2D k-barrier duty-cycle scheduling for intruder detection in Wireless Sensor Networks

Jie Tian a,⇑, Wensheng Zhang b, Guiling Wang a, Xin Gao a

a Department of Computer Science, New Jersey Institute of Technology, United States
b Department of Computer Science, Iowa State University, United States

ABSTRACT

Intruder detection is an important application in Wireless Sensor Networks (WSNs). Barrier coverage, which requires much fewer sensors than full coverage, is broadly known as an appropriate model of coverage for such an application. However, previous works all focus on providing 1D barrier coverage on a thin belt region based on global network information. In this paper, we introduce the concept of 2D k-barrier, and investigate establishing 2D barriers on a square region only with sensors’ local neighbor information. To provide 2D k-barrier coverage in the field, a distributed scheme is proposed. The scheme targets on providing low detection delay and low energy consumption. The barriers constructed in both horizontal and vertical directions guarantee the detection of any intruder whose crossing path can be in any of the two directions. Extensive performance evaluation based on NS2 simulator shows that the proposed scheme outperforms existing mechanism in terms of both detection delay and energy consumption.

1. Introduction

Wireless Sensor Networks are expected to be employed in many long-term applications such as military surveillance, infrastructure protection and scientific exploration [9,12,32,8,1]. These applications require that sensor networks have a long life time. However, the limited preloaded energy in small sensor nodes and the unavailability of rechargeable energy in many situations pose as a big constraint to achieve the longevity requirement. To deal with the issue, past research proposes sensor networks operate at a power conservation mode where sensor nodes are scheduled to be active for only a short period of time and then stay dormant for a long time. Scheduling sensors in this way can prolong the network lifetime, but on the other hand, it may incur a lower sensing quality. From a spatial perspective, the scheduling of sensor nodes directly governs sensor distribution and number of active sensors, and thus the sensing quality. A small number of active nodes and inappropriate distribution of them will reduce the network’s quality of monitoring. The design of sensor networks for these applications must strive to reduce average power consumption for increasing network life time and provide desired sensing service at the same time.

Desired sensing service varies by application. This paper focuses on intruder detection, an important sensor network application [1,32,15,29]. The objective of the application in this paper is to detect intruders before they cross the monitored field, instead of having to detect them immediately. The characteristic has been leveraged to save energy and maintain a required quality of sensing service. Some previous work has been proposed to solve the similar problem. For example, Cui et al. propose a simple and effective protocol, 2D-Mesh, for intruder detection [8]. The protocol divides the whole area into grids of the same size and each grid is further divided into bands of the same length. Each time, only the band of sensors within each grid wake up and sense the area. The bands take turns to wake up and go to sleep. However, it has a limitation when sensors are not evenly distributed in the network, which happens a lot when in many situations when manual deployment is not possible and other methods, such as scattering sensors from an aircraft, have to be used. In the situations, some areas have high sensor density while others have low density, but when the grids and bands are evenly divided based on location, the sensors in a band of a dense area have large unnecessary sensing overlap while sensors in a band of a sparse area may not be able to provide enough coverage. As shown in Fig. 1, the upper band (Group i + 1) cannot fully cover the area, thus some intruders can...
find a leak in the coverage and successfully run away without being detected; the lower two bands (Group i and Group i-1) have large sensing overlap when the bands are evenly divided. The best solution is to divide sensors into four groups: Subgroup A, B, C, and D, as shown in the Fig. 1. By this division, the field can be fully covered and one more shift is generated to further save the energy. This is one objective of the paper: divide sensors into as many shifts as possible and each shift can cover one line of the field such that no intruder can go across the field without being detected.

The above problem statement is similar to the 1D barrier construction problem [15,16]. For example, Kumar et al. proposed a centralized algorithm to divide sensors into barriers, each of which can fully cover one strip of the monitored area [16]. However, they do not consider minimizing sensing overlap between adjacent sensors in a band to maximize the sensor network lifetime. In addition, to cover a deep field, 1D barrier coverage may result in a long detection delay since it may take long time for the shifts to rotate from one side of the field to the other. Therefore, we propose to construct k-barrier coverage to reduce the detection delay by a factor of k. For example, by constructing a 2-barrier coverage, we can approximately reduce the detection delay to 1/2.

Having a k-barrier coverage can catch all the intruders who enter from one side of the field and try to cross the field with a low delay. However, for intruders who parachute into the field, a 1D coverage is not enough. For example, as shown in Fig. 2, if there are only the horizontal barriers, an enemy soldier can run along the horizontal barriers without being detected. Therefore, we propose to construct 2D k-barrier to catch even the intruders entering the field from the sky.

