# Stochastic Steepest Descent Optimization of Multiple-Objective Mobile Sensor Coverage

Chris Y. T. Ma, David K. Y. Yau, *Member, IEEE*, Nung Kwan Yip, Nageswara S. V. Rao, *Fellow, IEEE*, and Jiming Chen, *Senior Member, IEEE*

***Abstract*—We propose a steepest descent method to compute optimal control parameters for balancing between multiple performance objectives in stateless stochastic scheduling, wherein the scheduling decision is effected by a simple constant-time coin toss operation only. We apply our method to the scheduling of a mobile sensor's coverage time among a set of points of interest (PoIs). The coverage algorithm is guided by a Markov chain, wherein the sensor at PoI $i$ decides to go to the next PoI $j$ with transition probability $p_{ij}$. We use steepest descent to compute the transition probabilities for optimal tradeoff among different performance goals with regard to the distributions of per-PoI coverage times and exposure times and the entropy and energy efficiency of sensor movement. For computational efficiency, we show how we can optimally adapt the step size in steepest descent to achieve fast convergence. However, we found that the structure of our problem is complex, because there may exist surprisingly many local optima in the solution space, causing basic steepest descent to easily get stuck at a local optimum. To solve the problem, we show how proper incorporation of noise in the search process can get us out of the local optima with high probability. We provide simulation results to verify the accuracy of our analysis and show that our method can converge to the globally optimal control parameters under different assigned weights to the performance goals and different initial parameters.**

***Index Terms*—Mobile sensor network, multiple-objective optimization, steepest descent.**

## I. Introduction

IT IS KNOWN that stochastic scheduling techniques can achieve service differentiation by maintaining a minimal scheduling state. For example, stochastic fair queuing (SFQ) [1] allows a router to allocate weighted service rates among flows without bookkeeping of past service that was received by each flow. In SFQ, to give a $p_i$ $(0 \leq p_i \leq 1)$ fraction of service to flow $i$, we toss a coin and, with probability $p_i$, serve the next packet from $i$. SFQ has a constant-time computational complexity. In contrast, the deterministic scheduling technique of weighted fair queuing (WFQ) [2] selects the next flow for service based on a sorting of per-flow finish numbers. Bookkeeping the finish numbers has a computational complexity of $O(n^2)$ for $n$ flows. For central processing unit (CPU) time sharing, lottery scheduling [3] is a probabilistic algorithm that allows processes to receive long-term CPU rates in proportion to their assigned numbers of lottery tickets. In contrast, stride scheduling [4], a deterministic implementation of the lottery scheduling concept, selects processes to run based on a global pass number and per-process pass numbers similar to WFQ scheduling variables.

Aside from service rates, fairness is an important scheduling metric. Fairness determines how long a client, after receiving any amount of service, has to wait before being selected again for service. For example, a flow that is given the first of every ten packet times of service has the same rate as another flow that is given the first two of every 20 packet times of service. However, the first flow achieves finer grained fairness compared with the second flow. Deterministic schedulers can be designed for different fairness goals, independent of the service rates. For example, for the same set of rate guarantees to flows, WF$^2$Q [5] is designed to achieve finer grained fairness than WFQ. On the other hand, existing stochastic schedulers do not try to independently control the rate and fairness performance. For example, to increase the frequency at which flow $i$ is selected for service in SFQ, we have to increase $p_i$, but doing so will also increase the service rate of $i$. In other words, there is a coupling between rate and fairness in SFQ.

In this paper, we propose a steepest descent method to compute control parameters for achieving an optimal tradeoff between multiple objectives in stateless stochastic scheduling. By stateless we mean that the scheduling algorithm needs no bookkeeping of prior service received by clients, but the scheduling decision is entirely determined by a constant-time coin toss operation. The notion of optimal balance between multiple performance objectives is user or application defined. Our ability to support a wide range of multiobjective performance tradeoffs and optimize the tradeoff is absent in existing stochastic scheduling algorithms.

We apply our method to the problem of scheduling a mobile sensor's coverage time among a number of disjoint *points of*

Manuscript received June 20, 2011; revised October 13, 2011 and January 2, 2012; accepted February 9, 2012. Date of publication February 29, 2012; date of current version May 9, 2012. This work was supported in part by the U.S. National Science Foundation under Grant CNS-0964086; the National Natural Science Foundation of China under Grant 61028007; and the Office of Advanced Computing Research, U.S. Department of Energy, through the Mathematics of Complex, Distributed, Interconnected Systems Program. The review of this paper was coordinated by Prof. V. W. S. Wong.

C. Y. T. Ma is with the Advanced Digital Sciences Center, Singapore 138632 (e-mail: chris.ma@adsc.com.sg).

D. K. Y. Yau is with the Department of Computer Science, Purdue University, West Lafayette, IN 47907 USA, and also with the Advanced Digital Sciences Center, Singapore 138632 (e-mail: yau@cs.purdue.edu).

N. K. Yip is with the Department of Mathematics, Purdue University, West Lafayette, IN 47907-2067 USA (e-mail: yip@math.purdue.edu).

N. S. V. Rao is with the Computer Science and Mathematics Division, Oak Ridge National Laboratory, Oak Ridge, TN 37831-6367 USA (e-mail: raons@ornl.gov).

