Wireless Netw (2015) 21:1683–1698
DOI 10.1007/s11276-014-0873-0

You can't get there from here: sensor scheduling with refocusing delays

Yosef Alayev · Amotz Bar-Noy · Matthew P. Johnson · Lance Kaplan · Thomas F. La Porta

Published online: 18 December 2014

Abstract We study a problem in which a single sensor is scheduled to observe sites periodically, motivated by applications in which the goal is to maintain up-to-date readings for all the observed sites. In the existing literature, it is typically assumed that the time for a sensor switching from one site to another is negligible. This may not be the case in applications such as camera surveillance of a border, however, in which the camera takes time to pan and tilt to refocus itself to a new geographical location. We formulate a problem with constraints modeling refocusing delays. We prove the problem to be NP-hard and then study a special case in which refocusing is proportional to some Euclidian metric. We give a lower bound on the optimal cost for the scheduling problem, and we derive exact solutions for some special cases of the problem. Finally, we provide and experimentally evaluate several heuristic algorithms, some of which are based on the computed lower bound, for the setting of one sensor and many sites.

Keywords Sensor scheduling · Delay constraints · Sensor networks · Sensors · Surveillance · Resource allocation · Algorithms

Conference version of the paper appeared in IEEE MASS'10.

Y. Alayev (✉)
Computer Science, The Graduate Center, CUNY,
New York, NY, USA
e-mail: yosef.alayev@gmail.com

A. Bar-Noy
Computer and Information Science, Brooklyn College
and the Graduate Center, CUNY, New York, NY, USA
e-mail: amotz@sci.brooklyn.cuny.edu

M. P. Johnson
Computer Science, Lehman College and the Graduate Center,
CUNY, New York, NY, USA
e-mail: mpjohnson@gmail.com

L. Kaplan
Sensors and Electron Device Directorate, U.S. Army Research
Lab, Adelphi, MD, USA
e-mail: lkaplan@arl.army.mil

T. F. La Porta
Computer Science and Engineering, Penn State University,
State College, PA, USA
e-mail: tlp@cse.psu.edu

1 Introduction

Area monitoring is a common application of wireless sensor networks. In wireless sensor networks employed in monitoring or surveillance applications, individual sensors may perform pre-assigned or on-demand tasks. In particular, sensors such as visual cameras, radars, or passive infrared cameras may need to observe distinct geographical locations (or sites). It is usually the case that the number of tasks (or observations) to perform is larger than the number of sensors in the network. Hence, sensors' time must be shared to observe multiple sites. In addition, some observations may be more important than others. Thus, a schedule is required in order to specify which sensor observes what site at each particular time. This paper studies a problem of scheduling a single sensor to observe n distinct sites. Our work is motivated by the research of Yavuz and Jeffcoat [1, 2].

More formally, in the sensor scheduling problem we have m sensors (cameras, radars, PIRs, etc.) that need to observe $n > m$ distinct locations. The objective is to schedule the sensors to observe the most important sites as frequently as possible, in order to minimize the amount, or value, of information we fail obtain when particular sites are *not* observed. Applications of this kind of problem are common. For instance, camera surveillance may be used to monitor intrusions along a border, in which case there may be many distinct unprotected places along the border to observe, to protect against use by intruders. If it is too costly to dedicate a single camera to each site, then one or more cameras may be scheduled to alternate between observation of different locations.

Consider an example in which a sensor is required to observe sites A, B, and C. The sensor can only focus on one site at a time, say site A, collecting all the new potential information from that site. The other two sites B and C are left unobserved, in which case we must rely on earlier observations of them. Each time a site goes unobserved, we lose some new information. As in [1] and [4], two types of costs can be associated (within a given timeslot) with this lost information: (1) a *fixed cost* of not observing a particular site at a particular time, and (2) a *variable cost* that is a linear function of the time gap since its last visitation. Both costs provide an incentive to visit the site. The motivation of the second type of cost is to model the fact that information becomes increasingly obsolete over time. The goal is to construct a periodic schedule (i.e. a schedule that repeats itself every $T \in \mathbf{Z}^+$ timeslots), that minimizes the average cost of lost information per timeslot.

The crucial aspect of this paper's problem formulation is that sometimes a sensor may not be able to observe two distinct sites right after each other as it needs time to adjust and refocus. Prior research has neglected such time delays. Though time delay to refocus the sensor to a new location has a considerable impact on a feasible optimal schedule. Refocusing time delay may preclude many sites from being scheduled next. Suppose that in the example above it takes one idle timeslot to refocus the sensor from one site to another. Say a sensor is observing site A during the last timeslot. What should a sensor do during the next timeslot? Observing site B or C is not possible since it takes time for a sensor to refocus. The sensor may either continue observing A in the next timeslot or not observe anything in the next timeslot, in which case it takes one timeslot to refocus to another site in order to observe it in the second timeslot after the idle timeslot.

Throughout the rest of the paper we will refer to a sensor as a camera since camera is a typical sensor used for observations. Conserving the energy required for movement could be a motivation as well, although we defer optimizing for energy to future work. The rest of this paper is organized as follows. Section 2 discusses some related work. In Sect. 3, we formulate different variants of a single sensor scheduling problem with delay constraints and prove some hardness and structural properties. In Sect. 4, we derive a lower bound on the optimal solution cost for the sensor scheduling problem. We derive exact solutions to some special case settings in Sect. 5. Then in Sects. 6 and 7 we propose greedy-based algorithms and evaluate them on synthetic data. We conclude by discussing future work in Sect. 8.

2 Related work

A min–max variant of the Single-Sensor Scheduling Problem (SSSP) is studied by practitioners in [1] and [2]. In [1], a formulation minimizing the maximum information loss for any site and in any timeslot is studied; it is NP-hard, and heuristic algorithms are given. In that formulation, it is assumed that time transition between sites is negligible. Our relaxation of this assumption is a crucial aspect of the problem we study in the present paper. Because of the delays between observations of various site pairs, only a subset of sites in general are reachable in the next timeslot at any given time, which separates our problem from the previously studied Broadcast Disks and Maintenance problems. Other works on either single sensor scheduling or multi-sensor scheduling can be found in [3–6], and [7].

The min–sum sensor scheduling problem that we define here is a generalization of the Broadcast Disks [8] problem. In that problem, there is a quadratic cost paid (in total) for time gaps between receiving pages, and the objective is to minimize the sum of costs. In our problem, there are quadratic costs for gaps (between observations of sites) as well as *linear* costs for gaps. (Equivalently, during each timeslot within a site's gap, there is one penalty linear in the time since the gap's last observation, as well as one penalty that is a simple constant.) The Broadcast Disks problem does not consider time delay constraints that may preclude many pages from being scheduled next.

An equivalent problem of Broadcast Disks is the problem of scheduling for Teletext systems [9, 10]. The single-sensor (SSSP) special case of the sensor scheduling problem that we focus on in this paper, in which there is only one sensor scheduled to observe n sites, generalizes the Teletext problem, in the same two ways as above, viz., by adding fixed costs and delay constraints. In [10], Ammar and Wong show that there always exist optimal Teletext schedule solutions that are periodic; in [9], they show how to construct broadcast cycles. They also derive a square root rule according to which an item i's broadcasting frequency is proportional to the square root of i's request probability, which in our case corresponds roughly to normalized variable costs.

Also closely related is the Machine Maintenance problem [11]. In that problem, which is another generalization of Broadcast Disks, a (linear) operating cost is charged based on the time elapsed since the last service, while a constant maintenance cost is charged for each timeslot t when the machine *is* serviced. Note that the maintenance cost is charged in exactly the complement of the timeslots when our fixed cost is, whereas the operating cost corresponds to our variable cost. (There are no switching delays in the Maintenance problem.) Bar-Noy et al. [11] prove that the maintenance scheduling problem is NP-hard and provide approximation algorithms. Note that hardness for the single-sensor version of our problem does not immediately follow from this result. We give an unrelated hardness proof in the paper.

More broadly, various other scheduling problems involve the notion of travel delays between sites. In hard disk scheduling [12], e.g., the Shortest Seek Time First algorithm chooses the request closest to the current head position, in order to minimizes latency. In the k-server problem [13], k servers will service client requests, as they appear online, within a metric space. The goal is to minimize the total distance moved by the servers servicing requests. In the related offline Watchman problem [14], one server must choose a short route between a set of locations to guard, avoiding obstacles in the region. The SSSP differs from these problems, however, in that the switching delays are hard constraints rather than soft. That is, these delays rule out certain sequences of site observations as infeasible. We seek low-cost periodic schedules observing these restrictions.

Another related work [15] studies a problem where a single camera is required to observe multiple people. It does so by dividing the camera's time in order to view everyone. In their work an additional tracking camera with a fixed wide field of view is employed to detect, track, and classify moving targets. The information of the targets is then provided to the active camera that can pan, tilt, and zoom to collect high resolution video. The scheduling problem consists of deciding which person the active camera will focus its sensing resources on until the next state update. The arrival times of people entering the scene is not known in advance making it an online scheduling problem.

The work in [16] is an extension of [15] to multiple active cameras scheduling. This work studies a system of multiple pan-tilt-zoom (PTZ) cameras, assisted by a fixed wide-angle camera, to collect high resolution video of the (many) moving targets. The system first assigns a subset of the requests or targets to each camera. The cameras then select the parameter settings that best satisfy the assigned competing requests to provide high resolution views of the moving objects. The main difference in our paper is that the targets are not moving objects but stationary geographical locations with associated costs due to lost information. Other works on controlling PTZ cameras to optimize coverage can be found in [17, 18], and [19]. For a survey on modeling coverage in camera networks refer to [20].

