.

respectively. Since, *BNs A* and *B* can communicate directly with each other and their sensing ranges are overlapping, none of the *Non-BN* including *Non-BN D* is considered as a new *BN* or a forwarding node. However, in case of *BNs B* and *C*, since their sensing range overlaps and they cannot communicate with each other, one of

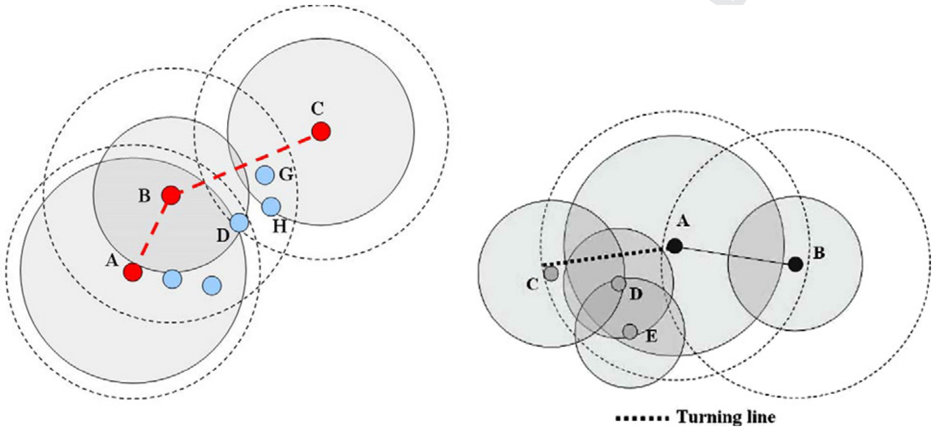
Q1 Please cite this article in press as: P.K. Sahoo et al., Target tracking and boundary node selection algorithms of wireless sensor networks for internet services, Inform. Sci. (2012), http://dx.doi.org/10.1016/j.ins.2012.07.034



(a) *Non-BN C* is selected to be a new *BN* to connect *BNs A* and *B*.

(b) *Non-BN C* is selected as a forwarding node for *BNs A* and *B*.

Fig. 4. Example pertaining to **Case 2** of selection phase.



(a) Example of *BN* and forwarding node selection for multiple *BN_Msg* packets.

(b) The Non-BN C has the maximum value will be a new BN.

Fig. 5. Example pertaining to Cases 3 and 4 of selection phase.

the *Non-BNs* among *D*, *H* and *G* will be a forwarding node. Hence, *Non-BN G* has the shortest vertical distance to the virtual line $\overline{BC}$. Finally, the *Non-BNs G* is selected as the forwarding node instead of node *D*, which does not change its role.

Case 4: If any of the *Non-BNs* receives the *BN_Msg* packet from one of the *BNs*, the *Non-BN* responds to it with a *Non_BN_ID* packet that contains its ID, sensing range and location information.

Upon receiving one or multiple *Non_BN_ID* packets from the *Non-BNs*, the *BN* checks the current number of *BNs* with whom it's sensing range overlaps. If any *BN* has already two *BNs* in its left and right hand side, whose sensing range overlaps with it and are connected to it, then it ignores that *Non_BN_ID* packet and the corresponding sender. If that *BN* finds only one *BN* with whom it's sensing range overlaps, it has to find out another one *BN*, whose sensing range overlaps with it in its left or right hand side. According to the information of the received *Non_BN_ID* packets, the *BN* rotates the turning line clockwise to select a new *BN* along its left or right hand side. The *BN* selects one *Non-BN* as a new *BN* among those *Non-BN* senders that comes first during rotation of the turning line and whose center intersects with it. Finally, *BN* sends a *BN_Info* packet to inform the selected new *BN* and waits for the acknowledgment. Here, the *BN_Info* packet contains the new *BN's* the location information, sensing range, and ID.

For example, as shown in Fig. 5b, *BN A* has one *BN B* along its right hand side. Since, it needs another *BN*, it broadcasts a *BN_Msg* packet to its one-hop neighbors and *Non-BNs C*, *D*, and *E* receive that *BN_Msg* packet and respond to it by sending *Non_BN_ID* packets. Though, the sensing range of *Non-BNs C*, *D*, and *E* overlaps with *BN A*, the *Non-BN C* is selected as the new *BN* of *BN A*, as its center intersects with the turning line and comes first as compared to *Non-BNs D* and *E*. The same procedure is applied to find out other two *BNs*, if the *BN* has no other *BNs* in its left and right hand side.

3.2.3. Pruning phase

After the initial and selection phases of DBNS are executed, enough *BNs* are selected to form a complete boundary. However, it could be possible that redundant *BNs* may exist after the selection of boundary nodes, which are needed to be pruned. Those redundant *BNs* are reset to *Non-BNs* to select the least number of *BNs* for saving energy. Even if, a *BN* has only two connected *BNs* in its left and right hand side, it could be possible that the *BN* is a redundant one. For example, as shown in Fig. 6, *BN A* selects the new *BN B* and *BN D* selects the new *BN C*. *BN B* receives the *BN_Msg* packets from *BN A* and *C*. Since, *BN A's* sensing range overlaps with *BN C*, *BN B* thinks it is a redundant *BN* and can change its role to a *Non-BN*. Similarly, *BN C* also thinks it is a redundant *BN* as *BN D's* sensing range overlaps with *BN B* and then it switches to a *Non-BN* as well. Therefore, the network is partitioned. In order to solve the this problem, we propose that each redundant *BN* should temporarily set itself as a *Mark-BN* before returning to a *Non-BN* and broadcasts *Mark_BN*(*ID*) packet to all of its neighboring *BNs* with its ID. Upon receiving the *Mark_BN* packet, *BN* checks whether this *Mark-BN* can be reset to the *Non-BN* or not.

