Maximum Lifetime Combined Barrier-coverage of Weak Static Sensors and Strong Mobile Sensors

Donghyun Kim, Senior Member, IEEE, Wei Wang, Junggab Son, Member, IEEE, Weili Wu, Senior Member, IEEE, Wonjun Lee, Senior Member, IEEE, Alade O. Tokuta, Member, IEEE

Abstract—Recently, the concept of barrier-coverage of wireless sensor network has been introduced for various civilian and military defense applications. This paper studies the problem of how to organize hybrid sensor network, which consists of a number of energy-scarce ground sensors with homogenous initial battery level and energy-plentiful mobile sensors, to maximum the lifetime of barrier-coverage. Two key observations are (a) as the lifetime of each mobile sensor is much longer than that of the static ground sensors, each mobile sensor is capable of contributing multiple sensor barrier formations, and (b) no mobile sensor node can join two hybrid barriers which will be successively used to continuously protect the area of interest due to the moving delay. Based on these, we introduce a new maximum lifetime barrier-coverage problem in hybrid sensor network. We first propose a simple heuristic algorithm by combining existing ideas along with our own. Then, we design another efficient algorithm for the problem and prove that the lifetime of hybrid barrier constructed by this algorithm is at least three times greater than the existing one on average. Our simulation result shows that the second algorithm outperforms the first algorithm at least 33% and up to 100%.

Index Terms—Sensor networks, maximum lifetime, hybrid sensor networks, barrier-coverage, graph theory.

1 INTRODUCTION

For more than a decade, the concept of wireless sensor network, a wireless network of micro-electronic computing devices, each of which is equipped with one or more sensors and a radio transceiver for wireless communications (and therefore it is commonly referred as a wireless sensor node), has been investigated for a wide range of monitoring and surveillance applications. In the majority of the studies, researchers considered a wireless sensor network of a number of cheap air-deployed ground sensor nodes, whose physical capabilities are limited, and who are redundant, e.g. a target or an area of interest is likely to be within the sensing area of more than one sensor node. Naturally, one popular research topic in this area is how to better exploit the redundancy so that the lifetime of the coverage of a sensor network, which is defined as the continuous time period during which the surveillance quality provided by the sensor network satisfies a certain requirement, is maximized.

There are several coverage models which attracted much attention from the research community over years. Among those, full-coverage model is the most intensively investigated: a wireless sensor network is told to provide full-coverage over an area of interest if any activity of interest which occurs in the area can be monitored by the sensor network. In contrast, barrier-coverage model is a new coverage model which is recently getting more considerations: a sensor network provides barrier-coverage over an area of interest if an intruder which is trying to move across the area will always be detected by a sensor node. By nature, the barrier-coverage model is useful for various civilian and military defense scenarios such as detecting enemy soldiers approaching an ally base at night or monitoring an approach of unauthorized personal to steal a valuable asset. Also, by definition, a sensor network loses its barrier-coverage over an area if there is a non-zero chance for an intruder to move across the area without being detected by the sensor network. Once a sensor barrier is deployed, some of the nodes may fail to due many reasons and an intruder can move into the area of interest without being detected. The space used to be monitored by a sensor node, but is not covered as the node dies, and now the intruder exploits to trespass, is frequently referred as “gap” in the literature [23].

Recently, various mobile sensor nodes such as drones become available in both military and commercial markets. Due to the reason, hybrid sensor networks, which consist of both weak ground sensor nodes and strong mobile sensor nodes, are getting more attentions. It is easy to imagine a situation in which a hybrid sensor network can provide barrier-coverage while a wireless
Fig. 1: In this figure, an intruder is assumed to move from $T$ (the top area of the square region) to $B$ (the bottom area of the square region), and is only allowed to move across the squared region. The sensor nodes are with uniform lifetime $s_1, s_2, \ldots, s_6$ and work together with a mobile node $m_1$ to detect the intruder. Apparently, any subset of the static ground sensors can provide the desired barrier-coverage. However, $\{m_1, s_4, s_5, s_6\}$ can. Also, by relocating $m_1$ to $m'_1$, $\{m'_1, s_1, s_2, s_3\}$ can. Unfortunately, after $\{m_1, s_4, s_5, s_6\}$ is employed, $\{m'_1, s_1, s_2, s_3\}$ is not immediately available as we need to wait for $m_1$ to relocate to $m'_1$.

Despite the profound potential of hybrid sensor network, we found that in opposition to barrier-coverage of wireless sensor networks of static sensor nodes, barrier-coverage of hybrid sensor networks with both static and mobile sensor nodes did not receive much attention so far. In particular, while a couple of reports such as [23] exploit fully controllable mobile sensor nodes with powerful hardware and plenty amount of resources to improve the fault-tolerance of barrier-coverage of wireless sensor network, there is generally a significant lack of efforts to exploit the mobile sensor nodes to maximize the lifetime of barrier-coverage in the context of hybrid sensor networks, which is the main research interest of this paper. To the best of our knowledge, the most relevant work to this problem is done by Kumar et al. [21], in which they showed that the lifetime of barrier-coverage of a wireless sensor network of homogenous ground sensors (i.e. with the same physical capabilities) could be extended by splitting the sensor nodes into the maximum number of node-disjoint subsets such that each subset can provide barrier-coverage over the area of interest, and operate one by one. They also provided a polynomial time exact algorithm for the problem. However, this result can hardly be applicable to the hybrid sensor network case.

This paper considers a hybrid wireless sensor network of two different homogenous sensor node groups. The first one consists of a number of energy-limited static sensor nodes with the same initial battery level and the second one consists of several energy-rechargeable (e.g. via onboard solar battery or mobile charger [35], [36], [37]) mobile sensor nodes. We introduce a new strategy to maximize the lifetime of such hybrid wireless sensor network by

(a) carefully organizing static ground sensors into subsets, and

(b) properly relocating the available mobile sensors, which assumes to have plentiful amount of resources, e.g. ground vehicles with solar battery.

We formally define this new maximum lifetime coverage problem as the maximum lifetime barrier-coverage in hybrid sensor network problem (MLB-HSN). We observe that a feasible solution of MLB-HSN can be obtained by going through the following steps:

(a) **First step (first subproblem):** given a set of static sensors and mobile sensors, this step is about how to compute the maximum number of barriers, which are static-node-wise-node-disjoint subsets, each of which can provide barrier-coverage by collaborating with the available mobile sensors.