3 Model

In the SSSP, we have one sensor that observes a collection of n sites at discrete time intervals. In each timeslot, the sensor can choose (at most) one site to observe. The problem is to find a periodic schedule (with some period T), minimizing total costs, as described below. Initially we will assume that the time for switching from one site to another is negligible. Later we will incorporate this delay into the model.

3.1 Preliminaries

Our model uses the following notation:

- a_i—fixed penalty for not observing site i
- $b_{i,t}$—variable penalty for not observing site i at time t
- $y_{i,t}$—time of last observing site i before time moment t, set to t iff the sensor is observing site i at time slot t, and otherwise equals $y_{i,t-1}$
- $g_{i,t} = (t - y_{i,t})$—the time length (or gap) since last observation of site i, prior to time t

Let $x_{i,t}$ be a decision variable taking value 1 if the ith site is observed at time t ($1 \leq t \leq T$), and 0 otherwise. The penalty for not observing a site i at time t is expressed as follows:

$$a_i(1 - x_{i,t}) + b_{i,t}(t - y_{i,t})$$

A variant of the sensor scheduling problem was formulated for one-sensor case in [1] and [2]. In their formulation, the objective was to minimize the maximum cost $a_i(1 - x_{i,t}) + b_{i,t}(t - y_{i,t})$, among all sites i *and* times t, for a schedule defined over period T. The factor $(t - y_{i,t})$ is the gap $g_{i,t}$, the length of time since the last observation of site i. Following the Broadcast Disks and Maintenance problems, however, we optimize for the average (or equivalently, the total) penalty per slot, over all sites i and times t. The parameter $b_{i,t}$ could be tuned based on the needs of the application, e.g. increased during rush hours or decreased otherwise if the activity level of site i at time t diminishes. For the bulk of the paper, however, we will assume for simplicity that $b_{i,t}$ as a time-invariant parameter b_i.

Thus, our problem is specified by the integer program formulation given in Table 1.

Table 1 IP formulation for SSSP

min	$\frac{1}{T}\sum_{i=1}^{n}\sum_{t=1}^{T} a_i(1-x_{i,t}) + b_i(t-y_{i,t})$		
s.t.	$\sum_{i=1}^{n} x_{i,t} \le 1$	$\forall_{t\in\{1,\dots,T\}}$	(1)
	$0 \le y_{i,t} - y_{i,t-1} \le t \cdot x_{i,t}$	$\forall_{i\in\{1,\dots,n\}}$ $\forall_{t\in\{1,\dots,T\}}$	(2)
	$tx_{i,t} \le y_{i,t} \le t$	$\forall_{i\in\{1,\dots,n\}}$ $\forall_{t\in\{1,\dots,T\}}$	(3)
	$x_{i,0} = 1$	$\forall_{i\in\{1,\dots,n\}}$	(4)
	$y_{i,0} = 0$	$\forall_{i\in\{1,\dots,n\}}$	(5)
	$x_{i,t} \in \{0,1\}$	$\forall_{i\in\{1,\dots,n\}}$ $\forall_{t\in\{1,\dots,T\}}$	(6)
	$d_{i,j}x_{i,t} \le \tau + (1-x_{j,t-1})\Delta$	$\forall_{i\in\{1,\dots,n\}}$ $\forall_{j\in\{1,\dots,n\}}$ $\forall_{t\in\{1,\dots,T\}}$	(7)

The first constraint prevents multiple sites from being observed at the same time. The second and third constraints of the formulation above ensure that $y_{i,t}$ takes on value t if an observation of site i occurs at time t, and otherwise equals $y_{i,t-1}$. The fourth and fifth constraints dictate that all sites are treated as having been observed at time 0. In some of our experiments, we relax these constraints and allow $x_{i,0}$ and $y_{i,0}$ to be specified in the input, which allows us to code for certain sites having initially already been unobserved for some larger number of timeslots.

The sixth constraint restrict the $x_{i,t}$ variables to 0 or 1 values. We do not restrict the values of the $y_{i,t}$ variables, allowing them to take on negative values when the $y_{i,0}$ values are specified as described above. The $y_{i,t}$ will, however, always take on integer values due to the second and third constraints, and so a constraint of the form $\mathbf{y_{i,t}} \in \mathbf{Z}$ is superfluous. The seventh constrain is a delay constraint which is explained in the next subsection.

3.2 Delay constraints

Let $\mathcal{D}$ be an $n \times n$ matrix with entries $d_{i,j}$ corresponding to transition times (in units of timeslots) between sites j and i. ($d_{i,j} = 0$ indicates that the sensor can observe site i after observing site j, without requiring any idle slots.) We can then formulate the constraint as:

$$d_{i,j}x_{i,t} \le (t - y_{j,t-1} - 1) \tag{8}$$

Recall that $(t-1) - y_{j,t-1}$ is the value of gap $g_{j,t-1}$, which is the length of time since the last observation of site j prior to time $t-1$. The right hand side is nonnegative since the gap cannot be negative. If i is not scheduled at time t, then the constraint is satisfied trivially because the LHS is 0; if it is, then the LHS is the transition time from the previously scheduled site j to the current site i. Note that for the previously scheduled j, the RHS is smaller than for the j that was not scheduled previously. Consider the following example, in which we have sites A, B, C, and D, and transition matrix $\mathcal{D}$ as shown in Table 2.

Assume the schedule up to now has been $ABCA$. Now we need to make a decision for the next timeslot, $t = 5$. Note that the previously scheduled site j at time $t-1$ is A, and that $g_{j,t-1} = (t-1) - y_{j,t-1} = 4 - y_{A,4} = 0$. For any other j the gap will be even greater, and so therefore will the right hand side. Thus the RHS of the constraint Eq. 8 is 0. This means that on the LHS we must have 0 to satisfy the constraint. Hence, only those sites in row A with 0 entries can be considered (i.e., A and B) for timeslot $t = 5$. Now, suppose we leave $t = 5$ idle. This would give us schedule $ABCA\square$ (where $\square$ indicates no site observed), with our next scheduling decision for timeslot $t = 6$. Note that the (unique) minimum $g_{j,t-1} = (t-1) - y_{j,t-1}$ is $5 - y_{A,5} = 1$. Therefore the RHS is 1. The LHS should have a site scheduled at timeslot $t = 6$ that is reachable in time at most 1 (i.e., A, B, or C). Similarly, if we leave timeslot $t = 6$ idle with the current schedule so far as $ABCA\square\square$, then for $t = 7$ the (unique) minimum $g_{j,t-1}$ is due to A and is equal to 2. Hence, at timeslot $t = 7$ we can schedule any site reachable from A within 2 timeslots (i.e., A, B, C, or D).

In most of this paper we restrict our attention to a special case in which the entries $d_{i,j}$ are 0 or 1, meaning the sites are reachable within at most one idle slot. The constraint can now be formulated as:

$$d_{i,j}x_{i,t} \le 1 - x_{j,t-1}$$

If, however, we let the $d_{i,j}$ be the time delay parameters themselves (rather than units of timeslots) associated with the refocusing of the sensor from site j to site i, set $\Delta = \max_{i,j} d_{i,j}$, and furthermore let τ be the time limit during which the sensor should finish its refocus. Then we can reformulate the constraint to incorporate the pan and tilt delay as: $d_{i,j}x_{i,t} \le \tau + (1 - x_{j,t-1})\Delta$. Both constraints are equivalent: one expresses the delay in terms of number of slots, and the other expresses it in terms of time.

Table 2 Transition matrix $\mathcal{D}$

	A	B	C	D
A	0	0	1	2
B	0	0	0	1
C	1	0	0	0
D	2	1	0	0

3.3 Hardness and periodicity

We prove hardness by means of a Cook reduction from the classical Maximum Independent Set (MIS) problem, which is NP-hard [21].

Theorem 1 *Solving SSSP optimally is NP-hard.*

Proof Given an instance $G = (V, E)$ of MIS, we create a family of SSSP instances, indexed by value T ranging from 1 to $n = |V|$, as follows. For each node $v_i \in V$, we introduce a site i, with constant cost $a_i = 0$ and linear cost $b_i = 1$. For each edge $(v_i, v_j) \in E$, we set the delay between nodes i and j to be infinity (or some sufficiently large constant); for each edge not present, we set the corresponding delay to zero. Assume we have solved all n SSSP instances optimally.

We distinguish between three possible situations that a given site may be in, within a particular problem solution: it may be observed zero times, one time, or multiple times. The difference in cost between zero observations and one observation is much greater than the difference between one observation and multiple observations: zero observations will incur a cost quadratic in both T and the number of times the schedule cycles; one observation will incur a cost at most quadratic in T and linear in the number of cycles. The bulk of the solution cost will depend on the number of sites not observed at all.

We note that for any value T over the specified range, there will exist a feasible solution visiting a site in every timeslot: assuming the delay between a site and itself is zero, a schedule observing the same site in every timeslot will always be feasible. By the argument above, however, sites will only be observed multiple times when it is impossible to add some other zero-visited site to the schedule. For a sufficiently small value of T, therefore, we will obtain a schedule in which all sites observed are observed only once. It is clear from the construction that the sites observed in the SSSP optimal solution of the instance with the largest such value T will form a maximum independent set in the underlying graph. □

We also note that we can restrict our attention to periodic schedules in the following sense, assuming that delay constraints are symmetric, i.e., the delay for moving from site i to j is the same as for moving from j to i.