In this paper, we design a new delicate distributed algorithm to divide sensors into shifts in both horizontal and vertical directions. Each shift can ensure sensing one line of the monitored area such that no intruders can cross the line without being detected, while at the same time, the sensors in one shift have minimum sensing overlap to conserve the energy. At any time, k shifts of sensors are on in both directions. k is a system parameter to balance the energy consumption and detection delay. The extremely fine division of sensors into barriers saves a lot of energy, but makes the scheme sensitive to sensor failure: once a sensor dies, there is a security leakage of the covered line. To deal with sensor failure, a reconstruction algorithm is designed to achieve fault-tolerance. Specifically, if one sensor node fails, other nodes can automatically run the reconstruction algorithm to regroup themselves. Extensive simulations are conducted and results show that our 2D k-barrier scheme can achieve lower detection delay and a much longer life time of network compared with previous work [8].

The remainder of the paper is organized as follows. We present the system model and assumptions in Section 2. The scheme overview is introduced in Section 3. Sections 4–6 present the construction, scheduling and reconstruction algorithm, respectively. Section 7 reports the performance evaluation results. Related work is discussed in Section 8 and conclusion is presented in Section 9.

2. System model and assumptions

In this paper, we deal with intruders who can enter the monitored field from any side of the field and who can also enter the field from the sky by parachute or other methods. The objective of the intruders is to cross the field following a crossing path. A crossing path is a path that connects one side of the region to the opposite side, which was first defined in [21]. We do not impose any constraints on intruders in terms of moving direction, moving speed, departing time, and other factors.

We introduce a concept called 2D k-barrier, achieving which is the primary objective of the paper. Before we formally define 2D k-barrier, we first introduce the definition of strong k-barrier, which was first defined in [21] and is shown as follows:

**Definition 1.** A path is said to be k-covered if it intercepts at least k distinct sensors.

**Definition 2.** A sensor network is said to be strongly k-barrier covered if:
P (any crossing path is k-covered) = 1 w.h.p.

Similar as [21], we define strong 2D k-barrier as follows:

**Definition 3.** A sensor network is said to be strongly 2D k-barrier covered if:

\[ P(\text{any horizontal crossing path is k-covered}) \land P(\text{any vertical crossing path is k-covered}) = 1 \text{ w.h.p.} \]

We make the following assumptions in the paper:

- The network is loosely synchronized. The synchronization can be achieved by many mature techniques with low overhead [38,40,7].
- All sensor nodes can calculate their relative positions to each other through many localization methods [6,22,13,35,34]. Existing algorithms can reduce the error to as well as 3% when calculate the relative positions [24].
- An isotropic sensing model is adopted [5,21,31]. The sensing area of each node is a circle with the same radius. The transmission range of a sensor, RC, is always more than twice the sensing range, RS [8].
- We assume the communication among sensors is secure, which can be easily taken care of with various protocols [27,28].

3. Scheme overview

Our scheme has two objectives: Firstly, provide 2D k-barrier coverage and ensure no intruder can cross the monitored field in both horizontal and vertical directions without being detected. Secondly, prolong the network lifetime as much as possible. To achieve the goals, k horizontal barriers and k vertical barriers of sensors work together and monitor the field at any time, such that no intruder can cross them without being detected. All other sensors can stay asleep to conserve energy. The ideal case is that all sensors are divided into barriers and the sensors in any barrier have very little sensing overlap with each other, such that as many as possible barriers can be constructed and take turns to work in the long run to conserve energy and maximize network lifetime.

There are two challenges in achieving the goals: the first one is how to divide sensors into a very fine level in a distributed manner considering the initial deployment is very likely to be uneven; the second one is when sensors die, how to reorganize the sensors and consider the initial deployment is very likely to be uneven; the second one is when sensors die, how to reorganize the sensors and consider the initial deployment is very likely to be uneven. In this step, sensors are first divided into grids. The division in both directions are independent of each other and the algorithm runs simultaneously in both directions. In either direction, sensors are divided into groups based on their locations. The groups and the corresponding sensors are further divided into grids. The information about a grid, such as the number of sensors, position of each sensor, status of each sensor, et al. is stored in a selected coordinator node of the grid and will be used for sensors in the grid’s group to self-organize into barriers. The detailed barrier construction algorithm is presented in Section 4.

After the initialization phase, a major portion of sensors are allocated into barriers and each sensor has a group ID and a barrier ID in both the horizontal direction and the vertical direction. The remaining small percentage of sensors do not belong to any barrier and will act as backup sensors when some sensors fail.

- **Initialization Phase:** in this phase, a barrier construction algorithm runs to divide sensors into barriers in both the horizontal and vertical directions. The division in both directions are independent of each other and the algorithm runs simultaneously in both directions.

- **Running Phase:** in this phase, sensors take turns to wake up and go to sleep based on their group ID and barrier ID in both horizontal direction and vertical direction. Note that each sensor has two group IDs and two barrier IDs. One in each direction. For most of the time, k barriers in each direction are on and cover 2k lines to provide 2D k-barrier coverage in the area and all the others are off. When shifting, neighbor barriers in each direction are on together for a short time to ensure no intruder missing during barrier rotations and there is enough time to reconstruct a subgroup if any of its sensors dies. The scheduling algorithm is presented in detail in Section 5.