(b) **Second step (second subproblem):** this step is about how to appropriately schedule the subsets computed from the first step. This is an important issue (and a new issue we found for the first time in the literature) because when we have a limited number of mobile sensor nodes, we cannot immediately employ a hybrid sensor barrier which requires all of the available mobile sensor nodes right after operating a hybrid sensor barrier which also requires some mobile sensor nodes as the mobile sensor nodes needs to relocate (e.g. see Fig. 1).

We would like to emphasize that the actual lifetime of the barrier-coverage of a hybrid sensor network achieved by the second step would be shorter than the sum of lifetime of the individual barriers which are formed by the subsets from the first step. This means that we need to devise to an efficient way to schedule the subsets such that the lifetime of barrier-coverage of the hybrid sensor network can be maximized. This also implies that the two steps are somehow related.

The list of the contributions of this paper can be summarized as follows.

(a) We introduce a new maximum lifetime barrier-coverage problem in hybrid sensor network, namely MLB-HSN, which considers the latency caused by the relocation of mobile sensor nodes in the scheduling for the first time in the literature. By the definition (Section 3), the lifetime of barrier-coverage of hybrid sensor network corresponds the number of static-node-disjoint barriers when the remaining energy level of each static node is same and each mobile node has a capability to recharge their energy. However, the number of mobile nodes in one barrier and another barrier which is used after all static nodes in the previous barrier are exhausted cannot
exceed the total number of mobile nodes. We also propose a simple heuristic algorithm for MLB-HSN based on the existing results in the literature.

(b) To deal with the second step, we introduce a new problem called the longest sequence problem (LSP). Given a set of non-negative integers (each of which represents the number of required mobile nodes to form a hybrid barrier-cover) and a fixed positive integer \( k \) (which is the number of all available mobile sensor nodes), the goal of LSP is to find the longest sequence of the integers such that the sum of any pair of adjacent nodes does not exceed \( k \). In this paper, we propose a polynomial time exact algorithm for LSP.

(c) Finally, we propose a polynomial time 4-approximation (on average sense, but we will refer this as 4-approximation during the rest of this paper for the sake of the convenience of the writing) algorithm for MLB-HSN based on our result for LSP as it can be used to evaluate the quality of any given set of hybrid barrier-covers. We first define the maximum proper-node disjoint paths problem (MaxPN-DPP), which is an abstracted mathematical formulation of MLB-HSN, whose goal is to find the maximum number of node-disjoint paths which can be arranged in a way that the sum of the weights of two consecutive path in the arrangement is no greater than \( k \). Then, we use an existing algorithm for the maximum node-disjoint paths with bounded average length problem (MaxPath-BALP), whose goal is to compute the maximum number of node-disjoint paths such that the average length of the paths is bounded by a given constant, to compute a set of paths. Next, we remove all the paths whose weight is greater than \( k \). Next, we arrange the remaining paths by using our optimal algorithm for LSP. Finally, we prove this strategy is a 4-approximation algorithm for MaxPN-DPP.

The rest of this paper is organized as follows. Section 3 presents the formal definition of MLB-HSN, our problem of interest. The related work is discussed in Section 2. In Section 4, we introduce a simple heuristic algorithm for MLB-HSN based on existing results. We propose a new 4-approximation algorithm for MLB-HSN in Section 5. We present our simulation result in Section 6 and conclude this paper in Section 7.

2 Related Work

Over years, the maximum lifetime sleep-wakeup scheduling problem in wireless sensor networks has attracted lots of attentions [27]. Frequently, the problem of finding maximum lifetime coverage scheduling is modeled as the problem computing the maximum number of node-disjoint subsets of wireless sensors in a way that each subset can meet a certain coverage requirement. Since this problem is NP-hard, most efforts are made to design approximation algorithms. In [14], Cardei et al. showed that by allowing one node to be included in more than one subset, the lifetime of coverage can be even further extended.

Recently, the concept of barrier-coverage of wireless sensor network has been introduced for various civilian and military application scenarios in which valuable assets and/or personals are protected from physical intruders by deploying a barrier of sensors around the assets and/or personals to observe the intruders. In [19], Kumar et al. introduced the concept of \( k \)-barrier-coverage. A sensor network provide \( k \)-barrier-coverage over an area of interest when an intruder moves into an area of interest, at least \( k \) sensors should be able to detect this. As \( k \)-barrier-coverage offers fault-tolerance which is deficient in the pure barrier-coverage model, a considerable amount of attention has given to this model. In [21], Kumar et al. investigated a sleep-wakeup scheduling problem for \( k \)-barrier-coverage of wireless sensors. The goal of this problem is to identify the maximum number of node-disjoint subsets in a way that each subset can offer \( k \)-barrier-coverage. Unlike some coverage model such as full-coverage, Kumar et al. showed that the problem of computing maximum number of node-disjoint subsets such that each subset can serve as a seamless barrier is polynomial time solvable. A distributed algorithm for this problem is introduced by Ban et al. [28].

With the recent advances in micro-electronic technologies, a number of mobile sensor nodes are introduced. As a result, a number of researches have been conducted to utilize mobile sensor nodes to improve barrier coverage. In [33], Shen et al. has studied how to efficiently relocate mobile sensor nodes to provide a barrier. In [34], Wang et al. investigated a barrier coverage problem in which gaps among static sensor nodes are filled by minimum number of directional mobile sensor nodes. In [23], Wang et al. studied the minimum number of mobile nodes to achieve \( k \)-barrier-coverage with a given set of static sensor nodes. In [24], Ma et al. showed that by allowing one node to be included in more than one subset, the lifetime of coverage can be even further extended.

One may believe that an algorithm for \( k \)-barrier-coverage would be used for the maximum lifetime barrier coverage problem in hybrid sensor network. That is, we may try to compute maximum \( k \)-barriers and use one by one. In fact, this argument is correct for purely static sensor networks as Kumar et al. claimed [21]. However,
as we discussed earlier in Section 1, this is not true in hybrid sensor networks, which is our main motivation to conduct this research. As a result, our problem becomes very unique and there is no result directly applicable to solve our problem of interest.