Theorem 2 *Every SSSP problem instance with symmetric delay constraints admits an optimal solution that is periodic.*

Proof (sketch) As with the case of the Maintenance Problem [11], a proof can be given by adapting the argument of Anily et al. [22]. The proof begins by bounding from above the distance between two consecutive observations of any given site in an optimal solution; indeed, if there is a longer interval between the two observations, then the schedule could be improved by inserting a third, intervening observation of the site. Since there are then only a finite number of possible *states*, any schedule can be transformed into a periodic schedule at least as good. □

4 Lower bound

In this section we derive a lower bound on the optimal solution cost for the single sensor scheduling problem. Assume that the schedule is perfectly periodic, that is, each site i is visited periodically with a fixed period τ_i. (We will show later that this assumption is justified.) We ignore for now the transition delay matrix $\mathcal{D}$ whose entries $d_{i,j}$ dictate schedulability of the sites, given the previously scheduled site. As in [11], we give a nonlinear relaxation to the sensors scheduling problem. Since the introduction of delay constraints only increases the optimal solution cost, the lower bound still holds (albeit less tightly) when delays are present.

Proposition 1 *Suppose the site i is scheduled at timeslot t and is not scheduled in timeslots $t+1, \ldots, t+x-1$. Then the total variable cost incurred by scheduling site i for the x timeslots $t, \ldots, t+x-1$ is given by $(b_i/2)(x-1)x$.*

Proof Site i incurs a variable cost of $b_i \cdot j$ in timeslots $t+j$, for $j = 0, \ldots, x-1$, so the total variable cost incurred by site i in timeslots $t, \ldots, t+x-1$ is given by:

$$\sum_{j=0}^{x-1} b_i \cdot j = \frac{b_i}{2}(x-1)(x)$$

□

Proposition 2 *Suppose the site i is scheduled in timeslot t and is not scheduled in timeslots $t+1, \ldots, t+x-1$. Then the total fixed cost incurred by scheduling site i for the x timeslots $t, \ldots, t+x-1$ is given by $(x-1)a_i$.*

Proof Site i incurs a fixed cost of a_i in timeslots $t+j$, for $j = 1, \ldots, x-1$, and 0 cost in timeslot t, so the total fixed cost incurred by site i in timeslots $t, \ldots, t+x-1$ is:

$$0 + \sum_{j=1}^{x-1} a_i = (x-1)(a_i)$$

□

The next proposition shows that the lower bound schedule is indeed a perfectly periodic schedule.

Proposition 3 *Lower bound schedule is* perfectly *periodic.*

Proof Given is a schedule with time horizon T. Suppose site i is scheduled at time t and not scheduled at times $t+1,\ldots,t+x-1$, then again is scheduled at time $t+x$, and not scheduled at times $t+x+1,\ldots,t+x+y-1$. In other words, the schedule consists of site i, followed by a skip of $x-1$, followed by site i, followed by a skip of $y-1$, so that there are two periods: x and y. We will show that on the time horizon T, either $x=y$ or a better schedule exist where the period $z=\frac{x+y}{2}$. If $x=y$, then the costs of the first x slots and of the second y slots are the same, and thus is perfectly periodic, so assume otherwise. That is, when $x \neq y$, we have $x^2+y^2-2xy=(x-y)(x-y)>0$. Multiplying both sides by b and grouping terms, we get

$$(2bx^2-bx^2)+(2by^2-by^2)+(2bx-2bx)+(2by-2by)-2bxy>0$$

Dividing both sides by $4(x+y)$ yields

$$\frac{bx^2-bx+by^2-by}{2(x+y)}-\frac{bx}{4}-\frac{by}{4}+\frac{b}{2}>0$$

Rearranging the terms, we get

$$\frac{b(x^2-x+y^2-y)}{2(x+y)}>\frac{bx+by}{4}-\frac{b}{2}$$

Similarly,

$$\frac{b\left(\frac{x(x-1)}{2}+\frac{y(y-1)}{2}\right)}{x+y}>\frac{bx+by}{4}-\frac{b}{2}$$

The LHS is the cost associated with b for non-regular schedule. The RHS is the cost associated with b for a regular schedule with $z=\frac{(x+y)}{2}$. The costs associated with a for both schedules on the time horizon T are the same. Thus the regular schedule indeed has lower cost, which completes the proof. □

Now consider the following nonlinear program with variables $\tau_1,\ldots,\tau_n$.

$$\begin{aligned} \min:\quad & \sum_{i=1}^{n}\frac{b_i(\tau_i-1)}{2}+\sum_{i=1}^{n}\left(a_i-\frac{a_i}{\tau_i}\right) \\ \text{s.t.}:\quad & \sum_{i=1}^{n}\frac{1}{\tau_i}\le 1 \\ & \tau_i\ge 1 \quad \forall i \end{aligned} \tag{9}$$

Note that the average cost of the schedule as time goes to infinity is equivalent to average cost over the period, and thus the total cost over the period $\frac{b_i(\tau_i-1)\tau_i}{2}+a_i(\tau_i-1)$ is being divided by the period τ_i. In the nonlinear program we schedule each site in fixed periods such that the average cost per slot is minimized. This is a relaxation because these periods may not be integers. Furthermore, even if we round these periods to integers, the schedule may not be achievable simultaneously for all the sites, since more the one site will need to be scheduled in same timeslot for some timeslots. Since the a_i terms in the second summation are constants, we can change the objective function to:

$$\begin{aligned} \min:\quad & \sum_{i=1}^{n}\frac{b_i(\tau_i-1)}{2}-\sum_{i=1}^{n}\frac{a_i}{\tau_i} \\ \text{s.t.}:\quad & \sum_{i=1}^{n}\frac{1}{\tau_i}\le 1 \\ & \tau_i\ge 1 \quad \forall i \end{aligned} \tag{10}$$

The objective function is concave with convex constraints. We can use Lagrangian relaxation to solve for optimal $\tau_1,\ldots,\tau_n$.

Theorem 3 *An optimal solution to the nonlinear relaxation is given by* $\tau_i=\sqrt{\frac{2(\lambda^*-a_i)}{b_i}}$, *where* $\lambda^*>\max_i(a_i)$ *and* $\lambda^*>\frac{b_i+2a_i}{2}$ $(1\le i\le n)$.

Proof We obtain the following nonlinear program by applying Lagrangian relaxation to Eq. 10.

$$\min:\sum_{i=1}^{n}\frac{b_i(\tau_i-1)}{2}-\sum_{i=1}^{n}\frac{a_i}{\tau_i}-\lambda\left(1-\sum_{i=1}^{n}\frac{1}{\tau_i}\right)-\sum_{i=1}^{n}\mu_i(\tau_i-1) \tag{11}$$

For any fixed $\lambda\ge 0$ and $\mu_i\ge 0$ for $1\le i\le n$, the optimal value of the Lagrangian relaxation 11 is a lower bound on the optimal value of the nonlinear program 10, and the optimal solution of relaxation 11 is given by $\tau_i=\sqrt{\frac{2(\lambda-a_i)}{b_i-2\mu_i}}$ $(1\le i\le n)$, provided that $b_i-2\mu_i>0$ (by taking partial derivatives of the Lagrangian with respect to τ_i).

In order to find global minima, we need to satisfy Karush–Kuhn–Tucker conditions. The constraints could either be loose or tight. If the constraints are tight then the corresponding Lagrange multipliers will be positive, whereas if the constraints are loose then the corresponding Lagrange multipliers will be 0. None of the constraints of $\tau_i\ge 1$ $(1\le i\le n)$ can be tight, since if at least one $\tau_i=1$ then the constraint of $\sum_{i=1}^{n}\frac{1}{\tau_i}\le 1$ will only be satisfied when all the other $\tau_j=\infty$ $(j\neq i)$. Thus, all the constraints $\tau_i\ge 1$ $(1\le i\le n)$ are loose (i.e., $\tau_i>1$ $(1\le i\le n)$) and, hence, all $\mu_i=0$ $(1\le i\le n)$. On the other hand the constraint $\sum_{i=1}^{n}\frac{1}{\tau_i}\le 1$ is tight, and hence $\lambda>0$. Let λ^* be the value of λ. In fact, since $\mu_i=0$, in order for $\tau_i=\sqrt{\frac{2(\lambda-a_i)}{b_i-2\mu_i}}$ $(1\le i\le n)$ to have a real solution, λ^* must be at least $\max_i\{a_i\}$. In addition, in order to satisfy the loose constraints of $\tau>1$ we need $\sqrt{\frac{2(\lambda-a_i)}{b_i}}>1$, which is equivalent to $\lambda^*>\frac{b_i+2a_i}{2}$ $(1\le i\le n)$. □

In order to find such λ^* we can solve an equation $\sum_{i=1}^{n}\sqrt{\frac{b_i}{2(\lambda-a_i)}}=1$. This equation can be solved using some variation of Newton's method. Let $q_i=\frac{1}{\tau_i}$. Observe that if we let all a_i be the same and equal to say a, then $\lambda^*=\frac{\left(\sum_{i=1}^{n}\sqrt{b_i}\right)^2+2a}{2}$. Furthermore, $q_i=\frac{1}{\tau_i}=\frac{\sqrt{b_i}}{\sum_{i=1}^{n}\sqrt{b_i}}$. We see that in the optimal solution with all a_i equal, $\frac{q_i}{q_j}=\frac{\sqrt{b_i}}{\sqrt{b_j}}$, as expected.

5 Special cases

In general, the problem is NP-hard, so the best we can hope for is approximation algorithms. The problem can be solved optimally, though, for some special cases. The case of one sensor and one site is trivial since the optimal schedule is to observe the site perpetually, in which case there is trivially zero information loss. However, the case of a sensor scheduled to observe two distinct sites is interesting. Should the sensor switch between the site observations or not? If the sensor is required to switch between the site observations, then in what pattern must it do so in order to minimize the cost?