If it is possible, the *BN* sends *Go_NonBN*(*ID*) packet with the *Mark-BN's* ID. If the *Mark-BN* receives all *Go_NonBN*(*ID*) packets from its neighboring *BNs*, then the *BN* resets to the *Non-BN* and broadcasts a *Re_NonBN(ID)* packet with its ID to declare its return as the *Non-BN*. For example, in Fig. 6, *BN B* sets itself as a *Mark-BN* and broadcasts the *Mark_BN*(*B*) packet to its neighboring *BNs A* and *C*. Upon receiving the *Mark_BN*(*B*) packet by *BN A* and *C*, each of them checks whether it is a *Mark-BN* or not. Obviously, *BN C* is also a *Mark-BN*, but its sensing range is larger than the *Mark-BN B*. Hence, *BN C* sends *Go_NonBN*(*C*) packet back to the *Mark-BN B* to allow *Mark-BN B* to change its role to a *Non-BN* node. Since, *BN A* is not a *Mark-BN*, it directly sends the *Go_NonBN*(*A*) packet back to the *Mark-BN B*. When *Mark_BN B* receives the *Go_NonBN*(*ID*) packets from all of its neighboring *BNs A* and *C*, *Mark_BN B* returns to the *Non-BN* and broadcasts a *Re_NonBN*(*B*) packet to declare its return as the *Non-BN*. In this case, *Non-BN B* becomes a forwarding node. Eventually, the DBNS algorithm can form a complete boundary dynamically after removing the redundant nodes.

3.3. Centralized Boundary Node Selection (CBNS) algorithm

In this section, a Centralized Boundary Node Selection algorithm (CBNS) for comparing the performance with SBNS and DBNS algorithms is proposed. At first, the sink broadcasts a *Collect_Info* packet to the entire network. Upon receiving the packet, each node rebroadcasts the same packet to its neighbors and then waits for a random time to send a *Collect_Info_Ack* packet back to the sink. Each node helps other nodes to forward their *Collect_Info_Ack* packet. The random waiting time of each node may be calculated from its unique ID $\times$ user defined time interval.

In our simulation, the user defined time interval is assumed to be 0.1 seconds and the delay time of the node with unique ID 35 and 36 are 35×0.1 is 3.5 and 3.6 s, respectively. If the time interval is long enough, each time only one node sends a *Collect_Info_Ack* packet. Although, several same *Collect_Info_Ack* packets may collide during forwarding, it is possible that at least one *Collect_Info_Ack* packet can reach at the sink. Therefore, the sink has high probability to collect the complete information of the entire network.

Next, the sink applies the same SBNS algorithm to select the initial *BNs* along the border area of the monitoring region. The information of those initial *BNs* is sequentially recorded into one table of the sink in order. Since, some *BN's* neighboring nodes (*Non-BNs*) may have larger sensing range than those *BN's* that can replace some of them. Therefore, the number of selected *BNs* may be reduced. For example, as shown in Fig. 7, the CBNS, first applies the SBNS algorithm to select six *BNs* in sequence of A → B → C → D → E → F. However, CBNS can select only four *BNs* in sequence of *BNs* A → G → H → F, which have larger sensing range. Accordingly, after finishing the SBNS algorithm, the CBNS starts to prune the unnecessary *BNs* by

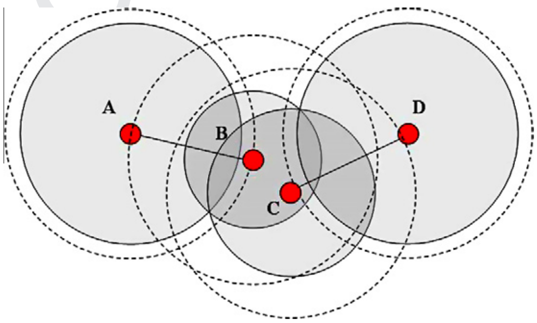


Fig. 6. Pruning the redundant *BN B*.

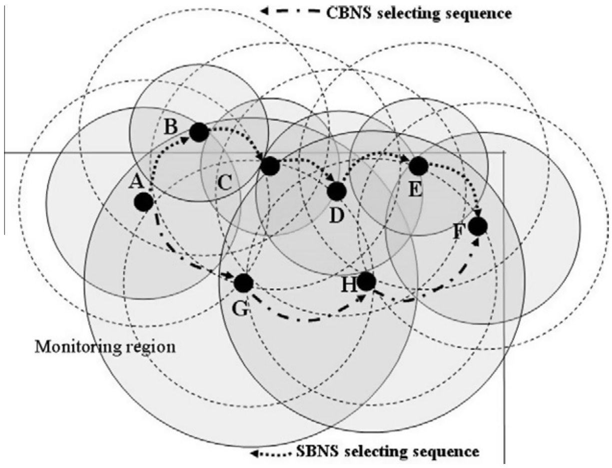


Fig. 7. Illustration of selecting new BN.

selecting fewer new *BNs* to replace initial *BNs*. Firstly, sink checks the first *BN* itself in the table and verifies if there exists a *Non-BNs* whose sensing range overlaps with it and can communicate with each other.