3 Formal Definition of MLB-HSN

3.1 Notations and Assumptions

In this paper, we assume that there are a set $S$ of $n$ static homogenous ground sensor nodes, $\{s_1, \ldots, s_n\}$ and a set $M$ of $k$ mobile homogenous sensor nodes, $\{m_1, \ldots, m_k\}$ over an area of interest $A$. We assume that the battery lifetime of the static sensors are same. We also assume that the mobile sensor nodes have the capability of replenishing their battery via some means such as solar battery, and therefore, their lifetime is much longer than the lifespan of a static sensor node as well as the length of the mission period during which the barrier coverage should be provided. We would like to emphasize that the actual way to replenish the battery of each mobile node is out of the scope of this research. Possibly, the relatively much longer lifetime of each mobile node is realized using several existing technique such as the adoption of mobile chargers as discussed in [35], [36], [37]. As observed in [21], in such case, we cannot improve the lifetime of barrier coverage by allowing one static sensor node to join more than one barrier. Finally, we assume that the speed of each mobile sensor nodes is fast enough such that a mobile node can move from one location to another location in $A$ during the lifetime of a static sensor node with fully charged battery. For instance, given a 100 mile by 100 mile area, when SmartDusts are deployed (whose regular lifetime is known to be one week, or equivalently 168 hours [1]), the speed of a mobile node should be at least 0.84 mile per hour (note that the speed of state-of-art toy drones easily exceeds 11 mile per hour [2]). As a result, this assumption is reasonable even the time for recharging is factored in.

3.2 Problem Statement

Now, let us introduce some important definitions first and provide the formal statement of the problem of interest.

Definition 1 (Barrier-coverage). A sensor network provides barrier-coverage over $A$ if it can form a seamless barrier with the sensing ranges of the active sensors (which are not sleeping) and detect any intruder from upper side of $A$ to lower side of $A$, e.g. $T$ and $B$ in Fig. 1, respectively.

Definition 2 (Hybrid (sensor) barrier). A hybrid barrier consists a subset of static ground sensors and mobile sensors and cooperatively provides barrier-coverage over $A$.

Note that a hybrid barrier may consist of one kind of sensors only, e.g. mobile sensors only or static sensors only. However, we assume the number of mobile sensor nodes are not sufficient to provide permanent barrier-coverage over $A$, otherwise the problem of our interest is trivial.

Definition 3 (Legitimate alternation of two barriers). Consider two hybrid barriers, $b_1$ and $b_2$. Then, the alternation (replacement) of $b_1$ to $b_2$ is legitimate only if the number of mobile sensor nodes used to form $b_1$, namely $M_{b_1}$, and the number of mobile sensor nodes to form $b_2$, which is $M_{b_2}$ is no greater than $k$, i.e. $|M_{b_1}| + |M_{b_2}| \leq k$.

The concept of legitimate alternation of two barriers is new, but significant as no mobile barrier can jump from one location to another location instantly, and any alternation of two barriers which is not legitimate will cause a hole in one of the barrier, due to the reason.

Definition 4 (Maximum lifetime barrier-coverage in hybrid sensor network problem). Given $S, M,$ and $A,$ the goal of the maximum lifetime barrier-coverage in hybrid sensor network problem (MLB-HSN) is to find the largest sequence $B$ of subsets $\{b_1, \ldots, b_l\}$ of $S \cup M$ (or alternatively maximize $l$) such that for any two $b_i, b_j \in B$,

(a) $(b_i \cap S) \cap (b_j \cap S) = \emptyset$, and
(b) for any $1 \leq i \leq l - 1$, $|M_{b_i}| + |M_{b_{i+1}}| \leq k$.

4 A Simple Heuristic Algorithm for MLB-HSN

In this section, we discuss a simple heuristic algorithm for MLB-HSN by exploiting existing results from Kumar et al.’s [21], Wang et al.’s [23], etc. Note that this algorithm may produce a sub-optimal solution. In detail, given a MLB-HSN problem instance $\langle S, M, A \rangle$, we first compute a feasible solution of the first subproblem and obtain a set of hybrid sensor barriers, $B = \{b_1, b_2, \ldots, b_p\}$, such that for each pair of $b_i, b_j \in B$, $(b_i \cap S) \cap (b_j \cap S) = \emptyset$, and $|M_{b_i}| \leq k$ and $|M_{b_j}| \leq k$. Then, we solve the second subproblem by constructing...
a sequence $B' = \{b'_1, b'_2, \cdots, b'_q\}$ of hybrid barriers from $B$ such that for any $1 \leq i \leq q-1$, $|M_{b^i}| + |M_{b_{i+1}}| \leq k$.

Note that if $k' \left\lceil \frac{x}{y} \right\rceil$ is used instead of $k$ to solve the first subproblem, the sequence of hybrid barriers $B$ computed by solving the first subprogram can always be enumerated in an arbitrary order and we can obtain a sequence $B'$ whose elements are exactly same to $B$. Therefore, now we discuss how to solve the first subproblem during the rest of this subsection. The detail of our heuristic algorithm for the first subproblem is as follows.

(a) Given $S$ and $A$, we first construct a new graph $\mathcal{G} = (V(\mathcal{G}), E(\mathcal{G}))$ (see Fig. 2(b)) from a given set $S$ of ground sensor nodes (see Fig. 2(a)) as follows:
   (i) set $V(\mathcal{G}) = S$,
   (ii) for each $u, v \in V(\mathcal{G})$, $(u, v) \in E(\mathcal{G})$ only if the sensing ranges of $u$ and $v$ are touching or overlapping,
   (iii) add two new vertices $s$ and $t$ to $V(\mathcal{G})$, and
   (iv) for each $u \in V(\mathcal{G})$, $(u, s) \in E(\mathcal{G})$ if the sensing range of $u$ is touching or covering the left border of $A$. Similarly, $(u, t) \in E(\mathcal{G})$ if the sensing range of $u$ is touching or covering the right border of $A$.

(b) $\mathcal{G}$ is transformed into $\tilde{\mathcal{G}}$ (see Fig. 2(c)) such that each vertex $u \in V(\mathcal{G})$ is split into two vertices $u_{in}$ and $u_{out}$, and there is a directional edge, $u_{in} \rightarrow u_{out}$. For each incoming edge from $w$ to $u$ in $\mathcal{G}$, there is a directional edge from $w_{out}$ to $u_{in}$ in $\tilde{\mathcal{G}}$ as well as for each outgoing edge from $u$ to $w$ in $\mathcal{G}$, there is a directional edge from $u_{out}$ to $w_{in}$ in $\tilde{\mathcal{G}}$ with its capacity to be 1. For the rest of the edges, we set their capacity as infinity.