Since the algorithm for constructing both horizontal and vertical barriers in the proposed scheme is the same, in the follow-up sections, we only use horizontal barriers’ construction, scheduling and re-construction as the example to describe the algorithms in detail.

4. K-barrier construction algorithm

There are two steps in the barrier construction algorithm. The first is to assign sensors some basic information and let sensors learn the basic information of their neighbors through Hello messages. The second is sensors self-organize into different barriers. As described above, in the following, only the process in constructing horizontal k-barrier is introduced.

4.1. Information assignment

In this step, sensors are first divided into k groups based on their locations. Suppose the width of the monitored area is w_x. Then the width of a group is w_{g} = w_x / k. The group ID goes from 1 to k. Based on the location, each sensor can easily calculate its group ID. The group division process can be shown in Fig. 3.

After the group division, groups are divided into grids. The grid size is determined by the sensing range. According to the assumptions, the sensing range of a sensor is a circle with the radius RS. The circle determines two squares. One is the Inscribed Square of the circle and the other is the Circumscribed Square of the circle. As shown in Fig. 4, when the radius of a circle is RS, the side length of the Inscribed Square is \( \sqrt{2}RS \) and the side length of the Circumscribed Square...
4.2. Self-organization

The second step is to further divide sensors into barriers, each of which acts as a shift, covering one line of the field. In each group, the Starting Node looks for a node next to it to incorporate into a barrier. The next node will do the same thing until a border node at the other side of the group is included. The process is dictated by a heuristic node connection algorithm. The algorithm has two parts. The first part determines which node should be selected as the next node, and includes two set of rules: Grid Searching Rules and Node Selection Rules. Grid Searching Rules specify from which grid(s) a next node can be selected. Node Selection Rules specify which node from the candidate grids determined by the Grid Searching Rules should be selected as the next node. The second part, Group/Barrier Determination Rules, determine the ending of a barrier and the starting of another one and dynamically decides a node’s group ID and barrier ID. This part is to prepare the information for scheduling the shifts of different barriers in the scheduling algorithm.

4.2.1. Grid Searching Rules

Without loss of generality, we assume the current node starts one connection from left to right to ease our description. The grid of the node has at most eight neighbors around it, as shown in Fig. 7. The eight positions are: right, top right, bottom right, top, bottom, top left, bottom left and left. Two situations are considered: one is all the grids are in the same group and the other is not.

When all the grids are in the same group, the searching rules are shown in Fig. 7. The number is the priority of searching. For example, there is a “1” in the right grid and the right grid has the highest priority. If there is any node in the right grid, no node in other grids will be considered. When a grid is on the border of a group as shown in Fig. 8(a), which has five neighbors, or in the corner of a group as shown in Fig. 8(b), which has three neighbors, the grid searching sequence is still the same, as labeled in Fig. 7.

When some neighbor grids are not in the same group as the center grid, the searching rule is slightly different. The situation is shown in Fig. 9. In Fig. 9(a), the center node is in group i, but its top left, top and top right grids are in group i + 1; in Fig. 9(b), the center node is in group i + 1, but its bottom left, bottom and bottom right grids are in group i. In this situation, the neighbor grids in the same group have a higher priority. The center grid first searches the neighbor grids in the same group, and then the neighbor grids in a different group. The searching priority of neighbor grids is shown in Fig. 9. For example, in Fig. 9(a), the center grid first searches the right grid, then bottom right, bottom, bottom left and left grid, the six grids in group i. If it cannot find any suitable node, it will search top right, top and top left grid, the three grids in the group i + 1.

There is an exception to the rules above. If a center node is connected from a different group, its searching rule is different. For example, as shown in Fig. 9(c), a node in group i + 1 connects the center node because it cannot find a suitable node in its own group. In this case, when the center node choose its next node, it gives higher priority to the nodes in group i + 1. This is to push the connection back to the original group. The priority is shown in Fig. 9(c).

Note that the grid searching priority is related to searching direction. When the searching direction is from left to right, the priority is specified in Figs. 8 and 9; when it is from right to left, the priority is just the opposite. The philosophy behind the priority...
4.2.2. Node Selection Rules

The next node is selected from the candidate grids determined in the previous step. There are at most two candidate grids. The coordinator in each candidate grid calculates one best candidate node based on the searching node’s information. This candidate node must have the least sensing coverage overlap with the searching node in the center grid. Specifically, this candidate node is selected according to the following two rules:

- **Rule 1:** The candidate nodes must be within the communication range of the searching node.
  \[ \|p_i - p_s\| \leq RS, \ c_i \in G \]
  \( G \) represents the node set and \( c_i \) represents a single node in the candidate grid. \( s \) represents the searching node. \( p_i \) and \( p_s \) are the positions of \( c_i \) and \( s \).