If there is no such *Non-BNs*, the sink checks the next selected *BN* in the table. To the contrary, the sink checks whether those *Non-BNs*' sensing range can overlap with it and can communicate with each other. Besides, the sink also checks whether those *Non-BNs*' sensing range can overlap with any one *BN* of the following initial *BNs* and can communicate with each other. For example, in Fig. 7, *BN A's* neighboring *Non-BN G*, whose sensing range can overlap with the following *BN D* and can communicate with each other. Then, *BN A* selects the new *BN G* to replace the two initial *BNs B* and *C*, and the sink removes *BNs B* and *C* from the table and inserts the new *BN G* to the table. If there exists more than one *Non-BNs* that satisfy the previous conditions, the *Non-BN* that can replace the most initial *BNs* and has the largest sensing range is selected. Next, the sink checks the next *BN*, new *BN G*, and continue the same procedure until it is revisited. Eventually, the CBNS selects fewest number of *BNs* to enclose the monitoring region.

3.4. Theoretical analysis

It is to be noted that whether it is SBNS, DBNS or CBNS, initially messages are exchanged among the nodes to select the boundary nodes. Upon receiving a control message, each node rebroadcasts it to its one-hop neighbors to select the boundary nodes as described in the above subsections. The selection mechanism is continued and is terminated as soon as the initiator (sink) is revisited. Based on the selection procedure of boundary nodes, we theoretically analyze the termination of boundary node selection mechanism in general. Let us assume that N number of sensors are deployed over the rectangular monitoring region and node i first initiates the boundary node selection procedure. Since, value of N is not known to a node, it does not know if all nodes of the network have already participated to select the boundary nodes and therefore, cannot terminate the selection procedure. However, each node terminates the algorithm only after sending and receiving boundary node selection related control packets to all its neighbors.

Let, nodes i and j discover each other as neighbors by time t by exchanging *BN_Msg* with their location information, ID and sensing range with a probability of p. Then, $p = 1/N$. Then probability of successful transmission p_s by node i can be given by Eq. (1).

$$p_s = \frac{1}{N}\left(1 - \frac{1}{N}\right)^{N-1}, \quad \text{for } 1 \leqslant i \leqslant N \tag{1}$$

Let, each node remains in receiving mode for an exponentially distributed time interval with mean $1/v$ and let γ be the transmitting duration between two nodes i and j. Then the value of v that maximizes the boundary node selection among neighbors can be given by Eq. (2).

$$v = \frac{1}{2\gamma N} \tag{2}$$

Considering the inter-transmission time among nodes is exponentially distributed, total traffic from N nodes can constitute a Poisson process with rate Nv. Hence, a transmission from a node at time instant t is successful only if no other transmission starts during $[t - \gamma, t + \gamma]$ and therefore the probability of a successful transmission can be re-derived as given in Eq. (3).

$$p_s = e^{-2N\gamma v} \tag{3}$$

Let, n be the number of rounds required to find the required number of boundary nodes among N nodes of the network, where $0 < n \leqslant N-1$ and Γ_n be the duration of nth round to select the boundary nodes that starts when nth node is selected and ends when $(n+1)$th node is selected. Thus, in the nth round, there will be still $(N-n)$ nodes to be discovered, each of which has a successful probability of p_s. The control packet transmission from these $(N-n)$ nodes can produce the Poisson process with rate $(N \times n)v$, with each of them having probability p_s. Thus, the total boundary node selection time of the network (Γ) can be given as stated in Eq. (4).

$$\Gamma = \sum_{n=1}^{N} \Gamma_n = \frac{2\gamma N}{(N-n)e^{-2N\gamma v}} \tag{4}$$

Taking X_n be the number of boundary nodes selected in n rounds, the termination condition used by any node i is that it stops at the end of nth round. Under such condition, the termination process can be achieved as given in Eq. (5).

$$X_{n-1} \geqslant 2^{n-2} \text{ and } X_n < 2^{n-1}, \quad \text{where } n \geqslant 2 \tag{5}$$

Suppose m number of control messages are exchanged among neighbors of any node i and κ is the mean number of neighbors of each node. Hence, the number of control packets M_j exchanged between a node i with its one-hop neighbors in each round can be calculated as $M_j = m \times \kappa$. The total number of control message can be calculated as given in Eq. (6).

$$M = \sum_{j=1}^{N} M_j = \sum_{j=1}^{N} m \times \kappa \tag{6}$$

4. Target tracking protocol

In this section, we propose the target tracking protocol, which detects the entry or exit time of a target over the monitoring region by utilizing the selected boundary nodes, as described in Section 3. As shown in Fig. 9a, let the bold curved line represents the outer boundary of all *BN's* sensing range. As soon as the target passes through the outer boundary, the time of entry or exit as detected by the *BN* is transmitted to the sink. Accordingly, two types of time stamps are used to record the entry and exit of a target. The time stamp T_e is used to record the entry of the target, whereas time stamp T_l is used if the target leaves the boundary region.

4.1. Coverage overlapping algorithm

As per the assumptions of our protocols, the monitoring region is fully covered. Besides, in our boundary node selection algorithms, it is verified that *BNs* must be connected with their neighboring *Non-BNs*. Since, the monitoring region is fully covered, circumference of each *BN's* sensing range within the monitoring region must be covered by neighboring *BNs'* and *Non-BNs'* sensing range. For example, as shown in Fig. 8, the *BN A's* circumference is covered by two *BNs'* (the dark gray curves) and one *Non-BN's* (the light gray curves) sensing range. Therefore, we always can find the nearby *Non-BNs* of each

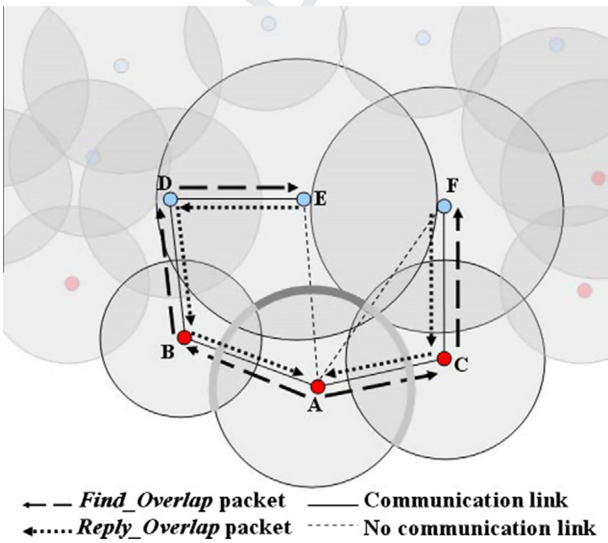


Fig. 8. Example of finding sensing overlapping.