(c) We construct a new complete graph $\hat{\mathcal{G}}$ from $\tilde{\mathcal{G}}$ as follows:
   (i) Copy $\tilde{\mathcal{G}}$ from $\mathcal{G}$ (i.e. set $\hat{\mathcal{G}} \leftarrow \tilde{\mathcal{G}}$).
   (ii) We set the cost of each node in $V(\hat{\mathcal{G}})$ to be 0.
   (iii) For each pair of nodes $u_{out}, v_{in} \in E(\hat{\mathcal{G}})$, if $(u_{out}, v_{in}) \notin \hat{\mathcal{G}}$, then we first add a new node $z_{u_{out}, v_{in}}$ to $V(\hat{\mathcal{G}})$. The weight of this new node is equivalent to the minimum number of mobile nodes to make a barrier from $u$ to $v$ (see Fig. 3).

(d) Now, we try to compute the maximum number of $s$ to $t$ path in $\hat{\mathcal{G}}$ such that the cost of each path (i.e. the sum of node weights in the path) does not exceed $k$.

Clearly, each path represents a hybrid sensor barrier crossing over $A$ such that the number of mobile sensor nodes (the cost of each path) does not exceed $k$. Also the paths are static-node-wise-disjoint, and therefore this strategy produces a feasible solution of MLB-HSN. Unfortunately, the problem of computing maximum number of node-disjoint paths such that each path cost does not exceed a given constant $k \geq 4$ is known to be not only NP-hard, but also APX-complete, which means that no PTAS exists for the problem [38]. To the best of our knowledge, there is no known approximation algorithm for this problem, but a few heuristics are reported in the literature [4], [5], [3], [6]. Apparently, we can use one of them to solve the first subproblem. Meanwhile, even if the the problem of computing maximum number of node-disjoint paths such that each path cost does not exceed a given constant $k \geq 4$, this approach still does not produce an optimal solution of MLB-HSN as at the beginning, we set $k' \left\lceil \frac{x}{y} \right\rceil$ to deal with the second subproblem rather than providing a more careful treatment of it. Apparently, MLB-HSN is a highly complicated problem.

5 A NEW 4-APPROXIMATION FOR MLB-HSN

Previously, we introduced a heuristic algorithm for MLB-HSN. Our main idea was to deal with each step one by one. We solved the first subproblem, which is APX-complete, using a new heuristic algorithm and used its output as an input of the second subproblem. While we manage to obtain a feasible solution, the step-by-step strategy is quite complicated and therefore it is difficult to obtain a performance guarantee. To address this issue, in this section, we introduce a polynomial time 4-approximation algorithm for MLB-HSN. In the next subsection, we first propose a polynomial time exact algorithm for the second step. Then, we use this result to obtain a 4-approximation algorithm for MLB-HSN in the following subsection.

5.1 A Polynomial Time Exact Algorithm for The Second Step

Definition 5 (The longest sequence problem). Given a (multi-) set of non-negative integers $S = \{c_1, c_2, \cdots, c_m\}$ and a positive integer $k$, the goal of the longest sequence problem (LSP) is to find a sequence $S' = (c'_1, c'_2, \cdots, c'_{m'})$, $c'_i \in S$ with the property that $c'_i + c'_{i+1} \leq k$ for $i = 1, 2, \cdots, m' - 1$ such that the length $m'$ of $S'$ is maximized.

We can reformulate LSP as the Longest Path Problem in Graph Theory. First, we construct a graph $\Gamma = (V, E)$ as follows: the vertex set $V = \{v_1, v_2, \cdots, v_m\}$ with each $v_i$ corresponding to the cost $c_i$, and $(v_i, v_j) \in E$ if and only if $c_i + c_j \leq k$. Then LSP can be reformulated as
the problem of finding the longest path in $\Gamma$. Notice that finding the longest path in general graphs is NP-hard, however, the graph $\Gamma$ we constructed above has very special structures which enable us to find the longest path in $\Gamma$ efficiently in an explicit way.

Without loss of generality, we assume $c_1 \leq c_2 \leq \cdots \leq c_m$. For each $2 \leq i \leq m$, let $I(i)$ be the maximum index among $1, 2, \cdots, i-1$ such that $c_i + c_{I(i)} \leq k$ (we assume $I(i)$ always exists, otherwise $v_i$ is an isolated vertex in $\Gamma$ and can be removed without affecting the results of the paper).

Define a subset of nodes $S_i = \{v_j | j = 1, 2, \cdots, I(i)\}$. Next, we present some structural properties of graph $\Gamma$.

**Lemma 1.** The induced graph $\Gamma[S_i \cup \{v_i\}]$ is a complete graph.

**Proof.** Note that $c_1 \leq c_2 \leq \cdots \leq c_{I(i)}$ and $c_i + c_{I(i)} \leq k$. It follows that $c_p + c_q \leq k$ for $1 \leq p, q \leq I(i)$. Thus, $(v_p, v_q) \in E(\Gamma)$. Moreover, $c_i + c_j \leq k$ for any $1 \leq j \leq I(i)$. It follows that $(v_i, v_j) \in E(\Gamma)$. Thus, any pair of vertices in $\Gamma[S_i \cup \{v_i\}]$ are adjacent.

**Lemma 2.** Let $l$ be the maximum index among $1, 2, \cdots, m-1$ such that $c_l + c_{l+1} \leq k$. Then $\Gamma$ has the following structures:

(a) The induced graph $\Gamma[\{v_{l+1}, \cdots, v_{m-1}, v_m\}]$ is an empty graph,

(b) $S_m \subset S_{m-1} \subset \cdots \subset S_l \subset S_{l+1}$, and

(c) $v_i$ is adjacent to every node in $S_l$ but not adjacent to nodes in $V \setminus S_l$, for $i = l+1, \cdots, m$.

**Proof.** For (a), notice that $l$ is the largest index such that $c_l + c_{l+1} \leq k$, it follows that $c_{l+1} + c_{l+2} > k$ and hence, $c_p + c_q > k$ for any $1 \leq p, q \leq m$. For (b), since $c_1 \leq c_2 \leq \cdots \leq c_m$, we have $I(p) \leq I(q)$ if $p \geq q$, and hence $S_p \subset S_q$. At last, (c) follows by the definition of $l$.

Fig. 4 gives an illustration for the structures of $\Gamma$.

**Theorem 1.** Suppose that $\Gamma$ has no isolated vertices and

$$|S_i| \geq m - i + 1, \quad (1)$$

for each $i = l + 1, l + 2, \cdots, m$. Then $\Gamma$ has a Hamiltonian path, i.e., a path that visits all the nodes of $\Gamma$.