J. Chen is with the Department of Control Science and Engineering and also with the State Key Laboratory of Industrial Control Technology, Zhejiang University, Hangzhou 310027, China (e-mail: jmchen@ieee.org).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVT.2012.2189591

*interest* (PoIs). In this case, the sensor moves between the PoIs to capture interesting information, and the notions of rate and fairness have similarly important interpretations. In particular, rate corresponds to the fraction of time that the sensor covers a PoI. The larger the fraction of time is, the more information the sensor can collect about the PoI. Fairness, on the other hand, determines the average return time of the sensor to a PoI. The longer the return time is, the longer the PoI can be exposed to significant incidents that have occurred but are left undetected. The delay in responding to an incident, for example, an accident that requires rescue operations, may increase. In addition to rate and fairness, movement entropy and energy efficiency are important scheduling metrics. An increased entropy implies an increased randomness of the coverage schedule, which will make it harder for smart adversaries to anticipate the sensor's location and thus avoid detection. Higher energy efficiency allows the sensor to stay operational longer, thereby reducing the replacement and management costs.

One important issue in our problem context is that the solution space is extremely complex due to a large number of control parameters that affect multiple performance goals. In particular, there exist a *surprisingly large number of local minima*. Hence the incorporation of noise in the optimization is essential to avoid getting stuck at these local optima. We will show that our stochastic optimization method can optimally balance among the performance metrics. To the best of our knowledge, there is no existing coverage algorithm—deterministic or stochastic and constant time or otherwise—that can provide such optimal tradeoffs.

The case for mobility in monitoring applications has extensively been discussed in the literature [6]–[8]. In general, mobile nodes are useful in situations where the installation of extensive static sensing/communication devices is difficult. For example, chemical sensors can be deployed at specific locations in a water distribution system (WDS) to monitor the water quality for the growth of biological masses, the presence of contaminants, and other factors [9], [10]. Long-range wireless communication under water is, however, infeasible due to high signal attenuation. To solve the problem, a mobile node can be used to move among the underwater sensors, gather data from the sensors, and carry the data to a sink for reporting. In this case, it has been recognized that the overall optimization problem has multiple objectives that may be antagonistic [10]. For example, focused data collection near the periphery of the WDS (in particular, immediate downstream of likely entry points of contaminants) may minimize the delay of detection, but focused data collection at the center of the WDS may maximize the detection probability. Researchers have also deployed *mobile catamarans* to monitor water quality in lakes and rivers [11]. A catamaran is energy limited and carries sensors to measure the temperature, pH, conductivity, and dissolved nutrients of the water. It autonomously moves on the water surface to cover different parts of an area of interest over time. When it runs out of energy, it must be recharged, or the monitoring will cease to work. Hence, the issue of energy efficiency is critical for these robotic monitoring agents, although this issue is not investigated in [11].

Our *contributions* in this paper are listed as follows.

1) We analyze a general stochastic algorithm to schedule a mobile sensor's coverage time among a set of disjoint PoIs, which is denoted by $S$. The scheduling is controlled by a discrete-time Markov chain that takes values in $S$, wherein the sensor at PoI $i$ decides to move to PoI $j$ in the next time step with transition probability $p_{ij}$. (If $i = j$, the sensor stays at the same PoI.) The performance of the Markov chain is measured by its time-limiting measure of a cost function. The key feature of our setting is that the cost reflects conflicting criteria such as the distributions of per-PoI coverage and exposure times. In addition, our Markov model accounts for physical constraints that are associated with geographical placements of the PoIs. In particular, travel from one PoI to another must occur along a physically feasible route and is subject to speed constraints. The sensor may also go through other PoIs that are on the route, thereby affecting the coverage time of these intermediate PoIs.
2) We provide closed-form analytical formulas in terms of the Markov transition probabilities to express the cost function. To provide a specific model example, we consider the following four criteria:
   - the achieved distribution of coverage times among the PoIs;
   - the achieved distribution of the per-PoI exposure times;
   - the entropy of the movement schedule;
   - the energy efficiency of the schedule.

   Such a formulation greatly simplifies the simulation and optimization procedure. Based on our analytical formulas, we develop a steepest descent optimization method to determine the Markov transition probabilities such that the cost function, expressed as a weighted sum of the performance metrics, is minimized. The optimization is comprehensive, because we search for an optimal solution in the space of *all* transition probabilities. The trend of our optimal solutions agrees with the physical interpretation. In particular, when considering the tradeoff between coverage time and per-PoI exposure time, we show that, as we reduce the weight of the exposure time metric, the sensor can adapt by moving less between the PoIs, thereby conserving motion energy and more closely achieving the target per-PoI allocation of coverage times.
3) As a practical numerical computation issue, we discuss how we can estimate optimal time steps in the steepest descent to achieve fast convergence to the optimal Markov transition probabilities.
4) We present empirical evidence that there may exist an extremely large number of local optima in the solution space of our problem due to our comprehensive search space of all the transition probabilities for multiple performance criteria. Hence, the basic steepest descent method may become trapped at a local optimum with high probability, i.e., from more than 60% of the random positions where the search starts. We show that, by properly adding

noise in the search process, we can get out of the local optima in practically all the test cases, arriving at the global optimum solution, irrespective of where the search began.

All of the aforementioned assertions are verified by robust numerical and simulation results.

## II. Related Work

The online scheduling of our mobile sensor is controlled by a Markov chain that takes values in a set of PoIs covered by the sensor. When the sensor is at a PoI, e.g., $i$, it determines the next PoI $j$ to visit with transition probability $p_{ij}$. Hence, the online scheduling is effected by a simple coin toss at each decision point. The approach is similar to existing scheduling algorithms such as SFQ [1] and lottery scheduling [3]. However, in SFQ and lottery scheduling, determining the probabilities of the coin toss is simple and based on only the target service rates for the clients. In this paper, we aim at optimizing the transition probabilities to achieve a user- or application-defined notion of the best tradeoff between a number of possibly conflicting performance objectives. This optimization of the transition probabilities through stochastic steepest descent represents a fundamental departure from existing scheduling approaches.