- **Rule 2:** The candidate node in one grid must have the longest horizontal distance to the searching node.
  \[ \arg\max_{c_i \in G} |x_i - x_s|, \ c_i \in G \]
  \( x_c \) and \( x_s \) are the horizontal positions of \( c_i \) and \( s \).

Adopting above two rules, at most two candidate nodes are calculated by the neighbor grids. The next step is to select one node from the two as the next connection node.

- **Rule 3:** The selected connection node must have the largest horizontal distance to the searching node.
  \[ \arg\max_{i \in C} |x_i - x_s|, \ i \in C \]
  \( C = \{c_1, c_2\} \)
  \( c_1 \) and \( c_2 \) are the candidate nodes in the candidate grids respectively.

One example is shown in Fig. 11. The candidate grids are the top right grid and bottom right grid, and there are multiple nodes in each of the grid. Based on the first two rules, the coordinator in the top right grid picks node 2 and the coordinator in the bottom right grid picks node 3 as the candidate node, and send the information to the searching node. The searching node decides which one is chosen as the connection node by checking condition 3. The distance between the search node and node 2 is longer than that between node 3 and the search node, so node 2 is selected to be the next node. One node can only be chosen as a next node once. When the connection comes to the grid again, the coordinator will select candidate nodes from other nodes, which have not been taken by any barriers.

4.2.3. Group/Barrier Determination

When a connection reaches a border node, which can cover the border of the area, a barrier is successfully established and a new barrier construction needs to be started. For example, as shown in Fig. 12, border node 1 is reached and its barrier is established. To start a new barrier, node 1 only searches border nodes above it. It finds node 2 and notifies node 2 about the new barrier.
Fig. 9. Grid Searching Rules crossing group.

Fig. 10. Zigzag shape in the grid searching.

Fig. 11. Node Selection Rules.
construction and the new searching direction. Node 2 learns that its searching direction is from right to left and then it uses corresponding grid search priority and Node Selection Rules to establish its barrier. The process is done when a border node on the left side is found and then another barrier construction process is initiated in the same way.

### Algorithm 1 Self-Organization Algorithm in Horizontal or Vertical Direction at sensor $n_i$

**Notations:**
- $loc_i$: physical position of sensor $n_i$.
- $C(n_i), C(grid_j)$: the coordinator of sensor $n_i$ or grid $j$.

1. **Upon entering Self-Organization:**
   - if $n_i$ is a Starting Node then
     - Send request $\langle n_i, loc_i \rangle$ to $C(n_i)$
   end if
   - else
     - Upon receiving request $\langle n_j, loc_j \rangle$
       - if $n_j = C(n_i)$ then
         - Run Grid Searching Rules and find $grid_m$ (and $grid_k$)
           - /* one candidate group or two*/
           - Send request $\langle n_j, loc_j \rangle$ to $C(grid_m)$ ($C(grid_k)$)
       - end if
     - else
       - Run Node Selection Rules and find node $n_k$
       - Reply response $\langle n_k, loc_k, n_j \rangle$ to $C(n_i)$
   end if
   - else if
     - Upon receiving response $\langle n_k, loc_k, n_j \rangle$
       - if $n_k = C(n_i)$ then
         - if Still waiting for another reply then
           - Cache the information
         - else
           - Run Node Selection Rules 3 and choose node $n_j$
           - response $\langle n_k, loc_k, n_j \rangle$ to $n_j$
         - end if
         - else if $n_j = n_i$ then
           - send connect $\langle n_i \rangle$ to $n_k$.
       - end if
     - end if
   else if
     - Upon receiving connect $\langle n_j \rangle$
       - if $n_j$ is a border node then
         - Run Group/Barrier Determination rules
       - Send request $\langle n_i, loc_i \rangle$ to $C(n_i)$. 
   end if

Each barrier is identified by a group ID and a barrier ID. The group ID is determined by the group IDs of its member nodes. Its member nodes may belong to one or more than one group. As shown in Fig. 1, barrier A involves only group $i = 1$, so its group ID simply is $i = 1$. On the contrary, barrier B, C, and D involve more than one group. In this case, the group ID is determined by the group which contributes the most number of nodes. For example, barrier C has one node in group $i = 1$, and three nodes in group $i$. Thus its group ID is $i$. Similarly, barrier B's group ID is $i = 1$, and barrier D's group ID is $i + 1$. The barrier ID within a group is determined by a sequence number, starting from 1. For example, within group $i = 1$, A's barrier ID is 1 and B's barrier ID is 2.

A barrier is successfully established if and only if in the end, a border node is reached. If a connection cannot reach a border node, which means it cannot find enough nodes to form a barrier, which can fully cover one line of the area, then the last searching node sends a message to release all the nodes in the unestablished connection. These released nodes will act as redundant nodes that can be borrowed by other connections.