**Proof.** By the assumption $(1)$, we have $|S_m| \geq 1, |S_{m-1}| \geq 2, \cdots, |S_{l+1}| \geq m - l$. It follows that there exist distinct nodes, say, $\tilde{v}_m, \tilde{v}_{m-1}, \cdots, \tilde{v}_{l+1}$ such that $\tilde{v}_i \in S_i$ for $i = l + 1, \cdots, m$. Let $S_{l+1} \setminus \{\tilde{v}_m, \tilde{v}_{m-1}, \cdots, \tilde{v}_{l+1}\} = \{v'_1, v'_2, \cdots, v'_s\}$. Then we can construct a path as follows:

$$v_m \to \tilde{v}_m \to v_{m-1} \to \tilde{v}_{m-1} \to \cdots \to \tilde{v}_{l+2} \to v_{l+1} \to \quad (2)$$

$$\tilde{v}_{l+1} \to v'_1 \to v'_2 \to \cdots \to v'_s. \quad (3)$$

It is easy to verify the above path is actually a Hamiltonian path.

Clearly, if $S_{l+1}$ is a proper subset of $S_i$ for $i = m - 1, m - 2, \cdots, l + 1$, then Eq. $(1)$ holds and hence, $\Gamma$ has a Hamiltonian path as shown in Eq. $(2)$. Next, we describe a simple method for finding the longest path in $\Gamma$ in the general situations, which can be reduced to the special case in Theorem 1.

Assume that Eq. $(1)$ does not hold for all $i = l + 1, \cdots, m$. Let $I = \{m, m-1, \cdots, l+1\}$. Then for $i = m, m-1, \cdots, l+2, l+1$, we check one by one whether the index $i \in I$ violates Eq. $(1)$, if it does, then $I \leftarrow I \setminus \{i\}$; otherwise, $I$ is kept unchanged. After this step is finished, let $I' = \{i_1, i_2, \cdots, i_t\}$ be the set of the remaining indices and $i_t > i_{t-1} > \cdots > i_1$. Then we have $|S_{i_t}| \geq t - j + 1$, for $j = 1, 2, \cdots, t$. Thus, if we let $\Gamma' = \Gamma[(\cup_{i \in I} v_i) \cup S_{l+1}]$ be the subgraph of $\Gamma$ induced by the nodes corresponding to the remaining indices in $I'$ and the nodes in $S_l$, then Eq. $(1)$ holds for $\Gamma'$. Thus, by the proof of Theorem 1, we can always find a Hamiltonian path $P'$ in $\Gamma'$, which in turn gives a longest path $P$ in $\Gamma$, by slightly modifying on $P'$ in the following ways:

(a) If $l + 1 \notin I'$, i.e., $l + 1$ is the last index that has been removed, and

$$P' = (v_m, \tilde{v}_m, \cdots, v_{l+2}, \tilde{v}_{l+2}, v'_1, \cdots, v'_s)$$

is the Hamiltonian path in $\Gamma'$, then the $P = (v_m, \tilde{v}_m, \cdots, v_{l+2}, \tilde{v}_{l+2}, v'_1, \cdots, v'_s, v_{l+1})$ obtained from $P'$ by adding a vertex $v_{l+1}$ is the longest path in $\Gamma$. In this case, $P$ has one more node than $P'$.

(b) If $l + 1 \in I'$, then $P = P'$.

A formal description of above algorithm is given in Algorithm 1. Fig. 5 illustrates one example which shows how this algorithm works.

**Theorem 2.** The path $P$ constructed above from $P'$ is the longest path in $\Gamma$.

**Proof.** Let $P^*$ be the longest path in $\Gamma$, and $P$ be the path computed above. We are going to show that the length of $P$ is no less than that of $P^*$. First we examine the process of removing indices that violate Eq. $(1)$. Assume that $I' = \{i_1, i_{i-1}, \cdots, i_t\}$ is the set of indices of the remaining nodes with $i_t > i_{t-1} > \cdots > i_1$. Notice that some of the indices are consecutive integers, some are not. This gives a partition of the set of all indices $I = \{m, m-1, \cdots, l+2, l+1\}$. More precisely, suppose that $I' = \bigcup_{i=i_t} V_i$, where $V_i = \{i_t, i_{t-1}, \cdots, p_i-1, p_i\} \subset I'$ is the subset of consecutive integers listed in a deceasing
Algorithm 1 Algorithm for LSP ($S = \{c_1, c_2, \ldots, c_m\}; k$)

1. Arrange $c_i$ in a non-decreasing order, say $c_1 \leq c_2 \leq \cdots \leq c_m$.
2. Construct graph $\Gamma = (V, E)$ with vertex set $V = \{v_1, v_2, \ldots, v_m\}$ and $(v_i, v_j) \in E$ if and only if $c_i + c_j \leq k$.
3. $l = \max\{|i|c_i + c_{i+1} \leq k\}$; $I = \{m, m-1, \cdots, l+1\}$.
4. for $i = m, m-1, \cdots, l+1$ do
5. $I(i) = \max\{|j|c_j \leq k\}$.
6. $S_i = \{v_j| j = 1, 2, \cdots, I(i)\}$.
7. if $|S_i| < m-i+1$ then /*if Eq. (1) is not true*/
8. $I \leftarrow I \setminus \{i\}$
9. end if
10. end for
11. Let $\Gamma' = \Gamma[(\cup_{i \in I} v_i) \cup S_{l+1}]$ be the induced subgraph
of $\Gamma$.
12. Find a Hamiltonian path $P'$ in $\Gamma'$ according to Eq. (2).
13. Modify $P'$ according to Rule (a) and (b) to obtain a path $P$ and output $P$.