Markov chain Monte Carlo (MCMC) methods [12], [13] can be used to determine the Markov transition probabilities if we target *only* the distribution of the sensor's coverage time among the PoIs, although care is needed to account for the intermediate PoIs visited as the sensor moves from the current PoI $i$ to the next PoI $j$. In this case, the solution that achieves a target coverage-time distribution is not unique and can be optimized for maximum entropy. MCMC methods are not applicable when the tradeoffs between multiple performance objectives are considered.

The case for multiple performance objectives in scheduling problems has been well recognized. Rate, fairness, delay, and jitters are common performance metrics for a range of time-shared resources such as communication and computation bandwidth. Given one of the performance goals, it is known that different algorithms may be designed to achieve different performance for another goal [14]. In sensor networks, the importance of multiple performance objectives and the challenge in balancing between the possibly conflicting objectives have similarly been acknowledged [10]. For the placement of static sensors against contaminations in a WDS, for example, different optimization objectives will lead to different placement strategies. To minimize the delay of detecting contaminants in the WDS, placement of sensors at the periphery, where the contaminants will likely enter the WDS, is called for. To maximize the probability of detection, however, placement in the middle of the WDS may be the most effective.

However, although the existence of diverse performance objectives is well known and individual algorithms may approach their simultaneous achievement differently, the design of a uniform approach to support a range of tradeoffs and optimize the tradeoff according to user/application requirements has been absent in the literature. This paper aims at providing such an approach in optimizing tradeoffs between important objectives in mobile sensor coverage, including the distributions of per-PoI coverage times and exposure times, energy efficiency, and entropy of the coverage schedule. To the best of our knowledge, there is no existing approach that can provide these optimal tradeoffs in sensor coverage problems.

## III. Physical Setup

We consider a mobile sensor that moves among a number of PoIs placed in a geographical area. The sensor has a sensing range of $r$. A PoI, e.g., $i$, is said to be *covered* by the sensor if $i$ is within distance $r$ of the sensor, in which case the sensor can gather useful information about $i$ (e.g., detect any interesting event that happens at $i$). The PoIs are disjoint, because no two PoIs can be covered by the sensor at the same time. However, they are fully connected, which means that a path exists for the sensor to move between any two PoIs. As the sensor travels from PoI $i$ to PoI $j$ on a path, it may pass through (and hence cover) other intermediate PoIs that are geographically situated along the path. We allow the user to specify a target allocation, which is denoted by $\Phi$, of the sensor's coverage time among the PoIs, where $\Phi_i$ is the target fraction of the coverage time that should be received by $i$. The target allocation may reflect the different importance of the PoIs.

The scheduling of our mobile sensor is controlled by a Markov chain, which describes the sensor's travel between the PoIs. We design the Markov chain to minimize a general cost function, which can account for factors such as the distributions of per-PoI coverage and exposure times, an entropy measure that quantifies the randomness of the sensor schedule, information capture about dynamic events, and energy efficiency. One key feature of our framework is that it can possibly incorporate conflicting performance objectives while accounting for the dependencies between the PoIs that were covered due to their geographical placement.

For discussion, we will focus on a cost function, denoted as $U$, that considers the distributions of per-PoI coverage times, per-PoI exposure times, randomness, and energy efficiency of the sensor schedule. Other performance criteria can be included without much difficulty. Our approach expresses the cost function in terms of the transition probabilities of the Markov chain and uses steepest descent to find the optimal transition probabilities. The advantages of our method are its *simplicity* and the *comprehensiveness of the search space*.

### A. Derivation of the Cost Function $U$

We now describe details of the mathematical setup. Suppose there are $M$ PoIs, denoted by the set $S = \{1, 2, \ldots, M\}$. Consider a *time-homogeneous* Markov chain $\{X_n\}_{n\geq 1}$ that takes values in $S$. The sensor moves between the PoI locations (i.e., states in the Markov chain) as the Markov chain state transitions. The transition probabilities are denoted by $\{p_{ij}\}_{i,j\in S}$ : $p_{ij} = P(X_{n+1} = j | X_n = i)$.

To relate the (discrete) transition time step $n$ and the physical elapsed time, we use the following notation.

- $T_{jk}$ is equal to the *sum* of the *travel time* from PoI $j$ to PoI $k$ and the *pause time* $P_k$ at $k$. Note that $T_{jj} = P_j$.
- $T_{jk,i}$ is equal to the time the sensor, upon traveling from $j$ to $k$, passes by PoI $i$. These quantities, in essence, reflect the side effects of coverage imposed by geographical constraints. A PoI, e.g., $i$, can still be covered even if it is not the intended destination of a transition, because the sensor has to go through $i$ to reach the destination, although it may not pass through the center of $i$. We use the following convention: $T_{jj,i} = 0$ for $j \neq i$, and $T_{jj,j} = P_j$ (the pause time at $j$).