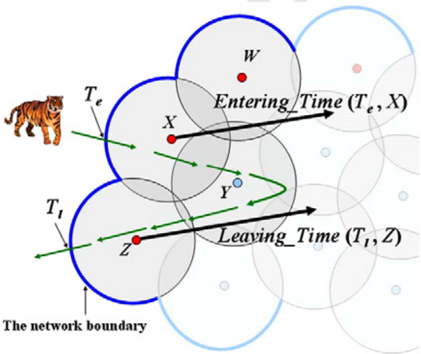
BN. First, the *BN A* broadcasts a *Find_Overlap* packet containing its location information and sensing range S_A to the neighbors as shown in Fig. 8. The *Non-BNs D*, *E*, and *F* whose distance from the *BN A* is less than or equal to ($S_A + S_{MAX}$) can ensure that its sensing range overlaps with the *BN A*, where S_{MAX} is the maximum sensing range among all the deployed nodes. Upon receiving the *Find_Overlap* packet, only *Non-BNs E* and *F*, who satisfy the above conditions rebroadcast the packet and transmit the *Reply_Overlap* packet with its location and sensing range back to the *BN A*. The *BN A* selects the closest *Non-BN E* and repeats the procedure until its circumference of sensing range within the monitoring region is fully covered. If the distance between the *BN* and some *Non-BNs* are same, the *Non-BN* having the largest sensing range is selected. Eventually, *BN A* selects *Non-BN E* to exchange the time stamp information when the target is passed through the routing path *A*, *B*, *D* and *E*.

4.2. Target tracking algorithm

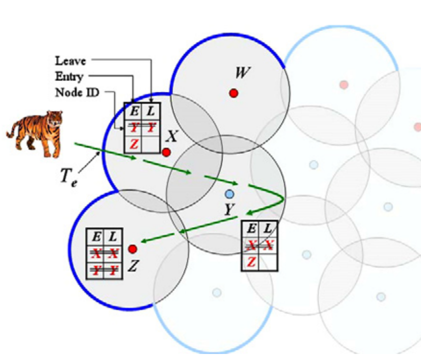
In this phase, we design algorithm to determine the entry or exit of the target through the monitoring region, which collaborates with the *BNs* and *Non-BNs* whose sensing range overlap with each other. When a target enters to the monitoring region, the *BNs* can detect the target, and then broadcasts the *Detect*(*ID*) packet to their neighbors, which contains its unique ID. Similarly, when the exit of the target is detected, the concerned *BNs* broadcast the *Leave*(*ID*) packet with its unique ID to their neighbors. It is assumed that each node maintains a record table to record the received *Detect*(*ID*) or *Leave*(*ID*) packets. The record table is composed of two fields. One is the field of entry (*E*) and another is the field of leave (*L*), which records the received *Detect*(*ID*) or *Leave*(*ID*) packets, separately. If a sensor node detects the target, it checks its field of entry is empty or not. If it is empty, the sensor node determines the entry of the target and broadcasts the *Entering_Time*(T_e, *ID*) packet to inform the sink and its neighboring nodes or it determines the target is within the network and then broadcasts the *Detect*(*ID*) packet.

Upon receiving the *Entering_Time*(T_e, *ID*) or *Detect*(*ID*) packet, each sensor node records received node's ID to the entry field of the record table. Similarly, if a sensor node has detected the target and again detects the target, it checks the entry field is empty or not. If it is not empty, the sensor node knows the target has left its sensing range and is still within the monitoring region. Then, the sensor node broadcasts the *Leave*(*ID*) packet to inform its neighboring nodes. To the contrary, the sensor node determines the target has left the monitoring region and then broadcasts the *Leaving_Time*(T_l, *ID*) packet to inform the sink and its neighboring nodes. Upon receiving the *Leaving_Time*(T_l, *ID*) or *Detect*(*ID*) packet, each sensor node records the received node's ID to the exit field of the record table. Since each sensor node has the location information of the sink, it can utilize the geographic routing protocol [7] to transmit the *Entering_Time* and *Leaving_Time* packets to the sink. Beside, if a sensor node receives the *Detect*(*ID*) and *Leave*(*ID*) packets from the same sensor node, it removes the node's ID of these two packets from its record table, as the target is no longer within the sensor node's sensing range.

For example, as shown in Fig. 9b, when the tiger enters into the monitoring region, the *BN X*, first detects the target at time T_e. Next, it checks and finds its recording table is empty, and then sends the *Entering_Time*(T_e,*X*) to the sink. After the target leaves the *BN X's* sensing range, it broadcasts the *Leave*(*X*) packet and checks its recording table again. The time during *BN X* broadcasts the *Leave*(*X*) packet to its neighbors; *Non-BN Y* has already sent the *Detect*(*Y*) packet to the *BN X*. Hence, *BN X* finds a non-empty field in its recording table and therefore does not transmit the *Leaving_Time*(T_l,*X*) to the sink. Later, when the target turns back and leaves the monitoring region, *BN Z* receives the *Detect*(*Y*) packet from the *Non-BN Y*, and finds its recording table is non-empty. Hence, when the *BN Z* detects the target, it does not transmit the *Entering_Time*(T_e, *Z*) to the sink. Prior to the target leaves the *BN Z's* sensing range, it must have received the *Leave*(*Y*) packet from the *Non-BN Y*. Since, the *Detect*(*Y*) packet and Leave(Y) packet coexist in the *BN Z's* recording table, it removes them form its recording table. After the target leaves *BN Z's* sensing range, it broadcasts the *Leave*(*Z*) packet to its neighbors, and finds its recording table empty. Therefore, the *BN Z* transmits the *Leaving_Time*(T_l,*Z*) to the sink, as it is the last sensor that detects the target leaving.