Order. Let $V'_j = \{p_1 - 1, p_1 - 2, \ldots, q_{j+1} - 1\} \subset I \setminus I'$ be the subset of consecutive indices that has been removed, for $i = 1, 2, \ldots, r$. Then $I = \bigcup_{i=1}^r (V_i \cup V'_i)$, which can be depicted below, where $\times$ denotes the index that has been removed.

$$
\begin{array}{cccccc}
q_1, q_1 - 1, \ldots, p_1, & x, x, \ldots, x, q_2, q_2 - 1, \ldots, p_2, \\
V_1 & V'_1 & V_2 \\
& x, x, \ldots, x, q_r, q_r - 1, \ldots, p_r, & x, x, \ldots, x, x, \\
V'_2 & V_r & V'_2
\end{array}
$$

Assume that $p_j$ is listed as the $h_j$-th element of $I'$. Since all the indices in $V'_j = \{p_j - 1, p_j - 2, \ldots, q_j + 1\}$ have been removed, we have,

$$|S_{p_j}| \geq t-h_j+1, |S_{p_j}| < t-h_j+2, \ldots, |S_{q_j+1}| < t-h_j+2.$$  (4)

Note that $|S_{p_j}| \leq |S_{p_j}|-1$. Combining with the first two inequalities in Eq. (4) gives that $|S_{p_j}| = |S_{p_j-1}| = t-h_j+1$. Similarly, we have $|S_{p_j}| = |S_{p_j-1}| = \cdots = |S_{q_j+1}| = t-h_j+1$, which implies that $S_{p_j} = S_{p_j-1} = \cdots = S_{q_{j+1}}$.

Next, we show that length of $P$ is at least that of $P^*$. Note that $P$ includes all vertices in $S_{l+1}$. We only need to show that the number of vertices of $P$ which is contained in $\{v_i|i \in I\}$ is no less than that of $P^*$, that is, $|V(P) \cap \{v_i|i \in I\}| \geq |V(P^*) \cap \{v_i|i \in I\}|$. We distinguish the following two cases:

- **Case 1.** $V'_i \neq \emptyset$, i.e., $i + 1$ is the last index in $I$ that has been removed. By previous discussions, we have $S_{p_i} = \cdots = S_{l+1} = |I'| = t$. Let $X = \{v_i|i \in I\}, T = Y = S_{l+1}$. According to Lemma 3 below, $|V(P^*) \cap X| \leq |Y|+1$. By the construction of $P$, we have $|V(P) \cap X| = |Y|+1$. Therefore, $|V(P^*) \cap X| \leq |V(P) \cap X|$.

- **Case 2.** $V'_i = \emptyset$, i.e., $p_i = l+1 \in I'$. In this case, we have $|S_{p_{l+1}}| = \cdots = |S_{q_j+1}| = t-(q_l-p_l+1)$, and $|S_j| \geq t-(j-p_l)$ for $p_i \leq j \leq q_l$. Now, let $X = \{v_i|i \in I\}$.

![Diagram](image-url)

Fig. 5: This figure gives an example which illustrates how to find the longest in $\Gamma$ in general situation (Case 2). $S$ is the input multiset of non-negative integers. $v_i(c_i)$ denotes a node $v_i$ with cost $c_i$. There is a line segment between $v_i$ and $S_j$ which means $v_i$ is adjacent to every node in $S_j$ and node $v_i(8)$ was removed (since $|S_7| = |S_8| = 3 < 4$), which is denoted by dashed line.

$I \setminus \{q_t, q_t - 1, \cdots, p_t\}$ and $T = Y = S_{p_i-1}$. Then $N(X) = T$. According to Lemma 3 below, either $|V(P^*) \cap X| \leq |Y|$ or $|V(P^*) \cap X| \leq |Y|+1$. In the first case, we have

$$|V(P^*) \cap \{v_i|i \in I\}| \leq |Y| + (q_t - p_t + 1).$$

In the second case, the two endpoints of $P^*$ are in $X$, so we have $V(P^*) \cap \{v_i|i \in I\} \cap X = \emptyset$ and hence,

$$|V(P^*) \cap \{v_i|i \in I\}| \leq |V(P^*) \cap \{v_i|i \in I\} \cap X| \leq |Y|+1.$$

By the construction of $P$, we have $|V(P) \cap \{v_i|i \in I\} = |T| = |Y|+(q_t - p_t + 1)$. Notice that $q_t - p_t \geq 1$. Therefore, in both cases, $|V(P^*) \cap \{v_i|i \in I\} \leq |V(P) \cap \{v_i|i \in I\}|$ holds. This completes the proof.

**Lemma 3.** Let $\Gamma$ be a connected graph, and $V = X \cup Y$ be a partition of the vertex set $V$. Suppose that the induced subgraph $\Gamma[X]$ is an empty graph and $N(X) = T \subset Y$. Let $P^*$ be a path in $\Gamma$ with maximum number of nodes in $X$. Then we have:

(a) $|V(P^*) \cap X| \leq |T|+1$, and equality holds only if $P^*[X \cup T]$ is a path and has two endpoints in $X$.

(b) $|V(P^*) \cap X| \leq |T|$, and equality holds only if $P^*[X \cup T]$ is a path and has one endpoint in $X$.

**Proof.** Let $\Delta = P^*[X \cup T]$ be the subgraph of $P^*$ induced by $X \cup T$. First consider the simplest case that $\Delta$ is connected, i.e., $\Delta$ is a path. We distinguish the following cases:
• **Case 1.** The two endpoints of $\Delta$ are in $X$. Suppose that $\Delta = v_1 \rightarrow P_1 \rightarrow v_2 \rightarrow P_2 \cdots \rightarrow P_{l-1} \rightarrow v_l$, where $v_i \in X$ for $i = 1, 2, \cdots, l$, and $P_j$ is a piece of path in $T$ for $j = 1, 2, \cdots, l$. Then, $l = \#(P_1, P_2, \cdots, P_{l-1}) + 1 \leq |T| + 1$, and equality holds if each $P_j$ is a node in $T$ and $\Delta$ transverse all the nodes of $T$.

• **Case 2.** Exactly one endpoints of $\Delta$ is in $X$. Suppose that $\Delta = v_1 \rightarrow P_1 \rightarrow v_2 \rightarrow P_2 \cdots \rightarrow P_{l-1} \rightarrow v_l$, where $v_i \in X$ for $i = 1, 2, \cdots, l$, and $P_j$ is a piece of path in $T$ for $j = 1, 2, \cdots, l - 1$. Clearly, $l = \#(P_1, P_2, \cdots, P_{l-1}, P_{l}) \leq |T|$, and equality holds if each $P_j$ is a node in $T$ and $\Delta$ transverse all the nodes of $T$.

• **Case 3.** No endpoints of $\Delta$ are in $X$. It can be shown in a similar way that $|V(P^*) \cap X| \leq |T| - 1$ holds.