Let us first consider a case where variable costs are zero.

Proposition 4 *Suppose we have two sites S_1 and S_2 with refocus delays $d_{1,2}$ and $d_{2,1}$, no variable costs (i.e. $b_1=b_2=0$), and fixed costs a_1 and a_2, with, say, $a_1\geq a_2$. Then an optimal schedule is to perpetually observe site S_1.*

Proof Observing site S_1 in a timeslot will incur cost a_2 per slot. Observing site S_2 in a timeslot will incur cost a_1 per slot. Not observing anything in a timeslot will incur cost of a_1+a_2 per slot. Since $a_2\leq a_1\leq(a_1+a_2)$, the minimum cost is obtained by observing S_1 in every timeslot. □

The result is easily extendable to a case with n sites and no variable costs, in which case the optimal schedule is still to observe a site with the biggest fixed cost.

Let us now consider a case where variable costs are not zero.

Proposition 5 *Suppose we have two sites S_1 and S_2 with refocus delays $d_{1,2}$ and $d_{2,1}$, variable costs $b_1\neq 0$ and $b_2\neq 0$, respectively, and fixed costs a_1 and a_2 respectively. Then an optimal schedule is of the form:*

$$((S_1)^x(\square)^{d_{2,1}}(S_2)^y(\square)^{d_{1,2}})^*$$

That is to say that the schedule is periodic with period $x+d_{2,1}+y+d_{1,2}$, where x and y are positive integers. The schedule observes site S_1 for x timeslots, followed by $d_{2,1}$ idle timeslots, followed by observation of site S_2 for y timeslots, followed by $d_{1,2}$ idle timeslots.

Proof Assume to contrary that the schedule is to only observe one site, say S_1. Then the cost of not observing S_2 grows quadratically while the size of schedule grows linearly. So the cost associated with not observing S_2 is quadratic function divided by linear function. Thus, as schedule size goes to infinity the schedule cost per slot goes to infinity. Therefore, S_2 must be observed at some point. Similarly, if we only observe S_2 and not S_1, the schedule cost per slot goes to infinity as well. Therefore, S_1 must be observed at some point as well. The only way to observe both site is to switch between them which require idle slots. Therefore, the schedule is of the form as described above, that is, observe S_1 for some x timeslots where x is a positive integer. Then the sensor transitions to site S_2, which requires $d_{2,1}$ idle timeslots in the schedule. Then the sensor observes site S_2 for some y timeslots and transitions back to S_1, which requires another $d_{1,2}$ idle timeslots in the schedule. □

The values of x and y depend on parameters a_1, a_2, b_1, b_2, $d_{1,2}$, and $d_{2,1}$. In order to choose optimal values of x and y, we set up a multivariate function $\mathcal{C}(x,y)$ to minimize, which depends on x and y and treats the rest of the parameters as constants. Let $D=d_{1,2}+d_{2,1}$. Solve the following minimization problem to find x and y.

$$\begin{aligned}\min\quad & \mathcal{C}(x,y)=\frac{a_1\cdot(D+y)+a_2\cdot(D+x)}{D+x+y}+\\ & +\frac{\frac{b_1}{2}\cdot((D+y+1)\cdot(D+y))+\frac{b_2}{2}\cdot((D+x+1)\cdot(D+x))}{D+x+y}\\ \text{s.t.}\quad & x\in\mathbf{Z}^+, y\in\mathbf{Z}^+\end{aligned} \tag{12}$$

Example

Problem: Given are two sites S_1, S_2 with the following costs: $a_1=8$, $a_2=2$, $b_1=1$, $b_2=2$. Let $d_{i,j}=d_{j,i}=2$, so that transitioning between the sites takes two idle timeslots. Find x and y that minimizes the cost $\mathcal{C}(x,y)$.

Solution: We need to minimize the following function. $\mathcal{C}(x,y)=\frac{8(4+y)+2(4+x)}{4+x+y}+\frac{(1/2)(5+y)(4+y)+(5+x)(4+x))}{4+x+y}$. (See Fig. 1.) This can be done using a tool Mathematica (i.e. $c=\mathcal{C}(x,y)$;

Find Minimum$[\{c,x\geq 1\ \&\&\ y\geq 1\},\{x,y\}]$). The critical point is at $x_c=2.16435$ and $y_c=2.8287$ with

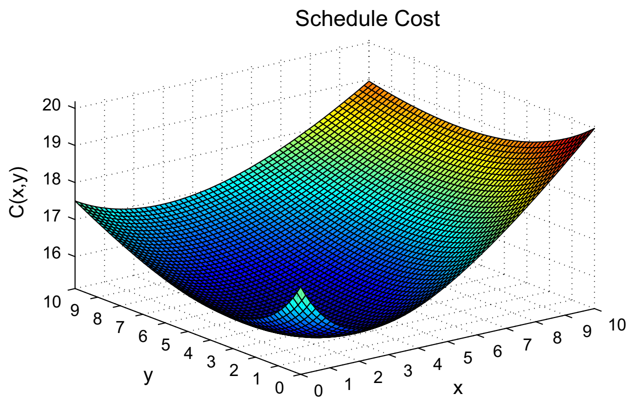

Fig. 1 Schedule cost plot: two sites

$\mathcal{C}(x,y) = 15.3287$. Since x and y must be integers we test four possible cases: $(x = \lfloor x_c \rfloor$ or $x = \lceil x_c \rceil$ and $(y = \lfloor y_c \rfloor or (y = \lceil y_c \rceil)$. The solution is $x = 2$ and $y = 3$ with $\mathcal{C}(x,y) = 15.3333$. Therefore, the schedule is $(S_1 S_1 \square\square S_2 S_2 S_2 \square\square)^*$.

6 Algorithms

One heuristic approach is to relax the IP formulation, given in Table 1, to obtain an LP-relaxation that could be efficiently solved, and then "round" the LP (fractional) solution to obtain a near-optimal solution. Unfortunately, solution to this LP-relaxation gives very loose bound. In fact, for increasing number of sites, the LP-relaxation optimizes the $y_{i,t}$ such that $(t - y_{i,t})$ for all i and t is 0 and the solver distributes fractional non-zero values among the $x_{i,t}$ such that the $\sum_{i=1}^{n} x_{i,t} \leq 1$. Fortunately, we have derived a tight lower bound in Sect. 4. We have implemented several greedy heuristics utilizing the solution of the LB given in Theorem 3.

The first greedy heuristic is called Greedy One-Step (see Algorithm 1). This heuristic tries to pick the next best site to visit based on the previously visited sites, as follows. It calculates a normalized frequency array q that dictates how often the sites must be scheduled based on a_i and b_i, where $q[i] = \frac{\sqrt{a_i+b_i}}{\sum_{i=1}^{n}\sqrt{a_i+b_i}}$. The idea behind this choice of $q[i]$ is due to the square-root rule, according to which if all a_i are the same then the $q[i]$ should be proportional to $\sqrt{b_i}$. However, if a_i are arbitrary, then it seems natural to try to incorporate the cost as the sum of a and b under the radical. Of course we can modify Algorithm 1 to calculate the frequency array by using τ_i's that are derived from the optimal solution of the nonlinear program 10 and let the $q[i] = \frac{1}{\tau_i}$. In fact, doing so may give better performance.

Algorithm 1 Greedy One-Step Algorithm.

```
for each site S_i ∈ 𝒮 do
  w[i] ← 0, r[i] ← 0
  q[i] ← √(a_i+b_i) / Σ_{i=1}^{n} √(a_i+b_i) (or q[i] ← 1/τ_i where τ_i is a solution to
  equation (10) of the lower bound)
end for
for t = 1 to T do
  for each site S_i ∈ 𝒮 do
    r[i] ← (1 + w[i])q[i]
  end for
  choose a site m maximizing r[m] and d_{i,j}[m][j] ≤ τ, where j is
  the previously scheduled site
  schedule S_m
  for each site S_i ∈ 𝒮 do
    w[i] ← w[i] + 1
  end for
  w[m] ← 0
  t ← t + 1
end for
```

Algorithm 2 Greedy Two-Steps Lookahead Algorithm.

```
for each site S_i ∈ 𝒮 do
  w[i] ← 0, r[i] ← 0
  q[i] ← √(a_i+b_i) / Σ_{i=1}^{n} √(a_i+b_i) (or q[i] ← 1/τ_i where τ_i is a solution to
  equation (10) of the lower bound.
end for
for t = 1 to T do
  for each site S_i ∈ 𝒮 do
    r[i] ← (1 + w[i])q[i]
  end for
  choose a site m maximizing r[m] and d_{i,j}[m][j] ≤ τ, where j is
  the previously scheduled site
  w2 ← w
  for each site S_i ∈ 𝒮 do
    w2[i] ← w2[i] + 1
  end for
  consider scheduling S_m
  w2[m] ← 0
  for each site S_i ∈ 𝒮 do
    r[i] ← (1 + w2[i])q[i]
  end for
  choose a site k maximizing r[k] and d_{i,j}[k][m] ≤ τ, where m is
  the previously considered scheduled site
  for each site S_i ∈ 𝒮 do
    r[i] ← (2 + w[i])q[i]
  end for
  let r[ℓ] ← max_i {r[i]}
  for each site S_i ∈ 𝒮 do
    w[i] ← w[i] + 1
  end for
  if r[ℓ] ≥ r[m] + r[k] then
    schedule blank
  else
    schedule m
    w[m] ← 0;
  end if
  t ← t + 1
end for
```

Notation 1 *In the following algorithms, let* $\mathcal{S} = \{S_1, S_2, \ldots, S_n\}$ *be the set of all sites*, $\mathcal{D}$ *be the angle matrix with entries* $d_{i,j}$, τ *be the limit angle that a camera can sweep within 1 time slot*, $q[\cdot]$ *be the frequency array of schedulability*, $w[\cdot]$ *be the gap array, and* $r[\cdot]$*be the potential cost array*.