The whole self-organization step is a back-and-forth process initiated by the Starting Node, which is relatively closest to position (0,0) of its horizontal or vertical group. There are $2 \times k$ Starting Nodes in the network. The connection process first starts from left to right or bottom to top, switches direction each time a border node on the searching direction is reached, and stops when no more node can be added. Every involved node asks for the help of its coordinator, which in turn selects the next node for it according to the rules specified above. When the next node is identified, it sends a connection message to the node. The node takes the search job afterwards and further searches its own next node. After the barrier construction, most nodes are connected and allocated to different barriers. Only a small portion cannot be included in any barrier. These nodes are labeled as redundant nodes. The redundant nodes are put into a special barrier of their original groups and will be utilized in the barrier reconstruction time when some sensors fail. Fig. 13 is an example to illustrate the horizontal connection process. The communication protocol of the barrier construction algorithm is shown in Algorithm 1.

Even in a very dense sensor network, it is possible that the network follows such a similar distribution as shown in Fig. 14, though the probability is very low. In this distribution, a Starting Node A cannot establish a barrier. When the establishment cannot proceed any more, a message is broadcasted to dismiss the sensors in the unfinished barrier. Node A learn such a situation and delegates its role of starting a connection to a border sensor in another grid in the same group. In Fig. 14, when Starting Node A knows the barrier cannot be established, it tells node B to start the connection. Node B starts the connection and constructs one barrier. Through this process, even if there exists one barrier in the group, it can be found and established.

4.3 Some discussions

In the beginning of the paper, we state several assumptions. In this section, we discuss the impact on the proposed scheme when the assumptions are not satisfied and how to deal with it. We also discuss the performance of our algorithm under some unusual conditions.

- **Communication range:** We assume the communication range $RC$ is 2 times larger than the sensing range $RS$. If the condition is satisfied, sensors within the neighbor grids can be reached by a Hello message. If the condition is not satisfied, we can increase broadcasting hop. The number of hops required is $\lceil \frac{RC}{2RS} \rceil$ to reach all the sensors in neighbor grids.

- **Node distribution:** We assume dense sensor networks. However, even in a dense network, the node distribution can be very irregular, as shown in Fig. 14. However, even if there exists a barrier, our algorithm can establish it following the algorithm described in Section 4.2.
A Starting Node in a relatively isolated area which cannot construct a barrier will delegate its role to other nodes until one barrier can be established. The only problem of an irregular distribution is after the barrier construction, there may be many redundant nodes.

- Construction time:
  The barrier construction time is the time it takes for all barriers to be constructed. When k groups’ barriers are constructed, the construction process is parallel. In each group, the construction is sequential. There is always a trade-off between the construction time and number of unused sensors. Some existing barrier construction algorithms adopt fully localized methods to establish barriers. However, more redundant sensors are left and less barriers are constructed.

5. Scheduling algorithm

The scheduling algorithm is to determine the shift of each barrier and how they turn on and off. It runs round by round. In each round, n barriers in one group take turns to wake up. The wake-up sequence is barrier 1, barrier 2, ..., barrier n – 1, barrier n, barrier n – 1, barrier 2, barrier 1. Each of the k groups only has one on-duty barrier.

The on-duty time of adjacent barriers has overlap. Suppose the on-duty time of a barrier is \( t \), the overlap time is \( \Delta t \), and the wake-up time of barrier \( i \) is \( t_i \). Then barrier \( i \) goes to sleep at \( t_1 + t \) and barrier \( i + 1 \) wakes up at \( t_1 + t - \Delta t \). The short overlap time prevents intruders from crossing the field during the shifting considering the loose time synchronization among groups. More importantly, it enables communication between adjacent groups. When some sensors fail, barrier and group reconstruction is needed and intergroup communication is a must. Because of this short overlap, the duration of a round \( t_r = 2n(t - \Delta t) + \Delta t \).

A barrier \( i \) in a group consisting of \( n \) barriers wakes up twice in a round. In the \( j \)th round, its first wake-up time and sleep time are:

\[
\begin{align*}
  t_{\text{wake}1} &= (i - 1)(t - \Delta t) + n(j - 1)t_f, \\
  t_{\text{sleep}} &= t_{\text{wake}1} + t.
\end{align*}
\]

The second wake-up time and sleep time are:

\[
\begin{align*}
  t_{\text{next}} &= (2n - i - 1)(t - \Delta t) \\
  t_{\text{wake}2} &= (i - 1)(t - \Delta t) + n(j - 1)t_f + t_{\text{next}} \\
  t_{\text{sleep}} &= t_{\text{wake}2} + t.
\end{align*}
\]

From the above formulations, one node not only can calculate its own wake-up and sleep time, but also can calculate the wake-up and sleep time of all the barriers in the same group. In terms of redundant nodes, they wake up for \( \Delta t \) time with every barrier of their group in case they are needed in reconstruction when some sensors fail. If there is no one barrier successfully constructed in one group during self-organization algorithm, this group will inform other groups in the same direction to take the sensing job. Such communication is through a different direction barrier.