With the aforementioned notation, the physical time that elapsed after $N$ state transitions is given by

$$T(N) = \sum_{n=1}^{N} \sum_{j,k} T_{jk} \mathbf{1}_j(X_n) \mathbf{1}_k(X_{n+1})$$

where $\mathbf{1}_i(x) = 1$ if $x = i$; otherwise, it is 0. Furthermore, the total coverage time of PoI $i$ by the sensor is given by

$$C_i(N) = \sum_{n=1}^{N} \sum_{j,k} T_{jk,i} \mathbf{1}_j(X_n) \mathbf{1}_k(X_{n+1}).$$

We now define three other quantities used in our performance metric. The first quantity is $\langle E_i(N) \rangle$—the average *exposure time*—which is the *arithmetic average* (taken in the first $N$ transitions) of the lengths of the *continuous* time intervals during which the PoI $i$ is *out of range* of the sensor. More precisely, let $E_{i,1}, E_{i,2}, \ldots, E_{i,m}$ be the consecutive time intervals during which the sensor is away from $i$ during the first $N$ state transitions. Then

$$\langle E_i(N) \rangle = \frac{E_{i,1} + E_{i,2} + \cdots + E_{i,m}}{m}.$$

The second quantity is entropy. Its main significance is introducing randomness into the mobile sensor schedule for its *unpredictability*. Hence, it is natural to search for the transition rates so that the Markov chain achieves *maximum* entropy. Let $p_{ij}$ be the probability for the sensor to travel from PoI $i$ to PoI $j$ and $\{X_1, X_2, \ldots, X_N\}$ be the sequence of the PoIs visited by the sensor in $N$ steps. Then, the entropy of the sensor schedule is approximated by [15, p. 76]

$$\begin{aligned} \langle H(N) \rangle &= -\frac{1}{N} \ln \mathrm{P}(X_1, X_2, \ldots, X_N) \\ &= -\frac{1}{N} \ln \left( \mathrm{P}(X_1) \Pi_{k=1}^{N-1} p_{X_k X_{k+1}} \right). \end{aligned} \tag{1}$$

The third quantity is $\langle D(N) \rangle$: the average energy cost of the sensor measured as the distance traveled by the sensor. Let $d_{ij}$ indicate the travel cost from PoI $i$ to $j$. The cost of energy of the sensor is given by

$$\langle D(N) \rangle = \frac{\sum_{n=1}^{N} \sum_{j,k} d_{jk} \mathbf{1}_j(X_n) \mathbf{1}_k(X_{n+1})}{T(N)}.$$

With the aforementioned expression, we define the cost function $U$ to be the following long time limit:

$$\lim_{N \to \infty} \left\{ \sum_{i \in S} \frac{1}{2} \alpha_i \left( \frac{C_i(N)}{T(N)} - \Phi_i \right)^2 + \sum_{i \in S} \frac{1}{2} \beta_i \langle E_i(N) \rangle^2 - \frac{1}{2} \eta \langle H(N) \rangle + \frac{1}{2} \rho \langle D(N) \rangle \right\}$$

where $\alpha_i$'s, $\beta_i$'s, $\eta$, and $\rho$ are user-defined constants, and $\Phi_i$'s represent the prescribed distribution of per-PoI coverage times.

The intuition of the aforementioned cost function is given as follows. The part with the $\alpha_i$ terms measures the discrepancy between the actual and prescribed distributions of per-PoI coverage times. The part with the $\beta_i$ terms measures the average per-PoI exposure time. The part with the $\eta$ term measures the randomness of the sensor schedule. The part with the $\rho$ term measures the energy cost of the sensor schedule. Intuitively, minimizing the function $U$ attempts to reduce the coverage-time discrepancy, the expected exposure time, and the energy cost while increasing the randomness of the schedule. However, the coverage time, exposure time, and energy cost measures are generally antagonistic; therefore, the optimal choice of the transition probabilities is the result of a delicate tradeoff between them. The parameters $\alpha_i$'s, $\beta_i$'s, $\eta$, and $\rho$ express the relative importance of the four measures, and they can be adjusted by the user.

### *B. Formula of the Cost Function U*

Our next goal is to express the aforementioned cost function as an explicit function of the $p_{ij}$'s, which completely characterize the Markov chain. For this case, we will assume, for the rest of this paper, that the Markov chain is ergodic.

Let $\{\pi_i\}_{i \in S}$ be the stationary distribution of the Markov chain. By ergodicity, we have

$$\lim_{N \to \infty} \frac{T(N)}{N} = \sum_{j,k} \pi_j p_{jk} T_{jk}, \lim_{N \to \infty} \frac{C_i(N)}{N} = \sum_{j,k} \pi_j p_{jk} T_{jk,i}.$$

Hence, the long-term average of the coverage time distribution $\bar{C}_i$ can be defined and computed as

$$\bar{C}_i = \lim_{N \to \infty} \frac{C_i(N)}{T(N)} = \frac{\sum_{j,k} \pi_j p_{jk} T_{jk,i}}{\sum_{j,k} \pi_j p_{jk} T_{jk}}. \tag{2}$$

To simplify the mathematical expression for the long time limit $\lim_{N \to \infty} \langle E_i(N) \rangle$, we will make the following assumptions, which will not qualitatively change our conclusions.

1) If a PoI is simply passed by while the sensor travels between two other PoIs in a transition, then the passing-by is not considered a return visit.
2) We assume that the time elapsed during each travel is always equal to one unit of time—this assumption is only enforced in the computation of the $\langle E_i \rangle$'s. [The exact analytical expression for the exposure times, without the aforementioned assumptions, are derived in Remark 4; see (12). However, its usage will unnecessarily increase

the computational load without giving any new interpretation.] With this simplification, physically, each exposure time segment $\{E_{i,l}\}_{l=1,2,\ldots}$ is measured from the PoI location (e.g., $i$) immediately after the sensor has *left* $i$. Let $R_{ji}$ be the *expected value* of the time the sensor takes to reach (or return to) $i$ from $j$. This quantity is also traditionally called the expected *first passage time*. Because the probability of traveling from $i$ to $j$ is given by $p_{ij}$, we have the following expression for the long time average of the exposure times:

$$\bar{E}_i = \lim_{N\to\infty} \langle E_i(N) \rangle = \frac{\sum_{j\neq i} p_{ij} R_{ji}}{\sum_{j\neq i} p_{ij}} = \frac{\sum_{j\neq i} p_{ij} R_{ji}}{1 - p_{ii}}. \quad (3)$$