The algorithm then proceeds as follows: at each step we keep track of the gap $w[i]$, which is the number of slots since the last appearance of the site i in the schedule. We schedule the site i which can be transitioned from previously scheduled site within one timeslot dictated by transition matrix $\mathcal{D}$ and that has the highest potential cost $r[i]$, which is calculated as $r[i] = (1 + w[i])q[i]$. We modified the algorithm to use $q[i] = \frac{1}{\tau_i}$, where τ_i are derived from the solution to the lower bound given in Theorem 3.

The problem with the Greedy One-Step Algorithm is that if two important sites, say A and D, are far away from each other separated by unimportant sites, say B and C, then in order to transition from A to D, the sensor needs to visit B in the next slot then C in the second slot and only then D in the third slot. It may however be more optimal not to schedule anything during the next slot and visit D right away during the second slot since in our formulation the sensor can use the entire idle slot just to do the transition to any other site. We have implemented a greedy solution that does lookahead: *Greedy Two-Steps Lookahead*. The algorithm allows for idle slots. In essence it tries to consider the next two best sites (one after another) and compare it with scheduling an idle slot with the next best site. It schedules an idle or non-idle slot next depending on whichever gave better performance. The algorithm considers sites as follows. Let $w_2 \leftarrow w$. For the next slot consider scheduling the site m that can be scheduled from the previously scheduled site (note: we assume that in timeslot 0 all the sites have been visited and hence scheduled) within one timeslot, dictated by transition matrix $\mathcal{D}$ and that has the highest potential cost $r[m]$, which is calculated as $r[m] = (1 + w_2[m])q[m]$. For each i, increment $w_2[i]$ by one and set $w_2[m] = 0$. Next consider scheduling the site k that can be scheduled from the previously scheduled site m within one timeslot dictated by transition matrix $\mathcal{D}$ and that has the highest potential cost $r[k]$, which is calculated as $r[k] = (1 + w_2[k])q[k]$. Compare it with the potential cost if there is an empty slot followed by scheduling any site ℓ. In other words, pick the largest $r[\ell] = (2 + w[\ell]) \cdot q[\ell]$. Compare $r[\ell]$ with $r[m] + r[k]$, and pick the largest residual cost. If the largest is $r[\ell]$, schedule an empty slot and increment all $w[i]$; otherwise, schedule site m and increment all $w[i]$, for $i \in \{1, \ldots, n\} - \{m\}$, and make $w[m] = 0$.

Algorithm 3 Greedy $(a_i + b_i(g_{i,t}))$-based Algorithm.

let $t(\ell)$ be the subscript of the site scheduled at time ℓ for $1 \le \ell \le k$
assume we have scheduled sites $\mathcal{D} = S_{t(1)}, \ldots, S_{t(k)}$
among immediately schedulable sites, schedule a site i maximizing $a_i + b_i(g_{i,t})$

Algorithm 4 Greedy Chained-IPs: IP(L,T).

for $s = 1$ to T **do**
  run $([x_s], [y_s]) \leftarrow IP(s, L, [x_{s-1}], [y_{s-1}])$
  update column s of $[x_{i,s}]$ matrix with array $[x_s]$ values, where $[x_s]$ is derived from IP solution
  update column s of $[y_{i,s}]$ matrix with array $[y_s]$ values, where $[y_s]$ is derived from IP solution
  $s \leftarrow s + 1$
end for
Output $[x_{i,t}]$ matrix as a solution.

Algorithm 5 Greedy $(a_i + b_i(g_{i,t}))$-based Look-back Algorithm.

let $t(\ell)$ be the subscript of the site scheduled at time ℓ for $1 \le \ell \le k$
assume we have scheduled sites $\mathcal{V} = S_{t(1)}, \ldots, S_{t(k)}$
next to schedule: site $S_{t(k+1)}$. Select it as follows:
let w_i be associated gaps at time k
pick the next site to schedule with biggest $\frac{(1+w_i)(2+w_i)}{2} b_i + (1+w_i)a_i$ that can be transitioned from site $S_{t(k)}$.

Other greedy algorithms do not calculate frequency array but use a_i and b_i costs directly. Algorithm 3 schedules sites as follows. At each step it greedily picks a site with maximum $a_i + b_i(g_{i,t})$. In other words, assume we already have a schedule up to time k which has a cost of $\mathcal{C} = \frac{1}{k}\sum_{i=1}^{n}\sum_{t=1}^{k} a_i(1 - x_{i,t}) + b_{i,t}(t - y_{i,t})$. Pick the next site i with maximum $C + (a_i + b_i(g_{i,t}))$ that can be transitioned from the previously scheduled site.

Another algorithm we implemented repeatedly solves IPs to construct a solution (see Algorithm 4).

Notation 2 *In the following algorithm 4 , let*

- T *be the schedule period*,
- L *be the number of slots to look ahead*,
- $[x_{i,t}]$ *be an* $n \times T$ *matrix with* $0/1$ *entries indicating which site is scheduled at time* t *(each column* t *has exactly one* 1 *entry)*,
- $[y_{i,t}]$ *be an* $n \times T$ *matrix that tells last appearances of all sites* i *at time* t,
- s *be a starting timeslot to run the IP*,
- $[x_p]$ *be a* $0/1$ *array of scheduled sites at time* $t = p$ *produced by IP*,
- $[y_p]$ *be an array of the times of last observing sites at time* $t = p$ *produced by IP*,

- $[x_0] \leftarrow \{0,\ldots,0\}$ *(initial values for* $t=0$*),* $[y_0] \leftarrow \{0,\ldots,0\}$ *(initial values for* $t=0$*), and*
- $IP(s, L, [x_p], [y_p])$ *be an IP that starts at timeslot s and period L with some initial values of* $[x_p]$ *and* $[y_p]$.

The algorithm chains IPs as follows. It finds an optimal solution with a period L for timeslots 1 to L. It selects the first scheduled site, updates the $x_{i,1}$ and $y_{i,1}$ values, and advances a window by 1. It runs IP again to find an optimal solution with a period L for timeslots 2 to $L+1$ using the updated $x_{i,1}$ and $y_{i,1}$ values, updates $x_{i,2}$ and $y_{i,2}$ values, and advances a window by 1 again. It continues doing this until it gets values for all $x_{i,t}$ and $y_{i,t}$ for $1 \le t \le T$. This fully specifies the schedule. The number of IP programs that we run is T, each time advancing a window by one and recording the best first site that the IP on L timeslots picks. This algorithm can be thought of as the cost-based L-Steps Lookahead greedy algorithm. Algorithm 3 is a special case of algorithm 4 where $L=1$, L is a lookahead parameter and T is a period of the schedule parameter.

Another cost-based greedy algorithm is the $(a_i + b_i(g_{i,t}))$-based Look-back (see Algorithm 5). Just as in Algorithm 3, it does not need to calculate frequencies. A drawback for Greedy $(a_i + b_i(g_{i,t}))$-based algorithm is that it assumes that previously scheduled sites are scheduled optimally and just selects the best next site based on the a_i and b_i costs. The $(a_i + b_i(g_{i,t}))$-based Look-back algorithm does not make this assumption. At each step it greedily picks a site with maximum residual cost without assuming that the previously scheduled sites were optimal. It uses gap information $g_{i,t}$ (which in our greedy solution we refer to by w_i with no reference to t since the schedule is built incrementally) to select the best site. In other words, assume we already have a schedule up to time k. At time $k+1$, the algorithm picks a site maximizing $(w_i+1)a_i + (1+2+\cdots+w_i+(w_i+1))b_i$, where w_i is a gap of site i at time k. Note that the fixed cost for a site of a schedule grows linearly, and variable cost for a site of a schedule grows quadratically with respect to the gap.

7 Testbed architecture

Our experimental evaluation concerns the following application. A border, which may be a straight line or a curve, has n unprotected sites that need to be monitored. These n potential sites are distributed uniformly at random. A single sensor is positioned, hidden in front of the border, that is responsible in monitoring intrusions from these unprotected sites. We implemented the algorithms above to schedule a single sensor to visit n sites periodically while minimizing the potential loss of limited observations. The refocus delay time is proportional to the angle that the sensor needs to sweep to switch from one site to the next.

7.1 Various experiments

We conducted various experiments where a single sensor is scheduled to observe four sites A, B, C, D, spaced equally apart, on a border located at distance 10 from the sensor (see Fig. 2; Tables 3, 4 for an illustration of this configuration, the respective costs, and the resulting angles between cameras and sites, respectively). Our assumption is that a camera can visit the site and be able to sweep 45° within one time slot. If the next site is within more than 45° from the currently visiting site, then the camera cannot observe those two sites in immediate succession. In this example, the only adjacent sites can be reached and observed in the next timeslot.