In general, if $\Delta$ is not connected, say, $\Delta$ consists of disjoint components $(\Delta_1, \Delta_2, \cdots, \Delta_r$ with $r > 1$. Let $V(\Delta_i) \cap X = X_i$ and $V(\Delta_i) \cap T = T_i$. Note that vertices in $\bigcup_{i=1}^{r} X_i$ are pairwise non-adjacent, each $\Delta_i$ must be connected with some $\Delta_j$ by a node $v \in T \setminus (\bigcup_{i=1}^{r} T_i)$, for $i \neq j$. Thus, for each $\Delta_i$, at least one vertex in $T_i$ is a degree one node (an endpoint). It follows from the previous discussions in Case 2 that $|X_i| \leq |T_i|$. Therefore, $|V(P^*) \cap X| = \sum_{i=1}^{r} |V(\Delta_i) \cap X| = \sum_{i=1}^{r} |X_i| \leq \sum_{i=1}^{r} |T_i| < |T|$. The last inequality follows since $\bigcup_{i=1}^{r} T_i$ is a proper subset of $T$. Thus, the lemmas follows. \qed

### 5.2 A New 4-Approximation Algorithm for MLB-HSN

In the followings, we will define two problems. Note that the first one, the maximum proper-node disjoint paths problem (MaxPN-DPP), is a mathematical formulation of our problem of interest, MLB-HSN, and therefore they are same.

**Definition 6 (Maximum proper-node disjoint paths problem (MaxPN-DPP)).** Consider a graph $G = (V, E)$ with two distinguished nodes $s$ and $t$, a positive integer $k$. A set of paths $P_1, P_2, \cdots, P_m$ from $s$ to $t$ is called ‘proper’, if there is a permutation $P_{i_1}, P_{i_2}, \cdots, P_{i_m}$ of $P_1, P_2, \cdots, P_m$ such that $l_{i_s} + l_{i_{s+1}} \leq k$ for $s = 1, 2, \cdots, m - 1$. Then, the goal of the maximum proper-node disjoint paths problem (MaxPN-DPP) is to find the maximum number of proper paths from $s$ to $t$.

**Definition 7 (Maximum node-disjoint paths with bounded average length problem (MaxPath-BALP)).** Consider a graph $G = (V, E)$ with two distinguished nodes $s$ and $t$, a positive integer $B$. Then, the goal of maximum node-disjoint paths with bounded average length problem (MaxPath-BALP) is to find the maximum number of node disjoint paths from $s$ to $t$ such that the average length is no more than a bound $B$.

Now, we shall give an efficient heuristic algorithm for MaxPN-DPP, which is a 4-approximation in an average sense. It is well-known that the problem of finding the maximum number of node disjoint paths with bounded average length is polynomial time solvable, by using minimum cost flow algorithm. Our algorithm for MaxPN-DPP is essentially based on this, a formal description of which is given in Algorithm 2.

**Algorithm 2 Approximation Algorithm for MAXPN-DPP $(G = (V,E), k)$**

1. Find the maximum number node disjoint paths from $s$ to $t$ with the average length bounded by $B := k/2$ by using the existing min-cost flow algorithm, say $P_1, P_2, \cdots, P_m$.
2. Select the paths with length $\leq k$ from $P_1, P_2, \cdots, P_m$, say $P_1', P_2', \cdots, P_m'$.
3. Let $S$ be the multi-set of integers the elements of which are length of $P_1', P_2', \cdots, P_m'$.
4. Apply Algorithm 1 for LSP with $S$ as an input to find the longest sequence.
5. Output the set of proper paths $\{P_1', P_2', \cdots, P_i\}$ whose length correspond to the numbers in the longest sequence.

**Lemma 4.** Let $OPT^* = \{P_1^*, P_2^*, \cdots, P_m^*\}$ be the optimal solution to MAXPN-DPP. $A = \{P_1', P_2', \cdots, P_m'\}$ be the results in Step 2 of Algorithm 2. Then, $m' \geq |OPT^*|/2$.

**Proof.** We distinguish the following two cases:

• **Case 1.** $m^* = |OPT^*|$ is an even number. Since $OPT^*$ is an optimal solution to MAXPN-DPP, it is also a feasible solution. Let $l^*_i$ be the length of the path $P_i^*$ for $i = 1, 2, \cdots, m^*$. Then we have:

\[
l^*_1 + l^*_2 + k, l^*_3 + l^*_4 + k, \cdots, l^*_{m^*-1} + l^*_{m^*} \leq k.
\]

Summing up the first, the third, the $(m^* - 1)$-th inequalities above gives that $l^*_1 + l^*_2 + \cdots + l^*_{m^*} \leq \frac{m^* - 1}{2} k$, i.e.,

\[
l^*_1 + l^*_2 + \cdots + l^*_{m^*} \leq \frac{k}{2}.
\]

That is, $\{P_1', P_2', \cdots, P_m'\}$ is a feasible solution to MAXPATH-BALP. Let $OPT$ be the optimal solution to MAXPATH-BALP with average bound $k/2$. Then clearly $|OPT| \geq m^* = |OPT^*|$. 

Next, we show $m' \geq |OPT|/2$ is always true. Actually, let $OPT = \{P_1, P_2, \cdots, P_m\}$ be the optimal solution to MAXPATH-BALP with average bound $B = k/2$. Then $l = l_1 + l_2 + \cdots + l_m \leq k/2$, where $l_i$ is the length of path $P_i$. Let $t'$ be the number of paths among $P_1, P_2, \cdots, P_m$ whose lengths are greater than $k$. It follows that

\[
k/2 \geq \frac{l_1 + l_2 + \cdots + l_m}{m'} > \frac{t'k}{m'}.
\]

That is, $t' < m/2$. Thus, we have $m' > m/2 = |OPT|/2 \geq |OPT^*|/2$.

• **Case 2.** $m^* = |OPT^*|$ is an odd number. Similarly, we have $l^*_1 + l^*_2 + \cdots + l^*_{m^*-1} \leq \frac{m^* - 1}{2} k$, i.e.,

\[
l^*_1 + l^*_2 + \cdots + l^*_{m^*-1} \leq \frac{k}{2}.
\]
In this section, we perform simulation to evaluate the performance of the two algorithms, the heuristic algorithm based on existing results introduced in Section 4, and our new 4-approximation algorithm introduced in Section 5. For the sake of easier understanding, in this section, we will refer each of the algorithms as HEU and 4-APP respectively. Both algorithms are implemented using Python language version 3.4.2 in Windows 8.1 64-bit operating system. Additionally, our code uses networkx library version 1.10rc2 for min-cost flow algorithm. In the simulation, we consider 100m by 100m virtual 2-D space and randomly deploy static nodes, whose count varies from 3000 to 10000 with an increment of 100. The number of mobile nodes varies from 10 to 100 with an increment of 10. We assume the sensing range is 1 and the communication range is 2. For each parameter setting, we create 100 instances and obtain an average result.