If all barriers in one group are about to run out of energy, before the group’s failure, this group will tell some other groups to take its sensing job and continue to maintain a 2D k-barrier coverage for the whole network. For example, a vertical group can ask its nearest vertical group to take its job through a working horizontal barrier, which shares a grid with both vertical groups and can be their mailman.

5.1. Some discussions

In our scheduling algorithm, barriers take turns to be on and off and eventually sensors should fail at approximately the same time. People may ask why we design this way and why not let a barrier to be on until its death and then let another barrier take over the task considering the objective is no intruder can cross the field without being detected and any on-duty barrier can prevent this from happening. In the 1D case, this can be a good solution. However, this is not a viable solution in the 2D case, in which work load balance is critical to maintain an on-duty barrier on both the horizontal and vertical directions.

In the 2D case, any barrier in one direction intersects with all the barriers in the other direction. The death of one barrier may result in a security breach of all its intersecting barriers. This means, the death of one horizontal barrier may destroy all the vertical barriers, even though other horizontal barriers can take over its task in the horizontal direction.

6. Reconstruction algorithm

In this section, a reconstruction algorithm is introduced. During the network operational time, sensors may fail due to energy depletion or casual damage. Group/barrier reconstruction is then conducted to maintain the sensing quality. To detect sensor failure, each node is required to broadcast an Alive Message to its neighbor after it wakes up. If a node does not receive the message from its neighbor node for certain time, \( t' \), it determines that the neighbor node is dead. \( \Delta t \) is set to be longer than \( t' \) to ensure there is enough time for the reconstruction.

Once a sensor node fails, two neighboring nodes in its barrier can detect the failure. The one which is the previous node of the dead node in the barrier construction algorithm is responsible for

![Fig. 13. Connection nodes searching example in one group.](image)

![Fig. 14. Start node delegation.](image)
looking for a replacement node. As shown in Fig. 15, node A, B, C, E and F belong to one barrier \( s_i \) and monitor one line of the field. Node B fails and both A and C detect the failure. Since A is the previous node of C in the searching, A takes the task to look for a replacement node. Node A finds redundant node D according to replacement Node Selection Rules, which is presented in the following paragraph, and sends D a borrow message, specifying all necessary information to calculate the wake-up/sleep time. Node D accepts it and broadcasts an Acknowledgement. Both A and C receive the message and establish a connection with D. After that, D joins barrier \( s_i \) and \( s_i \) composed of node A, C, D, E and F keeps monitoring the field effectively.

The replacement node of a failed node in a barrier must satisfy a prerequisite: with the replacement node, the barrier can monitor one line of the field and no intruder can go across the line without being detected. To satisfy the requirement, the distance between the replacement node and the two neighbors of the failed node must be shorter than twice of the sensing range. For example, in the case of Fig. 15, \( d_{AD} \leq 2R_S \) and \( d_{AC} \leq 2R_S \).

Among all the nodes which satisfy the prerequisite, the replacement node is selected according to the following rules:

- **Rule 1**: Closest redundant node to the two neighbors will be selected.
- **Rule 2**: If there is no redundant node nearby, a closest neighboring node in the same group will be selected. Note that in the scheduling algorithm, we require every node, which is a one-hop sensing neighbor of an active node, to be awake for \( \Delta t \), such that they can be borrowed in case of any node failure.
- **Rule 3**: If no suitable node can be found based on the above two criteria, the closest node in other group will be selected as the replacement node. In the scheduling algorithm, there is a short overlap time between previous shift and current shift. If the previous barrier is close enough to the current barrier, the current barrier will borrow nodes from previous barrier, which belongs to another group.
- **Rule 4**: If still no node can be found, which means, the current barrier cannot be repaired, every alive node in this barrier will become redundant nodes and wait to help other barriers in case some nodes fail.

If a replacement node is selected based on Rule 2 or Rule 3, the original barrier of the replacement node also needs to find a node to replace it. The procedure is the same except that it is the replacement node which notifies its previous node to start searching a node to replace itself. To avoid the situation that a node is borrowed back and forth between two barriers and the reconstruction is conducted infinitely, we specify that a node in a barrier can only be borrowed once.

The searching of a replacement node can be done pro-actively. In the beginning of a shift, every node checks its energy level. If there is no enough energy left to sustain the node through its shift, it will search for a replacement node pro-actively using the same selection rules.

### 7. Performance evaluation

#### 7.1. Simulation methodology

2D k-barrier is evaluated through simulation and implemented in the NS2 simulator. The objectives of the evaluation are three-fold: (1) testing the effectiveness of our protocol in conserving energy and prolonging network lifetime; (2) studying the quality of our protocol in detecting intruders in terms of detection delay; and (3) studying the performance of our protocol under different system parameters, such as shift duration, and under different system inputs, such as the maximum speed of intruders. We choose detection delay and network lifetime as metrics to evaluate the performance of 2D k-barrier algorithm. To conduct the evaluation, we use 2D-Mesh [8] as a benchmark, which is the related work closest to our research and is also about scheduling sensors on and off to save energy in intruder detection applications.