For the entropy of the Markov chain, the long time limit of the expression (1) is given by [15, p. 76]

$$\bar{H} = -\sum_i \pi_i \left[ \sum_j p_{ij} \ln p_{ij} \right]. \quad (4)$$

In addition, we can give the following analytical formula for the average energy cost of the sensor:

$$\bar{D} = \sum_i \pi_i \left[ \sum_{j\neq i} p_{ij} d_{ij} \right] \quad (5)$$

where $d_{ij}$ indicates the travel cost from PoI $i$ to $j$ (for $j \neq i$). The factor $\pi_i$ represents the stationary distribution of the sensor locations in the long run.

With the aforementioned expression, the cost function $U$ can be written as

$$\begin{aligned} U = & \sum_{i\in S} \frac{1}{2} \alpha_i \left[ \sum_{j,k} \pi_j p_{jk} (T_{jk,i} - \Phi_i T_{jk}) \right]^2 \\ & + \sum_{i\in S} \frac{1}{2} \beta_i \left[ \frac{\sum_{j\neq i} p_{ij} R_{ji}}{1 - p_{ii}} \right]^2 + \frac{1}{2} \eta \sum_{i\in S} \pi_i \left[ \sum_{ij} p_{ij} \ln p_{ij} \right] \\ & + \frac{1}{2} \rho \sum_{i\in S} \pi_i \left[ \sum_{j\neq i} p_{ij} d_{ij} \right]. \end{aligned} \quad (6)$$

Note that, in (6), the cost of matching is expressed in absolute time instead of a relative fraction in (2), because it has an easier computational expression. Qualitatively, both of them measure the discrepancy between the actual and desired coverage profiles.

*Remark 1—Search Space of the Optimization:* In the current setting, we search within the space of *all* transition probabilities. This approach is in contrast to most problems in stochastic control, in which the search or *action space* is usually parameterized by a finite number of parameters. Thus, the novelty of our approach is that the solution gives the true optimal among the class of all Markov chains. Because of the comprehensiveness of our search space, we are faced with the following two major difficulties, although our explicit analytical formulas have greatly simplified the problem.

1) *Superlinear growth of the number of search variables*. Let $n$ be the total number of PoIs. The number of transition probabilities $p_{ij}$'s is of the order $O(n^2)$. Because of this high number of search directions, the optimization algorithm can be extremely time consuming.
2) Large number of local minima. Our cost function $U$ is *nonlinear*, and as indicated by the large number of local optima in the solution space illustrated in the simulation results, our cost function $U$ is also highly *nonconvex*. Hence, appropriate stochastic perturbation is essential for the search algorithm to get out of the local optimal solutions.

Even with the aforementioned difficulties, our algorithm, which is *adaptive* (by incorporating changing time steps) and *stochastic* (by the addition of noise), shows that optimal solutions can be achieved within realistic time budgets.

*Remark 2—Utility Viewpoint of the Cost Function:* We minimize a cost function in our problem formulation. This minimization of a cost function is equivalent to the maximization of a corresponding utility function. According to this utility viewpoint, we assume that the user derives a "utility" (i.e., level of satisfaction) for each achieved value of a metric. Our procedure then optimizes the aggregate utility of all the metrics.

In the aforementioned derivations, because we weight the entropy and energy metrics by a constant factor in the cost function, we implicitly assume that the corresponding utility functions of these two metrics are *linear*. On the other hand, we use a square function, which is convex, for the terms of matching and unfairness in the cost function. This approach implicitly assumes that the corresponding utility functions of these two metrics are *concave*—a convex function in the cost minimization is equivalent to a concave function in the utility maximization. Concave utility functions are meant to model the law of diminishing marginal returns for certain quantities. Hence, the weights of different metrics in the cost function can all be obtained directly from the user's utility functions for these metrics. For the linear functions, the weights give the slopes of the functions. For the concave functions, the weights give the scaling factors of the square functions in the cost minimization.

Our problem formulation can easily be adapted to consider general forms of utility functions, provided that the functions are differentiable. For example, the optimization objective can be in the form of $\sum_i u_i(v_i)$, where $i$ is a given metric, $u_i$ is its utility function in general form, and $v_i$ is the value of the metric achieved, given the control parameters (i.e., transition probabilities). In addition, we can account for certain constraints on performance metrics if a general utility function is used. For example, to enforce a maximum constraint on exposure time, we can define the utility to be zero if the achieved exposure time is higher than the constraint.

### C. Analytical Representation of $U$ Using Generalized Inverse

Although (6) for the cost function is highly explicit, it still requires the computation of the stationary distribution $\{\pi_i\}$ and

the first passage time matrix $\{R_{ij}\}$. To efficiently compute them, we employ the concept of *generalized inverse*. In the following paragraphs, we review the basic properties of this mathematical object, with particular application toward Markov transition matrices. The material is mainly based on [16].

Let $P = \{p_{ij}\}_{i,j\in S}$ be an ergodic Markov transition matrix. Consider the generalized inverse $A^\sharp$ of $A = I - P$, which is defined as the matrix that satisfies

$$AA^\sharp A = A, \quad A^\sharp AA^\sharp = A^\sharp, \quad \text{and} \quad A^\sharp A = AA^\sharp.$$

The existence and computation of $A^\sharp$ is given by [16, Th. 2.1 and 5.1].