We solved the IP using CPLEX to find optimal schedules for several different scenarios for the costs of the four

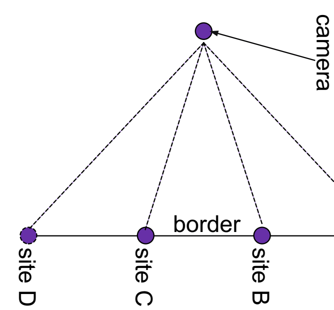

Fig. 2 Experiment 1 setup

Table 3 Seven cost scenarios for four sites

Site i	Cost													
	Exper. 1:		*Exper. 2*:		*Exper. 3*:		*Exper. 4*:		*Exper. 5*:		*Exper. 6*:		*Exper. 7*:	
	a_i	b_i	a_i	b_i	a_i	b_i	a_i	b_i	a_i	b_i	a_i	b_i	a_i	b_i
A	1	1	16	1	1	16	4	4	4	4	16	16	16	16
B	4	4	9	4	4	9	16	16	9	9	1	1	4	4
C	9	9	4	9	9	4	9	9	16	16	4	4	1	1
D	16	16	1	16	16	1	1	1	1	1	9	9	9	9

Table 4 Angles between sites

Angle	A	B	C	D
A	0°	30°	60°	90°
B	30°	0°	30°	60°
C	60°	30°	0°	30°
D	90°	60°	30°	0°

sites (see Table 3), to investigate what patterns might emerge. In all seven experiments we have selected costs to be perfect squares for simplicity, due to the square-root law [9], which states that in an optimal schedule (absent delay constraints and with $a_i = a_j = 0$), the ratio of visit frequencies should be proportional to the square-root of this ratio. For example, if $b_i = 4$ and $b_j = 16$, then site j will be visited twice as often as site i, since $\frac{\sqrt{16}}{\sqrt{4}} = 2$.

First we describe the nature of these cost value settings, as shown in Table 3. In experiment 1, fixed costs a_i and variable costs b_i increase for each additional site. In experiment 2, variable costs b_i increase for each additional site, but fixed costs decrease. Note, however, that the costs associated with b_i grow quadratically with the gap size, whereas the costs associated with the a_i grow linearly with the gap size. Experiment 3 reverses this, with variable costs b_i decreasing and fixed costs increasing. This setting is identical to the one in experiment 2 with the sites are renamed. In experiments 4 and 5, sites B and C have greater values for both costs, and are in the middle, surrounded by sites A and D. In experiments 6 and 7, the high-cost sites A and D are on the outside, surrounding sites B and C.

We solved the IP to obtain an optimal schedule (for period $T = 21$) for each of these cost settings. These optimal periodic schedules are shown in Fig. 3. We also ran the heuristic algorithms above on these problem instances and compared the resulting schedule costs with the optimal costs from the IP, as well as a lower bound on the optimal cost. (see Table 5. The cost of the solution to the lower bound is obtained by solving Eq. (10).

Next we conducted an experiment to compare the schedule costs of the different algorithms on randomly generated problem instances, where the *fixed* (i.e. a_i) and *variable* (i.e. b_i) costs are both chosen uniformly at random from the real interval $[0, 10]$. The number of sites n is

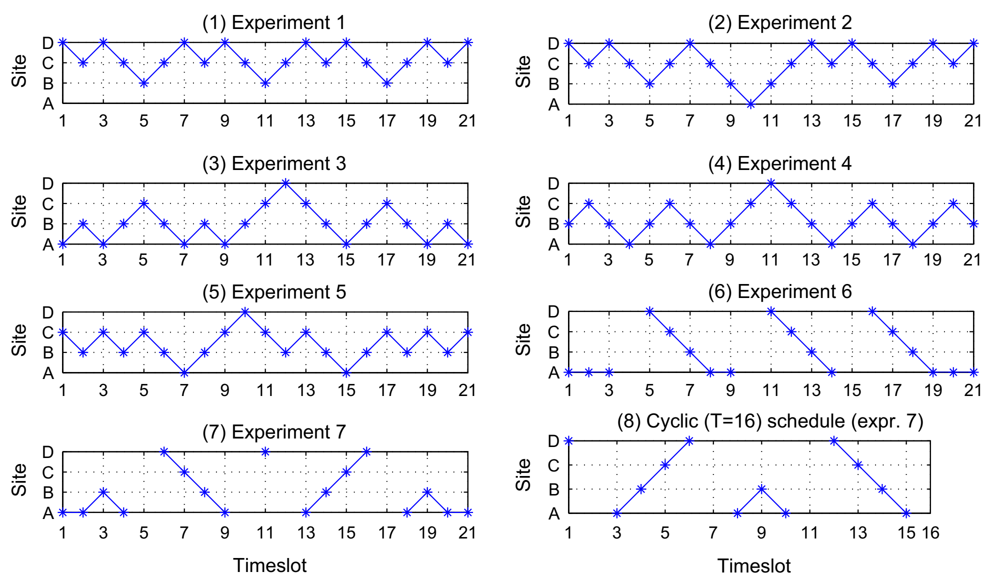

Fig. 3 IP solutions for the 7 experiments

Table 5 Comparison of IP and Greedy schedule costs per slot

	IP	One-Step $q_i = \frac{\sqrt{a_i+b_i}}{\sum_{i=1}^{n}\sqrt{a_i+b_i}}$	One-Step $q_i = \frac{1}{\tau_i}$	Two-Steps Lookahead	$a_i + b_i(g_{i,t})$-based	$a_i + b_i(g_{i,t})$-based Look-back	IP(10, 21)	LB
Exper. 1	61.05	74.57	67.52	67.57	66.57	62.95	64.81	54.85
Exper. 2	68.05	78.29	76.24	72.43	81.52	78.29	69.71	59.88
Exper. 4	53.09	55.67	55.67	55.67	58.29	54.67	54.86	54.85
Exper. 5	57.62	59.14	58.76	58.76	66.57	58.57	60.86	54.85
Exper. 6	77.43	93.95	98.71	80.48	168.00	108.19	78.81	54.85
Exper. 7	75.67	90.38	92.71	84.81	140.19	103.95	78.95	54.85

ranges from 4 to 8, placed linearly along a border. As in the previous seven experiments, the adjacent sites can be transitioned between instantly, and non-adjacent sites can be transitioned between in one idle slot. For each such n, we average the results of 100 trials (see Fig. 4).

We find that IP(L, T) performs quite well in all the experiments, even for small L. Of course, L should depend on the number of sites to be scheduled. For the lookahead parameter $L = 10$ the algorithm runs quite fast for any value of T. To investigate the influence of the value L ,we conducted experiments varying L with IP(L, T) solved for a fixed $T = 21$ (see Fig. 5). Using what we found to be the best two values for L, we conducted another set of experiments on IP(L, T) where L is fixed and the period T is varied (see Fig. 6).

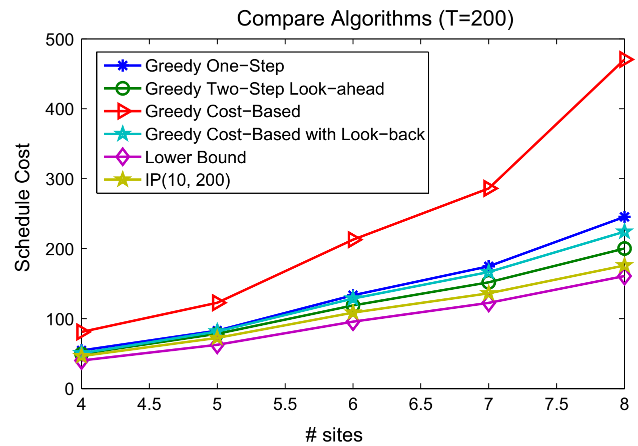

Fig. 4 Comparison of algorithms

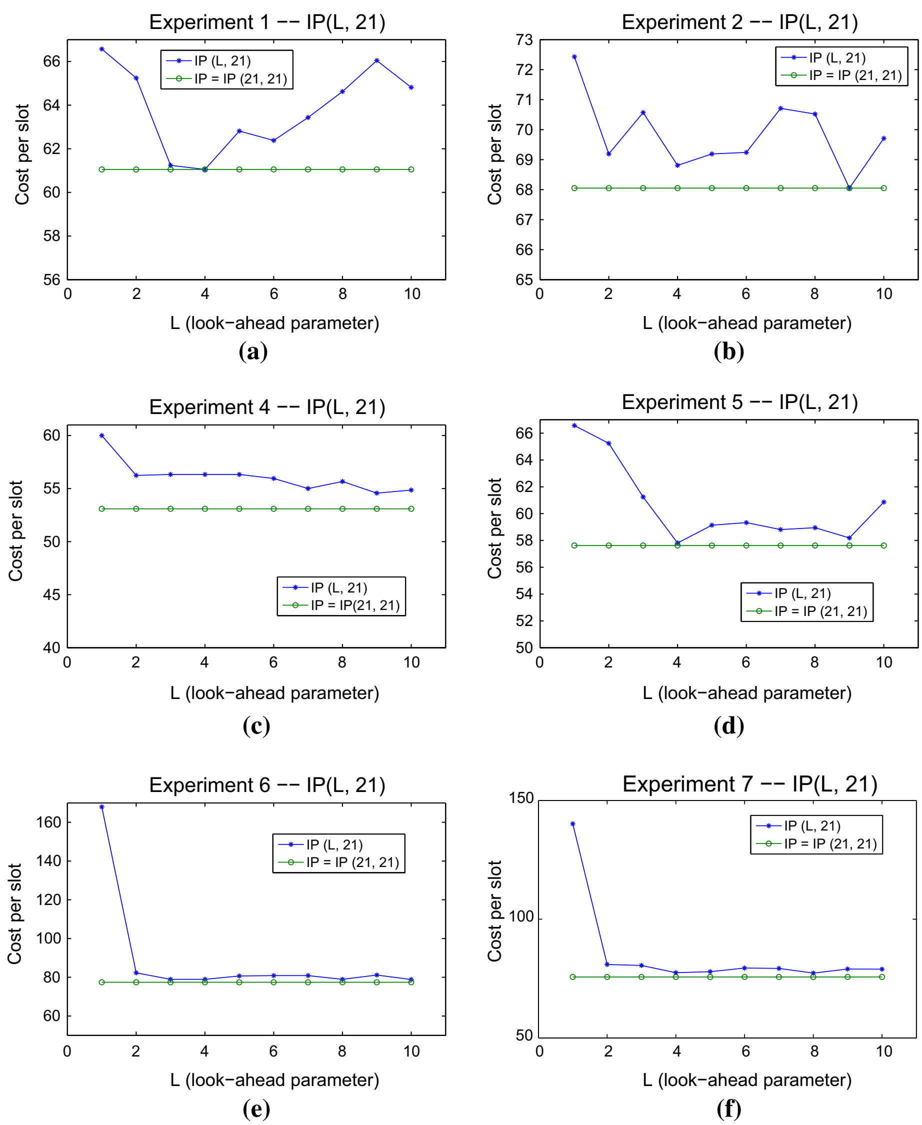

Fig. 5 Compare solutions of IP(L, 21) by varying L. **a** Experiment 1. **b** Experiment 2. **c** Experiment 4. **d** Experiment 5. **e** Experiment 6. **f** Experiment 7