In our simulation, 800 nodes are randomly deployed in a \( 400 \times 400 \text{ m}^2 \) rectangular field. Intruders try to cross the field from one side to the opposite side. Without loss of generality, we assume intruders always move from any of four edges of the field. Unless we study the impact of the system parameters and intruder characteristics, we choose the following parameter values in the simulation. The speed of intruders is 10 m/s. The number of intruder is 10. The sensing range is 20 m and communication range is 40 m [8]. The duration of a shift is 2 s. The protocol runs for 10,000 s. Each sensor carries 100 s of energy initially. When a sensor sleeps for 1 s, it consumes 0.05 unit energy. When a sensor is on for 1 s, it consumes a random unit of energy between 1 and 1.01 unit. This is to simulate the case that some sensors detect an intruder and send alert signal and some forward messages, so they consume more energy, while others do nothing and consume less energy. The intruders start to cross the field in a random time after the network initialization. To study our algorithm thoroughly, 1D 1-Barrier, which is the SCAN algorithm in [30], 2D 1-Barrier and 2D 2-Barrier are also evaluated, the latter two have different settings of k. All simulation results are the average of 10 experiments which are on different initial deployments.

#### 7.2. Impact of intruder speed and number

To evaluate the impact of intruder speed on 2D k-barrier, the intruder speed varies from 10 m/s to 60 m/s with an increment of 10 m/s. Fig. 16(a) shows the result. We can see that the detection delay in all four schemes decreases as the speed increases. The performance of 2D 2-Barrier is always the best and is half of the performance of 2D 1-Barrier, which is also half of the performance of SCAN. This is because 2D 2-Barrier can provide one more barrier in detecting intruders than 2D 1-Barrier, and 2D 1-Barrier can provide one time barrier than SCAN. In 2D k-barrier, when a node fails, a replacement node can be found and intruder can be detected quickly. However, in 2D-Mesh, when a node fails, an intruder may be able to pass the band without being detected. It may be detected by the sensors in the next band, but this increases the detection delay greatly.

To evaluate the impact of intruder number, we vary it from 10 to 60 with an increment of 10. Fig. 16(b) shows the result. We can see that the number of intruders has a significant impact on the detection delay in all schemes. Same as the previous results, 2D 2-Barrier performs the best since it provides more barriers. Even
SCAN can provide 30% lower detection delay compared with 2D-Mesh.

7.3. Impact of shift duration

The shift duration is set to be 1 s, 2 s, 3 s, 4 s and 5 s, respectively for each evaluation. Detection delay under different awake time is shown in Fig. 17(a). From Fig. 17(a), we can see 2D 2-Barrier and 2D 1-Barrier can provide about 83% and 63% shorter detection delay on average compared with 2D-Mesh. The delay for 2D 2-Barrier and 2D 1-Barrier increases with a longer shift or lower scanning frequency. Therefore, when an application requires low detection delay, a higher scanning frequency can be set to achieve that.

The lifetime under different shift durations is shown in Fig. 17(b). The lifetime of SCAN is at least four times longer than that of 2D-Mesh, the lifetime of 2D 1-Barrier is at least two times longer than that of 2D-Mesh, and the lifetime of 2D 2-Barrier is higher than 2D-Mesh. This is because when k is 2, the number of on-duty sensors in 2D 2-Barrier is close to that in 2D-Mesh. With a long shift duration, 2D k-barrier provides a longer lifetime. This is because with a longer shift duration and a fixed overlap time Δt, the proportion of two barriers of nodes working together decreases. Therefore, less energy is consumed and the network lifetime increases.

From Fig. 17(a) and (b) we can see that a longer shift duration increases network lifetime but also prolongs the detection delay. A proper shift duration can be set up to balance the two metrics.

7.4. Impact of preloaded energy

To evaluate the impact of preloaded energy, we set it to be 100, 150, 200, 250 and 300 units and evaluate the performance in each case. The results are shown in Fig. 18.

From Fig. 18(a), we can see that 2D 2-Barrier provides a shortest detection delay among four algorithms and both 2D 1-Barrier and SCAN have shorter detection delay than 2D-Mesh. This is because in 2D k-barrier, when a node fails, a replacement node can be found and intruder can be detected quickly. However, in 2D-Mesh, when a node fails, an intruder may be able to pass the band without being detected. It may be detected by the sensors in the next band, which increases the detection delay. From the figure, it also can be observed that as the preloaded energy increases, the detection delay of all protocols decreases. This is because with more energy, the network can operate in a good status for a longer proportion and thus the average detection delay is shorter.