Using $A^\sharp$, the stationary distribution $\pi = \{\pi_i\}$ and the expected first passage time matrix $R = \{R_{ij}\}$ can be expressed as

$$W = I - AA^\sharp, \qquad ([16, \text{Th. } 2.3]) \tag{7}$$

$$R = \left(I - A^\sharp + JA^\sharp_{dg}\right), \quad ([16, \text{Th.}3.3,\ 3.6]) \tag{8}$$

where $W$ is the matrix such that all of its rows are the stationary distribution $\pi$, $J$ is the $M \times M$ square matrix, with all of its entries equal to one, $A^\sharp_{dg}$ is the diagonal matrix that consists of the diagonal entries of $A^\sharp$, and $D$ is the diagonal matrix such that $D_{ii} = (1/\pi_i)$. Note that we can have such a simple expression (8) for the $R_{ij}$'s due to the simplifying assumptions that we have for the exposure times.

The generalized inverse is also related to the commonly known *fundamental matrix* $Z = (I - P + W)^{-1}$ by the formula [16, Th. 3.1]

$$Z = I + PA^\sharp. \tag{9}$$

Using $Z$, $R$ can then be expressed as

$$R = (IZ + JZ_{dg})D, \text{ where } R_{ij} = \frac{\mathbf{1}_i(j) - z_{ij} + z_{jj}}{\pi_j}. \tag{10}$$

The aforementioned result is the one used in the actual numerical computation.

*Remark 3:* Note that there are constraints for the $p_{ij}$'s. First, they must satisfy that condition that $0 \le p_{ij} \le 1$ and $\sum_{j\in S} p_{ij} = 1$, i.e., they lie in a higher dimensional polytope. Although this property is manageable within the realm of nonlinear optimization theory, to simplify the computational algorithm, we add a small penalization term $\epsilon$ to handle the case of $p_{ij} = 0$ or 1. More specifically, we consider the following penalized version $U_\epsilon$ of $U$:

$$\begin{aligned} U_\epsilon = &\sum_i \frac{1}{2}\alpha_i \left[\sum_{j,k} \pi_j p_{jk}(T_{jk,i} - \Phi_i T_{jk})\right]^2 \\ &+ \sum_i \frac{1}{2}\frac{\beta_i}{\pi_i^2(1-p_{ii})^2}\left[\sum_{j\neq i} p_{ij}\left(\mathbf{1}_j(i) - z_{ji} + z_{ii}\right)\right]^2 \\ &+ \frac{1}{2}\eta\sum_i \pi_i\left[\sum_{ij} p_{ij}\ln p_{ij}\right] + \frac{1}{2}\rho\sum_i \pi_i\left[\sum_{j\neq i} p_{ij}d_{ij}\right] \\ &- \sum_{ij}\frac{1}{\epsilon}[\ln p_{ij}](\epsilon - p_{ij})^2\mathrm{sgn}(\epsilon - p_{ij}) \\ &- \sum_{ij}\frac{1}{\epsilon}\left[\ln(1-p_{ij})\right](1 - \epsilon - p_{ij})^2\mathrm{sgn}(p_{ij} - 1 + \epsilon) \end{aligned} \tag{11}$$

where $\mathrm{sgn}(x)$ is equal to one for $x \ge 0$; otherwise, it is zero. Note that $U_\epsilon \longrightarrow +\infty$ as any of the $p_{ij}$ converges to 0 or 1. This condition is prohibited by the steepest descent algorithm, whose purpose is to minimize $U$. The constraint $\sum_j p_{ij} = 1$ can easily be handled by a projection method, which will be described later.

Second, in the current setting, all transitions between any two PoIs are possible. However, if some of the transitions are not allowed or are physically infeasible, we can simply set the corresponding $p_{ij}$'s to be zero or small.

*Remark 4—General Formula for the $R_{ij}$'s:* Here, we derive the exact analytical formulas for $R_{ij}$'s and the exposure times without the simplifying assumption (2) in Section III-B. Recall that the exposure time for a PoI is defined as the average of the time segments during which the sensor is away from the PoI. This is closely related to the *first passage time* $R_{ij}$ used in the analysis of Markov chains problems.

Define $T_{ji} = \min\{t : X_t = i, X_0 = j\}$ and $R_{ji} = ET_{ji}$, then, $\bar{E}_i = (1/1 - p_{ii})\sum_{j\neq i} p_{ij}R_{ji}$ [see (3)]. Now, let $t_{ij}$ be the travel time from PoI $i$ to $j$ and $t_{ij,k}$ be the first time the sensor will pass by PoI $k$ upon traveling from $i$ to $j$. (Note that $k$ is not necessarily equal to $j$ due to the underlying geographical configuration.) For convenience, we set $t_{ij,k}$ to be 0 if $k$ is actually not at all passed by the sensor. Then, we have the following expression (for $k \neq l$):

$$\begin{aligned} R_{kl} = p_{kl}t_{kl} &+ \sum_{m\neq l, t_{km,l}\neq 0} p_{km}t_{km,l} \\ &+ \sum_{m\neq l, t_{km,l}=0} p_{km}(t_{km} + R_{ml}). \end{aligned} \tag{12}$$

In principle, the $R_{ij}$'s can be solved from the aforementioned system of linear equations (see [17, p. 78]) for deriving and solving the traditional first passage times. The approach will certainly increase the computational load. Based on our simulation results, the results are not changed in any qualitative sense by using the simplifying assumption (2).