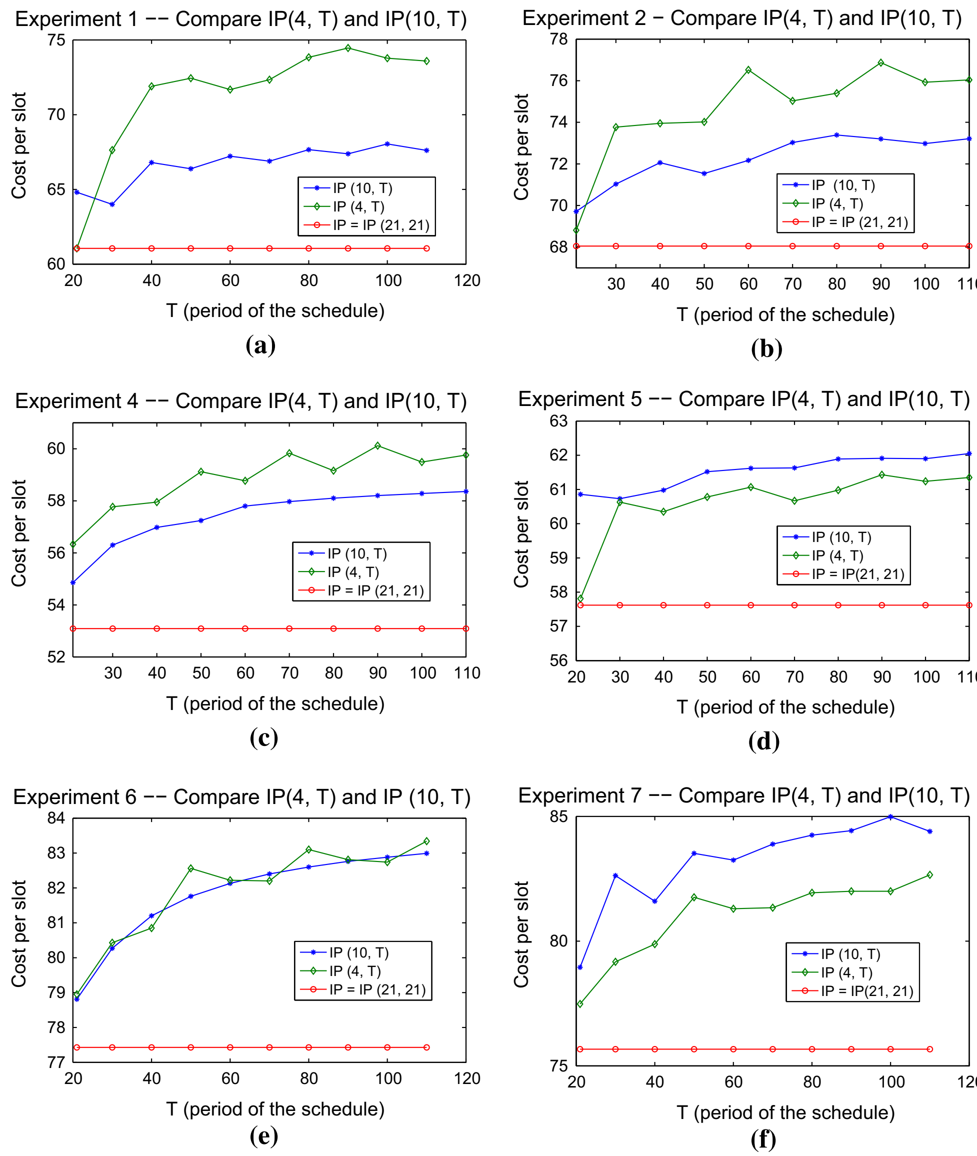

Fig. 6 Compare solutions of IP(4, T) and IP(10, T) by varying T. **a** Experiment 1. **b** Experiment 2. **c** Experiment 4. **d** Experiment 5. **e** Experiment 6. **f** Experiment 7

7.2 Insights from simulation

The schedules produced by IP for the seven experiments with sites A, B, C, D positioned along the border are depicted in Fig. 3. Recall that in experiment 1 [see Fig. 3(1)], the fixed and variable costs both increase for each site. Absent delay constraints, this would dictate that D be visited the most often, C the second most, and so on. Since site D is on the edge, instantaneously reachable from only C, D is actually visited less often than C is. The refocus delays, together with site positions, play important roles in determining the character of the optimal schedule. (Site A was not scheduled at all, though it would be for sufficiently large period T.)

Recall that in experiments 2 and 3 [see Fig. 3(2–3)], fixed cost grows as variable cost shrinks. What we find is that site A's high fixed cost results in it now being visited (with period still $T = 21$). Site C remains the most popular, however, due to its central position.

Recall that in experiments 4 and 5, the high-cost sites are in the middle, surrounded by the low-cost sites [see Fig. 3(4), (5)]. In both experiments, site A has the same fixed and variable costs, but because A's neighbor B is the highest cost site in experiment 4, it is scheduled twice as often in that experiment as it is in experiment 5, in which B is only the second highest cost site. When fixed and variable costs are held fixed, relative positions of sites play a crucial role in determining the schedule.

Finally, recall that in experiments 6 and 7, the high-cost sites are on the outside, surrounding the low-cost sites [see Fig. 3(6), (7)]. Now there is more incentive to schedule outside sites, but the problem is that the camera cannot switch instantaneously between these two important sites.

Thus we find the optimal schedules using idle slots for refocusing.

It is interesting to note that in all 7 experiments the schedules appear to be cyclic. In fact, when we decrease the time horizon in experiment 7–16, we again obtain a periodic schedule [see Fig. 3(8)].

From these seven experiments we observe that due to refocus parameter the frequency of a scheduled site in the optimal schedule does not only depend on the fixed and variable costs of a site, but on its relative position to other important sites in the region. In addition, it is sometimes optimal to have idle slots during which no site is scheduled but the entire time is taken by the sensor to move or refocus.

We also present results for the algorithms run on randomized problem instances (Fig. 4). The period in these experiments is arbitrarily chosen to be $T = 200$. We find that Greedy Two-Steps Lookahead outperforms Greedy One-Step for frequency-based algorithms, and the cost-based Look-back Greedy outperforms the simple cost-based Greedy. IP(10, 200), significantly outperforms the others, and comes close to the IP optimal cost. We conducted an experiment where we run algorithms without delay constraints and observed that the costs of optimal schedules can come arbitrarily close to the cost of the lower bound schedule.

Since IP(10,21) comes close to the IP optimal solution, what about IP(L,21) with smaller values of L? Next we performed versions of the seven 4-site experiments with different L (see Fig. 5, in which the IP optimal, which does not depend on a lookahead parameter, is plotted for comparison). What we find is that the curves given by IP(L, T) fluctuate. A common pattern is that as L increases, the IP(L, T) curve zigzags up and down. In Fig. 5(e, f) we see that the bigger the lookahead parameter L, the closer the IP(L, T) curve gets to the IP curve, but again with a fluctuating pattern that appears periodic.

We examine two particular lookahead values in our last set of experiments (see Fig. 6). Here we plot IP(4, T) and IP(10, T), varying time horizon T. Even though for $T = 21$, the IP(4, T) curve is closer to the IP curve (as seen in Fig. 6a, b, d), it jumps up for larger values of T. We also find that the curve for IP(10, T) is smoother than IP(4, T)'s in all settings, and lower for most of them. Even for those cases, the cost is nearly the same. We conclude that for longer time horizons, a larger lookahead value gives better results, as expected, and smoother behavior.

8 Conclusion and future work

In this paper we have studied a scheduling problem of a single sensor observing n sites. We considered the time of refocus delay and its impact on scheduling. Our work poses interesting new problems. Since refocusing sensors from site to site may involve physically rotating or moving sensors, refocusing may consume a considerable amount of energy. Energy conservation may be of the essence. Scheduling sensors optimally to observe sites while at the same time conserving energy is an interesting open problem which we leave for our future work.

The single sensor scheduling problem can also be extended to a multiple-sensors setting. In the multiple sensor scheduling we want to schedule multiple sensors observing much bigger quantity of sites. One approach here would be to efficiently partition a set of sites into subsets and then assign each sensor its own subset. Another approach would be to have sensors cooperate by scheduling all m sensors to observe n sites. We defer such problems to future work.