(a) Procedure of tracking a target.



(b) An example of record table.

Fig. 9. Illustration of the target tracking algorithm.

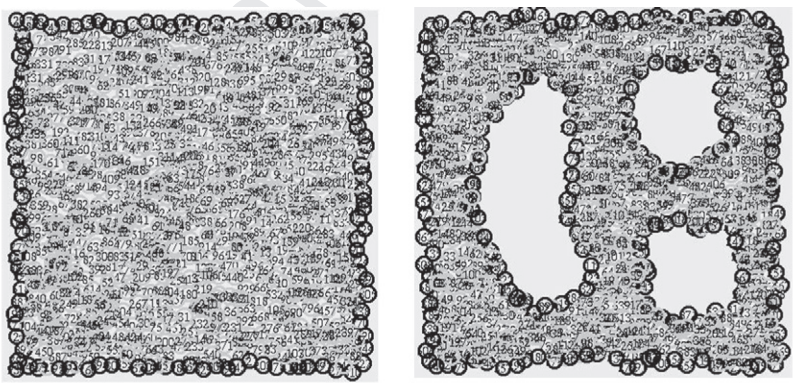
5. Performance evaluation

To evaluate performance of the proposed boundary node selection algorithms, all three algorithms are implemented in NS-2, version 2.31 on Linux platform. In simulation, the nodes are randomly deployed over a monitoring region of different size from 100 m × 100 m to 500 m × 500 m. The number of deployed nodes varies from 250 to 2000. Communication range for each sensor node is fixed at 20 m, though the sensing range of each node varies from 10 m to 30 m. The detail list of simulation parameters are given in Table 5. In Figs. 11, 13, 15 and 17, we fix the monitoring region as 100 m × 100 m with variable number of nodes. On the contrary, we fix the number of nodes as 1000, but the size of the monitoring region has varied as shown in Figs. 12, 14, 16 and 18. Then, the CBNS, SBNS and DBNS algorithms are implemented for different node numbers and area of the deployed region and are compared with each other. In the CBNS, sink knows locations of all nodes in the network. It can find the nodes on the border area of the monitoring region and chooses the nodes with larger sensing range as the *BNs*. The performance metrics such as the *number of boundary nodes* is defined as the number of selected *BNs* to enclose the monitoring region, the *number of control packet overheads* is defined as the number of control packets to find and select the *BNs*, the *selection time of boundary nodes* is defined as the total time of finding the entire *BNs* and the *remaining energy per node* is defined as the average remaining energy of each node when execution of algorithms is finished. Fig. 10a and b represent the simulated constructed boundary using DBNS algorithm, when 1000 nodes are deployed randomly over the monitoring region of size 500 m × 500 m with and without holes. These two figures are directly captured from the screen snapshot of the network animator, where the black and gray circles represent the finally selected *BNs* and *Non-BNs* separately. This demonstrates that our DBNS algorithm can correctly selects the *BNs* to enclose various shapes of the monitoring region and holes. Figs. 11 and 12 show the number of boundary nodes that are selected by the CBNS, SBNS, and DBNS algorithms. Here, SBNS and DBNS select more *BNs* than CBNS. This is because CBNS has the complete information of entire nodes within the network, which helps CBNS to determine the optimal *BNs*. Besides, unlike CBNS, SBNS and DBNS have to follow the right hand rule to select the first node whose sensing range intersects with the turning line as the *BN*. CBNS can select *BNs* with greater sensing range even if *BNs* are not located at the outside of the monitoring region. Therefore, the CBNS can select fewer *BNs* with larger sensing range to enclose the monitoring region than SBNS.

As shown in Fig. 11, as mentioned above, CBNS selects the least number of *BNs*. Besides, for CBNS, the simulation result also demonstrates that if the number of nodes in the network is increased, the number of selected boundary nodes is decreased. This is because increasing in the number of nodes increases the probability of selecting the nodes with greater

Table 5
List of parameters used in the simulation.

Simulation parameters	Initial values
Number of nodes	250–2000
Shape of monitoring region	Square
Size of monitoring region	100 m × 100 m ~500 m × 500 m
Communication range (Cr)	20 m
Sensing range (Sr)	10–30 m
Initial energy	100 J
Receive power	0.38 J
Transmit power	0.35 J



(a) The monitoring region without any holes.   (b) The monitoring region with three holes.

Fig. 10. Simulated boundary nodes using DBNS algorithm.

Q1 Please cite this article in press as: P.K. Sahoo et al., Target tracking and boundary node selection algorithms of wireless sensor networks for internet services, Inform. Sci. (2012), http://dx.doi.org/10.1016/j.ins.2012.07.034

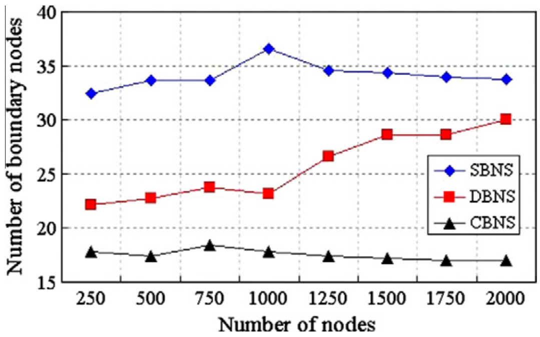


Fig. 11. Number of selected boundary nodes for different number of nodes.