From Fig. 18(b), we can see that, as expected, the network lifetime in all schemes increases as the preloaded energy increases. In the percentage perspective, SCAN increases the lifetime much more than other three schemes as the energy level increases. This
means, with less number of barriers in the network, the energy can be much more effectively utilized to provide the monitoring service by the proposed 2D k-barrier scheme. Note that, the ideal case is the network life increases the same percentage as the preloaded energy. For example, with a double energy amount, the lifetime increases by 100%. However, in reality, because of the random energy consumption and uneven sensor distribution, the more energy preloaded, the higher the remaining energy variance is among sensor nodes. This means, when the preloaded energy is high, it is possible some nodes still have a lot of energy when most sensors die, and thus more energy is wasted. Because the number of on-duty sensors in 2D 2-Barrier is close to in 2D-Mesh as mentioned above, thus the lifetime of 2D 2-Barrier and 2D-Mesh is close to each other.

8. Related work

Prior works using barriers for intruder detection have been studied a lot. Theoretical foundations of designing barriers of wireless sensors are developed in [15]. The barrier coverage problem is very difficult to solve in a decentralized way due to its globalized nature. Chen et al. address this challenge by introducing the concept of local barrier coverage in [2]. In [21], Liu et al. propose a distributed algorithm to construct multiple disjoint barriers for strong barrier coverage when sensors are distributed according to Poisson point process. Chen et al. investigate the quality of barrier coverage in [4]. In [26], Saipulla et al. study the barrier coverage problem when sensors are deployed along a line. To further utilize the barrier concepts in wireless sensor networks, Kumar et al. develop solutions for the case when wireless sensors are deployed to form an impenetrable barrier for detecting movements in [16]. In [3], Chen et al. introduce a concept of local barrier coverage and propose a localized barrier coverage protocol. The concept of one-way barrier coverage is introduced in [5], which requires that the network reports illegal intruders with certain direction of the movement. In [37], Yang et al. split sensor deployment into multiple rounds to reduce the number of sensors needed to provide guaranteed barrier coverage. Lin et al. investigate the probability of achieving connected barrier coverage on a finite narrow band when sensors with given sensing/communication radius are randomly deployed in [20]. Li et al. study the barrier’s detection probability under probabilistic sensing model in [17]. There are also some works using mobile sensors to detect intruders. In [14], mobile sensors are used to provide k-barrier coverage against moving intruders. Saipulla et al. study how to efficiently improve barrier coverage using mobile sensors with limited mobility in [25]. He et al. use mobile sensors to achieve barrier coverage in sensor scarcity case by dynamic sensor patrolling in [10]. However, these solutions do not consider minimizing sensing overlap between two adjacent sensors in a barrier to increase network lifetime.

Besides the intruder detection with barriers, the works using static sensors for target detection have also been presented in [18,36,32,13,33,29], which mainly focus on effectively detecting or tracking the presence of an intruder or event. Lin et al. develop a logic object-tracking tree to reduce the communication cost in [18], while in [9], two virtual patrol modes are proposed to conduct surveillance of a given area. A differentiated surveillance model is proposed in [36]. The concept of virtual sensor is introduced in [32] to improve coverage, which is based on neighbor sensors’ collaboration and value fusion. Employing path exposure as a measurement of coverage quality, a scheme is proposed in [1] to reduce sensor installation cost as much as possible while providing a minimum exposure. Tian et al. propose a dynamic intruder detection scheme using mobile and static sensors to detect smart intruders [29]. Wang et al. discuss expected intrusion distance under two WSN models: homogeneous and heterogeneous WSN in [33], which have different sensing ranges. However, none of the listed works considers the network lifetime.

To conserve energy consumption, a lot of works are proposed on duty-cycle scheduling in sensor networks. For example, LEACH [11] and HEED [39] are proposed to cluster sensor nodes to prolong the network lifetime. For field surveillance, Gui et al. develop two efficient sleep–awake schemes to minimize the power consumption in [8]. Another scheduling protocol is proposed in [12]. The work is based on the prediction of mobile target’s track in sensor networks. A TDMA sleep scheduling problem is studied in [23]. In [19], Lin et al. present an adaptive multisensor scheduling scheme for collaborative target tracking in WSNs. However, the proposed schemes do not especially focus on mobile target detection and energy consumption can be reduced in the context.

9. Conclusion

In this paper, we propose a new scheduling protocol 2D k-barrier in sensor networks for intruder detection, which constructs k barriers in both horizontal direction and vertical direction of the field of interest and make them work like a searchlight to monitor the field. The simulation results show that our scheduling protocol can maintain a lower detection delay compared with other scheme and consume much less energy and thus prolong the network lifetime greatly. This scheme also provides fault-tolerance. By running
the reconstruction algorithm automatically, sensor networks keep working effectively even when a percentage of sensors fail.

**Acknowledgment**

This work is supported by the National Science Foundation Grants NSF-1128369.

**References**