## IV. Steepest Descent Optimization

This section describes the use of *steepest descent* to search for the minimum of $U_\epsilon$. Let $U = U(q_1, q_2 \ldots, q_l)$ be some function, depending on the variables $q_1 \ldots q_l$. We would like to search for the minimum of $U$ by evolving the variable $q_i$'s. For simplicity, let $Q(t) = (q_1(t), \ldots, q_l(t))$ denote the values of the $q_i$'s at time $t$. Then, we compute

$$\frac{d}{dt}U\left(Q(t)\right) = \nabla U\left(Q(t)\right)\frac{dQ(t)}{dt}.$$

Now, if we choose $(dQ(t)/dt) = -\nabla U(Q(t))$, then

$$\frac{d}{dt}U\left(Q(t)\right) = -\left|\nabla U\left(Q(t)\right)\right|^2 < 0$$

i.e., $U$ is *decreasing*. Note that steepest descent, as implemented above, could only lead to a *local minimum* or *critical point*. In Section V, we will show how *stochastic perturbation* can solve this problem.

To apply the aforementioned formalism for the minimization of the cost function, we write $U$ as

$$U = U(\pi, Z, P) = U\left(\pi(P), Z(P), P\right) = U(P)$$

where $P = (p_{ij})$ is the transition matrix. Then

$$\frac{d}{dt}U\left(P(t)\right) = \sum_i \frac{\partial U}{\partial \pi_i}\frac{d\pi_i(t)}{dt} + \sum_{jk}\frac{\partial U}{\partial z_{jk}}\frac{dz_{jk}(t)}{dt} + \sum_{jk}\frac{\partial U}{\partial p_{jk}}\frac{dp_{jk}(t)}{dt}.$$

The expressions for $(\partial U/\partial \pi_i)$, $(\partial U/\partial z_{ij})$, and $(\partial U/\partial p_{ij})$ can easily be derived by straightforward partial differentiation. The formulas for $(d\pi_i(t)/dt)$ and $(dz_{jk}(t)/dt)$ can be obtained by considering the *perturbation analysis* of Markov chain [18]. Precisely

- Based on [18, eqs. (15), (17), and (23c)]

$$\frac{d\pi}{dt} = \pi \dot{P} Z, \text{ or component-wise, } \frac{d\pi_i}{dt} = \sum_{k,j}\pi_k z_{ji}\dot{P}_{kj}.$$

- Based on [18, eqs. (16), (18), and (23d)]

$$\frac{dZ}{dt} = Z\dot{P}Z - W\dot{P}(Z^2), \text{ or component-wise,}$$

$$\frac{dz_{ij}}{dt} = \sum_{kl}\left[z_{ik}z_{lj} - \pi_k(Z^2)_{lj}\right]\dot{P}_{kl}$$

where $(Z^2)_{lj} = \sum_m z_{lm}z_{mj}$.

Based on the aforementioned results, if we let $[D_P U]_{kl}$ be the following expression:

$$\left[\sum_i \pi_k z_{li}\frac{\partial U}{\partial \pi_i}\right] + \left[\sum_{ij}\frac{\partial U}{\partial z_{ij}}\left[z_{ik}z_{lj} - \pi_k(Z^2)_{lj}\right]\right] + \frac{\partial U}{\partial p_{kl}} \tag{13}$$

then we have

$$\frac{dU}{dt} = \sum_{kl}[D_P U]_{kl}\dot{P}_{kl} = \langle D_P U, \dot{P}\rangle.$$

We can simply take $\dot{P}_{kl} = -[D_P U]_{kl}$. However, as aforementioned, $P(t)$ must be a transition matrix at all $t$, i.e., it has to satisfy for all $j$, $\sum_k p_{jk}(t) = 1$. To accommodate this condition, we *orthogonally project* $D_P U$ onto this linear subspace, i.e.,

$$\frac{dU}{dt} = \left\langle \Pi[D_P U],\ \dot{P}\right\rangle$$

and then take

$$\dot{P}_{kl} = -\left(\Pi[D_P U]\right)_{kl}.$$

The formula for the projection $\Pi$ is given by

$$\Pi_{ij} = U_{ij} - \frac{\sum_k U_{ik}}{M} \quad \text{for all } i, j \tag{14}$$

where $U_{ij}$ and $\Pi_{ij}$ are the entries of $[D_P U]$ and $\Pi[D_P U]$, respectively. Note that $\sum_j \Pi_{ij} = 0$, i.e., the sum of each row of $\Pi$ is *zero*. Hence, the property that the quantities represent a transition probability matrix is preserved at all time.

## V. Computational Algorithm

The aforementioned steepest descent algorithm will be implemented as follows.

1) Start with an arbitrary ergodic transition probability $P$.
2) Compute $[D_P U]$ (13) and its projection $\Pi[D_P U]$ (14).
3) Set $V = -\Pi[D_P U]$.
4) Set the new $P$ as $P + V * \triangle t$, where $\triangle t$ is some small time step.
5) For the new $P$, compute $\pi$, $Z$, and $R$ by (7), (9), and (10).
6) Go back to step 2) or stop if the optimal is attained (within some tolerance level).

Note that the ergodicity of $P$ is ensured by the finiteness of all the $R_{ij}$'s, which is, indeed, preserved during the whole computation.

In the rest of this paper, we will study the following variants of the steepest descent algorithm.

V1: *Basic algorithm*. All the $p_{ij}$ values are initially set to $(1/M)$, where $M$ is the number of PoIs. A constant time step $\Delta t$ is used. This provides a basic test for the validity of our steepest descent algorithm.