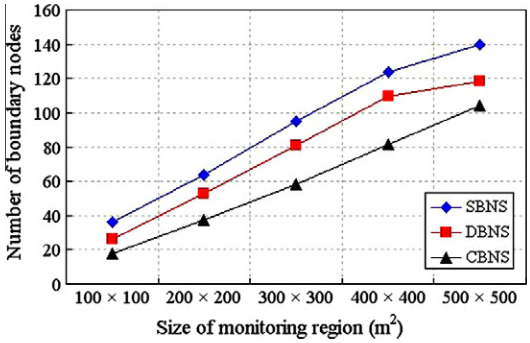


Fig. 12. Number of selected boundary nodes for different size of monitoring region.

sensing range, and therefore the number of boundary nodes is decreased. Hence, the monitoring region can be enclosed by fewer boundary nodes with greater sensing ranges. On the other hand, comparing the SBNS with DBNS, we find that DBNS has better performance over SBNS, as it can select few forwarding nodes to replace some *BNs* to exchange message between two *BNs*. Hence, DBNS usually selects less number of *BNs* than SBNS. With the increase in number of nodes, for CBNS, the number of *BNs* is still similar, since it has the complete information of every node that always can select the least number of *BNs*. However, for SBNS and DBNS, the number of *BNs* always increases with the number of nodes. This is because the node density at border area increases with the number of nodes in the network. High node density usually results in high probability of packet collision and data loss in exchanging the information. The incomplete information collection from *BNs'* neighboring nodes may cause to select incorrect or improper *BNs*. These incorrect or improper selected *BNs* again selects the same or redundant *BNs*. Besides, the distances among the nodes decrease with increase in the number of nodes at the border area. Thus, the *BNs* have high probability to select the next new *BN* with shorter distance among them. Therefore, the *BNs* have to select more *BNs* to form the boundary.

As shown in Fig. 12, obviously, the number of *BNs* is increasing with the size of the monitoring region. In high node density environment such as 100 m $\times$ 100 m, SBNS and DBNS select more 102% and 48% of number of *BNs* than CBNS individually due to high probability of packet collision. On the contrary, in low density environment such as 500 m $\times$ 500 m, SBNS and DBNS only select more 34.2% and 13.8% of number of *BNs* than CBNS separately. Although CBNS has the least number of boundary nodes, its control packet overhead is quite high, since each node in CBNS has to transmit its own information to the centralized node such as the sink. This needs to transmit lots of control packets to the sink, whereas only few nodes are involved in the procedures of SBNS and DBNS. Thus, as shown in Figs. 13 and 14, SBNS and DBNS have smaller control packets overhead than the CBNS. In addition, as DBNS has to broadcast the *BN_Start* packets to the entire network, the control packets overhead of DBNS will be higher than SBNS. In Fig. 11, it is reasonable that control packets overhead of three algorithms is in proportion to the number of nodes within the network. For CBNS, the control packets overhead for 2000 nodes is 64.74 times higher than that of 250 nodes. Besides, the control packets overhead of CBNS is from 31.87 to 313.89 times for SBNS and from 32.39 to 312.3 times for DBNS. This shows that the control packet overhead of CBNS in dense network is higher than in sparse network due to higher packet collision because of higher number of packets retransmission. In Fig. 14, the control packets overhead reduces with the increase in size of the monitoring region as the node density gradually decreases. The number of control packets of CBNS, DBNS and SBNS in 500 m $\times$ 500 m is only about 66%, 78% and 42% in 100 m $\times$ 100 m. Here, the SBNS has the largest decline in percentage of number of control packets as the node density at the border area becomes low and only few nodes join the procedure of SBNS.

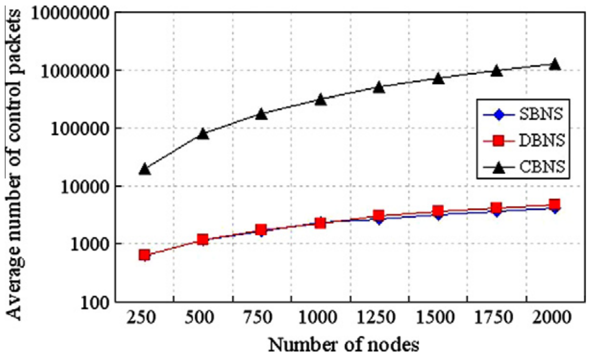


Fig. 13. Number of control packets for selecting boundary nodes with different number of nodes.

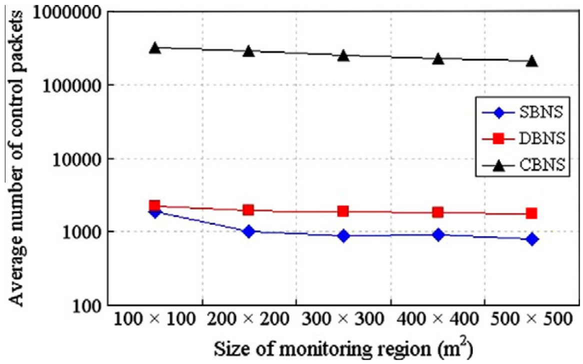


Fig. 14. Number of control packets for selecting boundary nodes for different size of monitoring region.