First, we fix the number of mobile nodes to 10 and study the impact of the number of static nodes to the performance of HEU and 4-APP in terms of the number of resulting hybrid barriers. Our simulation result is shown in Fig. 6. From the figure, we can learn that 4-APP clearly outperforms HEU. In our simulation, the performance of 4-APP tends to increase with higher gradient in early stage (1000 to 6000), in that 4-APP has higher utilization of mobile nodes. At the number of static nodes are 6000, the 4-app makes best result (35% higher than HEU). This is because the environment has the highest possibility to construct a sequence with a given set of mobile nodes. After 6000, the performance increases little slowly, and the two algorithm has almost same performance after 9000. Since the sufficient number of nodes are deployed, HEU can generate as much sequences as 4-APP. In our environment, 10000 is the maximum number of static nodes.

Next, we fix the number of static nodes to 3000 and study the impact of the number of mobile nodes to the performance of HEU and 4-APP in terms of the number of resulting hybrid barriers. Our simulation result is shown in Fig. 7. From the figure, we can observe that the 4-APP outperforms HEU. Especially, as the number of mobile nodes increases, the performance gap between HEU and 4-APP grows. Their performance gap gradually grows as the number of mobile nodes increases. In case if the system operates 100 mobile nodes, 4-APP generates almost 10 more sequences than HEU. This is important example to show that our scheme is more efficient in management of mobile nodes.

In conclusion, we can convincingly argue that our new strategy used to design 4-APP is high efficient to operate than HEU, and their performance gap grows as the number of mobile nodes increases and possibility of operating mobile nodes increases. This difference was maiden by how many mobile nodes can be used to generate a sequence. In case of HEU, it only considers a sequence which is not exceed a half of mobile nodes, whereas 4-APP considers every sequence which is not exceed the number of mobile nodes. Therefore, 4-APP has high efficiency especially in terms of operating mo-

![Fig. 6: The figure shows 4-APP outperforms HEU independent from the number of static sensor nodes.](image-url)
Fig. 7: The figure shows 4-APP outperforms HEU regardless from the number of mobile sensor nodes.

7 CONCLUDING REMARKS

In this paper, we introduce MLB-HSN, an interesting scheduling problem in hybrid sensor network whose goal is to maximize the lifetime of the barrier-coverage offered by the hybrid sensor network. Significantly different from the most existing maximum lifetime barrier coverage problems in which sensors are static, MLB-HSN is distinguished by the fact that a mobile node may assist more than one sensor barrier. However, due to the fact that each mobile sensor incurs a delay for relocation, no mobile node may be used for two barriers which will be successively used. This means that the problem of how to organize the static and mobile sensors and the problem of how to organize the computed barriers are co-related, which makes the problem of our interest extremely difficult. To overcome the daunting challenge, we first design a scheduling algorithm for a given set of hybrid sensors such that no mobile node may be used for two barriers. Then, we use this algorithm to design an efficient algorithm for MLB-HSN with some meaningful performance guarantee.

Future Research Directions. During the course of our study, we managed to design an algorithm for MLB-HSN, but was not able to prove its complexity. Our conjecture is that MLB-HSN is NP-hard and we will continue our investigation on this. We will also strive to design a new algorithm for MLB-HSN with better performance guarantee. We also notice that our two new observations which provided the motivation for MLB-HSN might be applicable for the other hybrid sensor network scheduling problems and thus plan to look for more applications of the observations.

The current version of the algorithm does not provide any fault-tolerance but solely focuses on maximizing the lifetime of barrier coverage. To provide fault-tolerance to our maximum barrier coverage problem, one can try to compute the maximum number $q$ of static sensor node disjoint barriers such that the $q$ sensor barriers can be divided into a sequence of $p$ subset of barriers, $B = B_1, B_2, \ldots, B_p$ such that each $B_i$ includes $\lfloor \frac{q}{p} \rfloor$ barriers which does not share any mobile node in common and at the same time no two consecutive subsets $B_i$ and $B_j$ share any mobile node in common. Once such $B$ is constructed, we can adopt the subset of barriers in $B$ in the order of appearance in the sequence. In this way, any failure of less than $\lfloor \frac{q}{p} \rfloor$ static nodes in $B_i \in B$ can be amended. On the other hand, the failure of mobile nodes is more serious. On straightforward solution for this issue is that the algorithm has to be executed again with available static sensors and mobile nodes to obtain new schedule $B'$ as $B$ cannot be executed anymore. However, any solution which is better than this straightforward is open.

ACKNOWLEDGEMENT

This work was supported in part by US National Science Foundation (NSF) CREST No. HRD-1345219. This research was jointly supported by National Natural Science Foundation of China under grants 11471005. This work was also supported by National Research Foundation grant (MSIP No. 2013R1A2A20101400) and Basic Science Research Program (MEST 2011-0012216) funded by the Korea government.

REFERENCES


Weili Wu received the PhD degree in computer science from the University of Minnesota in 2002. Her main research interest is in database systems. She has produced a number of research papers in spatial data mining, distributed database systems, and algorithm design. She was also a co-editor of a book on clustering and information retrieval. She is currently a professor in the Department of Computer Science at the University of Texas at Dallas. She is a member of the IEEE.

Wonjun Lee received B.S. and M.S. degrees in computer engineering from Seoul National University, Seoul, Korea, in 1989 and 1991, respectively; and his Ph.D. degree in computer science and engineering from the University of Minnesota, Minneapolis, Minnesota, USA, in 1999. In 2002, he joined the faculty of Korea University, Seoul, where he is currently a Professor with the School of Information Security. He has authored or co-authored over 180 papers in refereed international journals and conferences. His research interests include communication and network protocols, optimization techniques in wireless communication and networking, security and privacy in mobile networking and computing, and radio frequency powered computing and networking. He is a senior member of IEEE.

Alade O. Tokuta received the BS and MS degrees in electrical engineering from Duke University, Durham, the EE degree in electrical engineering from Columbia University, NY, and the PhD degree in electrical engineering and computer science from University of Florida, Gainesville. Currently, he is a professor in the Department of Mathematics and Physics at the North Carolina Central University, Durham, USA. His research interests include robotics; computer image synthesis/robot; networking, and algorithm design.