Acknowledgments This research was sponsored by US Army Research laboratory and the UK Ministry of Defence and was accomplished under Agreement Number W911NF-06-3-0001. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the US Army Research Laboratory, the US Government, the UK Ministry of Defence, or the UK Government. The US and UK Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

References

1. Yavuz, M., & Jeffcoat, D. E. (2007). *An analysis and solution of the sensor scheduling problem*. Berlin: Springer.
2. Yavuz, M., & Jeffcoat, D.E. (2007). Single sensor scheduling for multi-site surveillance. Air Force Research Laboratory : Technical report.
3. Boyko, N., Turko, T., Boginski, V., Jeffcoat, D.E., Uryasev, S., Zrazhevsky, G., & Pardalos, P.M. (2009). Robust multi-sensor scheduling for multi-site surveillance. *Journal of Combinatorial Optimization*, ISSN 1382–6905. doi:10.1007/s10878-009-9271-4.
4. Boyko, N., Turko, T., Boginski, V., Jeffcoat, D. E., Uryasev, S., Zrazhevsky, G., et al. (2011). Robust multi-sensor scheduling for multi-site surveillance. *Journal of Combinatorial Optimization*, *22*(1), 35–51.
5. Jarray, F. (2010). Complexity results for wireless sensor network scheduling. *Wireless Sensor Network*, *2*(5), 343–346.
6. Kalinchenko, K., Veremyev, A., Boginski, V., Jeffcoat, D. E., & Uryasev, S. (2011). Robust connectivity issues in dynamic sensor networks for area surveillance under uncertainty. *Structural and Multidisciplinary Optimization*, *7*(2), 235–248.
7. Uryasev, S., & Pardalos, P. (2011). *Detecting and jamming dynamic communication networks in anti-access environments*. DTIC Document: Technical report.
8. Acharya, S., Alonso, R., Franklin, M.J., & Zdonik, S.B. (1995). Broadcast disks: Data management for asymmetric communications environments. In SIGMOD conference, pp. 199–210.
9. Ammar, M. H., & Wong, J. W. (1985). The design of teletext broadcast cycles. *Performance Evaluation*, *5*(4), 235–242.
10. Ammar, M. H., & Wong, J. W. (1987). On the optimality of cyclic transmission in teletext systems. *IEEE Transactions on Communications*, *35*(1), 68–73.

11. Bar-Noy, A., Bhatia, R., Naor, J. S., & Schieber, B. (2002). Minimizing service and operation costs of periodic scheduling. *Mathematics of Operations Research*, *27*, 518–544.
12. Silberschatz, A., Galvin, P., & Gagne, G. (2002). *Operating system concepts* (6th ed.). New York: Wiley.
13. Fiat, A., Rabani, Y., & Ravid, Y. (1994). Competitive k-server algorithms. *Journal of Computer and System Sciences*, *48*(3), 410–428.
14. Chin, W., & Ntafos, S.C. (1986). Optimum watchman routes. In Symposium on computational geometry, pp. 24–33.
15. Costello, C.J., Diehl, C.P., Banerjee, A., & Fisher, H. (2004). Scheduling an active camera to observe people. In *VSSN '04: Proceedings of the ACM 2nd international workshop on video surveillance and sensor networks*, (pp. 39–45). New York, NY: ACM.
16. Yiliang, X., & Song, D. (2010). Systems and algorithms for autonomous and scalable crowd surveillance using robotic ptz cameras assisted by a wide-angle camera. *Autonomous Robots*, *29*(1), 53–66.
17. Sharma, N., Irwin, D., Zink, M., & Shenoy, P. (2012). Multi-sense: Proportional-share for mechanically steerable sensor networks. *Multimedia Systems*, *18*(5), 425–444.
18. Piciarelli, C., Micheloni, C., & Foresti, G.L. (2010). Occlusion-aware multiple camera reconfiguration. In *Proceedings of the fourth ACM/IEEE international conference on distributed smart cameras, ICDSC '10* (pp. 88–94). New York, NY: ACM.
19. Starzyk, W., & Qureshi, F.Z. (2011). Learning proactive control strategies for ptz cameras. In Distributed smart cameras (ICDSC), 2011 fifth ACM/IEEE international conference on, pp. 1–6.
20. Mavrinac, A., & Chen, X. (2013). Modeling coverage in camera networks: A survey. *International Journal of Computer Vision*, *101*(1), 205–226.
21. Garey, M. R., & Johnson, D. S. (1979). *Computers and intractability: A guide to the theory of NP-colmpleteness*. San Francisco: Freeman.
22. Anily, S., Glass, C. A., & Hassin, R. (1998). The scheduling of maintenance service. *Discrete Applied Mathematics*, *82*(1–3), 27–42.

Yosef Alayev received B.S. degree in Computer Science and B.A. degree in Applied Mathematics in 2002 from Queens College of the City University of New York. He received M.Phil. and Ph.D. degrees from the Department of Computer Science at the Graduate Center of the City University of New York in 2009 and 2014 respectively. His graduate research was in scheduling and resource allocation algorithms in wireless sensor network environment. His interests include combinatorial optimization, mathematical programming, and algorithm analysis and implementation.

Amotz Bar-Noy received the B.Sc. degree (1981) in Mathematics and Computer Science (summa cum laude) and the Ph.D. degree (1987) in Computer Science, both from the Hebrew University, Israel. He was a post-doc fellow in Stanford University, California (1987–1989); a Research Staff Member with IBM Research Center, New York (1989–1996); an associate Professor with the Electrical Engineering Department of Tel Aviv University, Israel (1996–2001); a Principal Technical Staff Member with AT&T research labs in New Jersey (1999–2001). Since 2002 he is a Professor with the Computer and Information Science Department of Brooklyn College—CUNY, Brooklyn New York and with the Computer Science Department of the Graduate Center of CUNY, Manhattan New York. He has published more than eighty refereed journal articles and more than one hundred refereed conference and workshop articles. He served as a program committee member for many conference. He was an editor for the IEEE Transactions on Mobile Computing (TMC) journal and for the Wireless Networks (WINET) journal. He served as a guest editor for two special issues one of Wireless Network and one of Mobile Networking and Applications. He served as a co-chair of the ALGOSENSORS 2012 Symposium. In 2011 he was awarded the Edsger W. Dijkstra Prize in Distributed Computing for being an author of an outstanding paper on the principles of distributed computing, whose significance and impact on the theory and/or practice of distributed computing has been evident for at least a decade. His field of expertise belongs to the Theoretical Computer Science community and to the Networking community. The scope of his research is to bridge the gap between these two communities.

Matthew P. Johnson studied philosophy and computer science at Columbia and Lawrence Universities and received his Ph.D. in computer science from the City University of New York in 2010. He is currently an assistant professor in the Department of Mathematics and Computer Science at Lehman College, CUNY, and a member of the doctor faculty at the CUNY Graduate Center.

Lance Kaplan received the B.S. degree with distinction from Duke University, Durham, NC, in 1989 and the M.S. and Ph.D. degrees from the University of Southern California, Los Angeles, in 1991 and 1994, respectively, all in Electrical Engineering. From 1987–1990, Dr. Kaplan worked as a Technical Assistant at the Georgia Tech Research Institute. He held a National Science Foundation Graduate Fellowship and a USC Dean's Merit Fellowship from 1990–1993, and worked as a Research Assistant in the Signal and Image Processing Institute at the University of Southern California from 1993–1994. Then, he worked on staff in the Reconnaissance Systems Department of the Hughes Aircraft Company from 1994–1996. From 1996–2004, he was a member of the faculty in the Department of Engineering and a senior investigator in the Center of Theoretical Studies of Physical Systems (CTSPS) at Clark Atlanta University (CAU), Atlanta, GA. Currently, he is in the Networked Sensing and Fusion branch of the U.S. Army Research Laboratory (ARL). Dr. Kaplan serves as Editor-In-Chief for the IEEE Transactions on Aerospace and Electronic Systems (AES). In addition, he also serves on the Board of Governors of the IEEE AES Society, 2009–present, and on the International Society of Information Fusion (ISIF) Board, 2012–present. He served as Technical Co-Chair (with Neil Gordon) for the 2011 ISIF/IEEE International Conference on Information Fusion in Chicago, IL. He is a three time recipient of the Clark Atlanta University Electrical Engineering Instructional Excellence Award from 1999–2001. Dr. Kaplan has published over 170 technical articles. His current research interests include signal and image processing, information/data fusion, resource management, and network science.

Thomas F. La Porta is the William E. Leonhard Chair Professor in the Computer Science and Engineering Department at Penn State. He received his B.S.E.E. and M.S.E.E. degrees from The Cooper Union, New York, NY, and his Ph.D. degree in Electrical Engineering from Columbia University, New York, NY. He joined Penn State in 2002. He is the Director of the Institute of Networking and Security Research at Penn State. Prior to joining Penn State, Dr. La Porta was with Bell Laboratories since 1986. He was the Director of the Mobile Networking Research Department in Bell Laboratories, Lucent Technologies where he led various projects in wireless and mobile networking. He is an IEEE Fellow, Bell Labs Fellow, received the Bell Labs Distinguished Technical Staff Award in 1996, and an Eta Kappa Nu Outstanding Young Electrical Engineer Award in 1996. He also won a Thomas Alva Edison Patent Awards in 2005 and 2009. His research interests include mobility management, signaling and control for wireless networks, security for wireless systems, mobile data systems, and protocol design. Dr. La Porta was the founding Editor-in-Chief of the IEEE Transactions on Mobile Computing and served as Editor-in-Chief of IEEE Personal Communications Magazine. He was the Director of Magazines for the IEEE Communications Society and was on its Board of Governors for 3 years.