The selection time of boundary nodes in our simulation is the duration of the first packet that is transmitted until to the last *BN* is selected, and the result is shown in Fig. 10a and b. In CBNS, it is observed that the computation time is much less than the transmission time and we can ignore it. However, the sink has to spend lots of time for waiting the information from each node of the network. That is why the CBNS spends much more selection time of boundary nodes than the SBNS and DBNS. In SBNS, it sequentially selects the boundary nodes one by one and only one selected *BN* can select another one. However, in DBNS, the selected *BNs* can select other *BNs*, simultaneously as the algorithm is distributed. Therefore, DBNS algorithm takes less time to find the boundary than the SBNS.

In Fig. 15, the selection time of boundary nodes is proportional to the number of nodes. Unlike CBNS, in SBNS and DBNS, the selection time of *BNs* does not increase rapidly with the number of nodes. Note that, the selection time of *BNs* of DBNS is 52.78 times of CBNS for 2000 nodes. This shows that the distributed algorithm is more efficient than the centralized one. As shown in Fig. 16, the size of the monitoring region does not affect a lot on the selection time of boundary nodes, especially for the distributed algorithm such as DBNS. Hence, distributed algorithm can be simultaneously executed, which can substantially reduce the execution time. On the other hand, for SBNS, it has to spend much more time in selecting *BNs* to form the

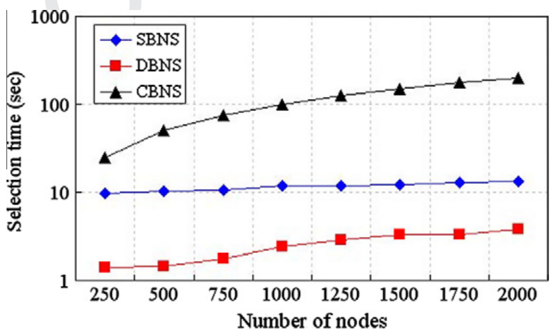


Fig. 15. Boundary nodes selection time for different number of nodes.

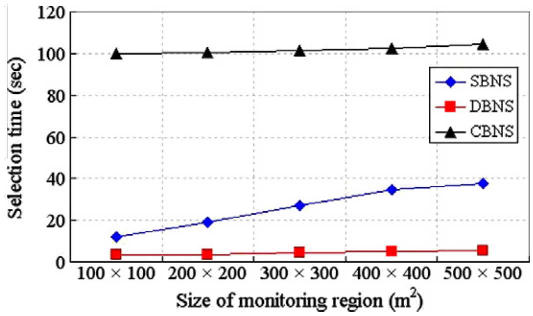


Fig. 16. Boundary nodes selection time for different sizes of monitoring region.

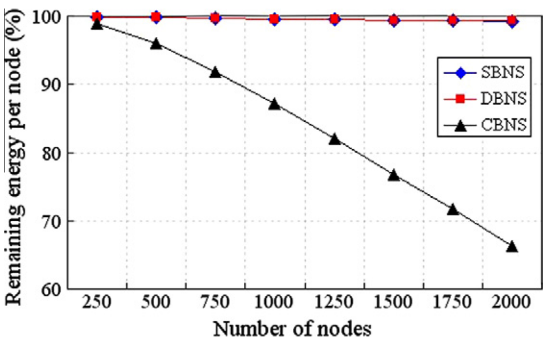


Fig. 17. Remaining energy per node for different number of nodes.

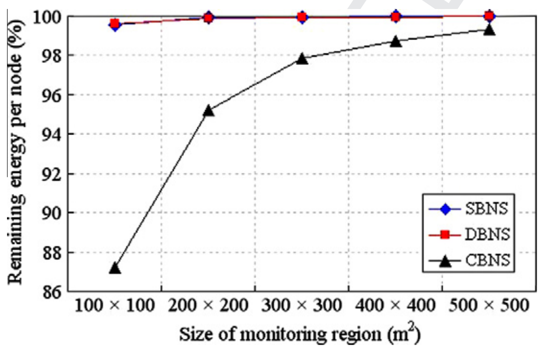


Fig. 18. Remaining energy per node for different sizes of monitoring region.

boundary due to larger circumference of the monitoring region. In CBNS, although the node density decreases with size of the monitoring region, the low probability of packet collision should reduce the selection time of the boundary nodes. However, the distance between the sink and each node becomes larger in lower node density than in higher node density case. The longer distance takes more time to transmit packets form each node to the sink. In summary, our DBNS is more efficient in time than the SBNS and CBNS for a larger monitoring region.

In Figs. 17 and 18, since DBNS and SBNS transmit fewer control packets than CBNS; their average remaining energy per node is relatively higher than CBNS. In Fig. 17, when the number of nodes is 2000 in CBNS, they more frequently send and receive packets due to higher density of nodes that substantially drops the average remaining energy of each node to 66.19%. On the other hand, for CBNS in Fig. 18, we can find the energy costs per node of CBNS in 100 m $\times$ 100 m (12.81%) is 2.67 times than in 200 m $\times$ 200 m (4.79%). However, in Fig. 14, the number of control packets in 100 m $\times$ 100 m (320,971 packets) only increases more 12.68% (36,142 packets) than in 200 m $\times$ 200 m (284,829 packets). Obviously, the incremental ratios between the number of control packets and energy costs per node are not in equal proportion. The main reason why the nodes in 200 m $\times$ 200 m can save much more energy than in 100 m $\times$ 100 m is the probability of the nodes receiving packets from other unconcerned nodes in the dense network is higher than in spare one. Note that, in Table 5, the energy cost of receiving power is higher than the transmitting power. Therefore, the nodes in dense network waste much more energy for receiving unnecessary packets than in spare network and cause the substantially decrease in the remaining energy per node from 95.21% (in 200 m $\times$ 200 m) to 87.19% (in 100 m $\times$ 100 m).

6. Conclusion

In this paper, a Sequential Boundary Node Selection algorithm (SBNS), Distributed Boundary Node Selection algorithm (DBNS) and Centralized Boundary Node Selection (CBNS) algorithm for the WSNs are proposed to find the boundary nodes of a monitoring region. Besides, a target tracking protocol is proposed using those boundary nodes to know the entry and exit of the targets through the monitoring region. The simulation results show that the DBNS algorithm has similar performance with the SBNS in selecting the boundary nodes. However, the communication overhead of DBNS is lower than the SBNS due to the information exchange among fewer neighboring nodes in DBNS. In addition, with increase in network size, the boundary node selection procedure in DBNS is much faster than the SBNS.

Acknowledgments

We are grateful to the Editor and anonymous reviewers for their constructive comments on an earlier version of this paper. This research was supported in part by the National Science Council of Taiwan, Republic of China, under Grant Nos. NSC-100-2221-E-182-069, NSC-101-2923-E-182-001-MY3 and CGURP-100-2221-E-182-027.

References

[1] T. Alhmiedat, A.A. Taleb, M. Bsoul, A study on threats detection and tracking systems for military applications using WSNs, International Journal of Computer Applications 40 (15) (2012) 12–18.
[2] K. Bi, K. Tu, N. Gu, W. Dong, X. Liu, Topological hole detection in sensor networks with cooperative neighbors, in: Proceedings of International Conference on Systems and Networks Communications, 2006.
[3] Q. Fang, J. Gao, L.J. Guibas, Locating and bypassing routing holes in sensor networks, Mobile Networks and Applications 11 (2) (2006) 187–200.
[4] M. Guo, E. Olule, G. Wang, S. Guo, Designing energy efficient target tracking protocol with quality monitoring in wireless sensor networks, Journal of Supercomputing 51 (2) (2010) 131–148.
[5] K. Hadi, C.M. Krishna, Management of target-tracking sensor networks, International Journal of Sensor Networks 8 (2) (2010) 109–121.
[6] W. Kim, K. Mechitov, J.Y. Choi, S. Ham, On target tracking with binary proximity sensors, in: Proceedings of ACM/IEEE International Symposium on Information Processing in Sensor Networks, 2005, pp. 301–308.
[7] W.H. Liao, Y.C. Tseng, K.L. Lo, J.P. Sheu, Geo GRID: a geocasting protocol for mobile ad hoc networks based on GRID, Journal of Internet Technology 1 (2) (2000) 23–32.
[8] C.Y. Lin, W.C. Peng, Y.C. Tseng, Efficient in-network moving object tracking in wireless sensor networks, IEEE Transactions on Mobile Computing 5 (8) (2006) 1044–1056.
[9] L. Liu, B. Hu, L. Li, Algorithms for energy efficient mobile object tracking in wireless sensor networks, Cluster Computing 13 (2) (2010) 181–197.
[10] A. Rafiei, Boundary node selection algorithms by simple geometrical properties in WSNs, in: Proceedings of Fifth Asia Modelling Symposium, 2011, pp. 221–226.
[11] A. Rafiei, M. Abolhasan, D. Franklin, F. Safaei, Boundary node selection algorithms in WSNs, in: Proceedings of IEEE 36th Conference on Local Computer Networks, 2011, pp. 251–254.
[12] O. Saukh, R. Sauter, M. Gauger, P.J. Marrón, K. Rothermel, On boundary recognition without location information in wireless sensor networks, in: Proceedings of International Symposium on Information Processing in Sensor Networks, 2008, pp. 207–218.
[13] J.P. Sheu, C.S. Hsu, J.M. Li, A distributed location estimating algorithm for wireless sensor networks, in: Proceedings of International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing, 2006, pp. 218–225.
[14] K.-P. Shih, S.-S. Wang, H.-C. Chen, P.-H. Yang, CollECT: collaborative event detection and tracking in wireless heterogeneous sensor networks, Computer Communications 31 (14) (2008) 3124–3136.
[15] K.F. Su, C.H. Ou, H.C. Jiau, Localization with mobile anchor points in wireless sensor networks, IEEE Transactions on Vehicular Technology 54 (3) (2005) 1187–1197.
[16] S.P.M. Tran, T.A. Yang, A novel target movement model and energy efficient target tracking in sensor networks, in: Proceedings of Technical Symposium on Computer Science Education, 2006, pp. 97–101.
[17] G. Wang, M.Z.A. Bhuiyan, L. Zhang, Two-level cooperative and energy-efficient tracking algorithm in wireless sensor networks, Concurrency and Computation: Practice and Experience 22 (4) (2010) 518–537.
[18] Y. Wang, J. Gao, J.S.B. Mitchell, Boundary recognition in sensor networks by topological methods, in: Proceedings of International Conference on Mobile Computing and Networking, 2006, pp. 122–133.
[19] Y. Xu, J. Winter, W.C. Lee, Dual prediction-based reporting mechanism for object tracking sensor networks, in: Proceedings of International Conference on Mobile and Ubiquitous Systems: Networking and Services, 2004, pp. 154–163.
[20] W.L. Yeow, C.K. Tham, W.C. Wong, Evaluations of target tracking in wireless sensor networks, in: Proceedings of IEEE Vehicular Technology Conference, 2005, pp. 2825–2829.
[21] W. Zhang, G. Cao, DCTC: dynamic convoy tree-based collaboration for target tracking in sensor networks, IEEE Transactions on Wireless Communications 3 (2004) 1689–1701.

Q1 Please cite this article in press as: P.K. Sahoo et al., Target tracking and boundary node selection algorithms of wireless sensor networks for internet services, Inform. Sci. (2012), http://dx.doi.org/10.1016/j.ins.2012.07.034