V2: *Random initial data*. In this case, the initial values of the $p_{ij}$'s are random. They are set to $(rand \times rem/M)$, except for the entries in the last column, which will be set to $rem$, where $rand$ is a random number between 0 and 1, $rem$ is the remaining probability for $p_{ij}$'s within a row (as the summation of a row should be one), and $M$ is the number of PoIs. This test will ensure that our algorithm is stable with respect to the initial condition.

V3: *Adaptive time step* $\Delta t$. With the gradient information $\nabla U$ in the current transition matrix $P$, the optimal time step $\Delta t_*$ is chosen as follows:

$$\Delta t_* = \min_{s>0} U\left(P - s\nabla U(P)\right).$$

The boundaries of $\Delta t$ are first determined with respect to the constraint that $0 \le p_{ij} \le 1$ after the update. Because it is unclear how the utility changes within the range of $\Delta t$, a conservative *trisection method* (wherein only one subsection is removed as we tighten the bounds) is used to tighten the range until $\Delta t_*$ is found. The algorithm terminates when $\Delta t_* = 0$. We define the terminating point as a *local optimum* when, in fact, there exists another terminating point with a lower cost.

V4: *Stochastic perturbations*. This step ensures that our algorithm will not get stuck at a local minimum of the cost function, which is shown to be essential in Section VII. The $[D_P U]$ values are perturbed by mean-zero Gaussian noise with standard deviation $\delta$. $\Delta t_*$ is then computed using the perturbed $[D_P U]$. If $\Delta t_* = 0$, then $\Delta t = rand$, where $rand$ is a random number between the boundaries of $\Delta t$, which are determined with respect to the constraint that $0 \leq p_{ij} \leq 1$ after the update. The updates to $p_{ij}$'s are accepted if the computed utility is better than the original values; otherwise, they are accepted with probability $e^{(-\Delta_U / k \times \log(count))}$, where $\Delta_U$ is the worsening in cost $U$ after normalization by the *best cost* computed so far, $count$ is the number of iterations already performed by the algorithm, and $k$ is a constant parameter. The normalization is important, because it allows us to determine how high the algorithm should "jump" (in trying to get out of a local optimum) without knowing the range of the cost function $U_\epsilon$ (which is frequently unknown beforehand). Intuitively, the algorithm accepts unattractive changes with higher probability at the beginning of the search, and this probability decreases as the algorithm proceeds. This way of accepting the updates is similar to the cooling process in simulated annealing. The optimality and convergence of simulated annealing has a strong theoretical basis and will be discussed in Section VI. However, convergence, in practice, is usually much faster, as we show in Section VII-B.

Henceforth in this paper, we will use the *steepest descent algorithm* to refer to our general solution approach. In addition, we will call variant V1 of the algorithm the *basic algorithm*, the basic algorithm modified by variants V2 and V3 the *adaptive algorithm*, and the basic algorithm modified by all of variants V2–V4 the *stochastically perturbed algorithm* or, simply, the *perturbed algorithm*. We have verified the performance of the steepest descent by using the transition matrices computed by the algorithm at each iteration to drive a corresponding simulation of the mobile sensor's coverage. We found that the algorithm can achieve good tradeoffs between the coverage time, exposure time, entropy, and energy efficiency metrics, as the weightings of the four parameters in the cost function are varied. In addition, the adaptive algorithm can speed up convergence to the final solution, and the perturbed algorithm can additionally converge to the globally optimal steady state in practically all of the scenarios.

## VI. Convergence Issues of the Perturbed Algorithm

In this section, we address the question of convergence of our algorithms. In essence, our goal is to optimize some objective functionals. The most intricate part is the presence of a large number of *local minimizers*. Such a setting can be handled by the classic method of *simulated annealing* [19]. Our approach is in direct analog with this method. The dynamics simulated by our algorithm can formally be modeled by

$$dP(t) = -\nabla U\left(P(t)\right)\, dt + \sqrt{T(t)} dW(t) \tag{15}$$

where the first term on the right corresponds to the steepest descent, whereas the second term introduces noise to stick the state out of the local minima. The key is the choice of the *temperature* $T(t)$, which should follow some *cooling schedule* to produce realistic results. The key is to let $T(t)$ converge to zero as time goes to infinity—too fast a rate leads to getting stuck in local minima, whereas too slow a rate implies slow convergence of the overall algorithm. There has been much theoretical work that addresses this question [20], [21]. It has been established that, if $T(t)$ satisfies the condition

$$T(t) \geq \frac{C}{\log(1+t)} \tag{16}$$

for some large-enough $C$, then the solution $P(t)$ of (15) will converge to the global minimum of $U$. The actual choice of $C$ depends on the *shape* of and the *depths of the local minima* of $U$. The condition (16) is, in fact, optimal [21]. However without knowing *a priori* the properties of $U$, it will be difficult to obtain the *optimal* choice of $C$. Hence, to avoid delving into the theoretical issues, we resort to our simulation algorithms. In fact, many practical simulations work better than the theoretical prediction [22, pp. 311–316]. Our results also indicate so. We will also refer to [23] for some recent surveys about the convergence and the choice of cooling schedules in simulated-annealing-type algorithms.

## VII. Performance Measurements

In this section, we study the performance of the steepest descent algorithm. For simplicity of exposition, we consider the case that the $\alpha_i$'s and $\beta_i$'s in (11) are all equal, i.e., $\alpha_1 = \alpha_2 = \cdots = \alpha_M = \alpha$ and $\beta_1 = \beta_2 = \cdots = \beta_M = \beta$. We now write the cost function in (6) as

$$U = \frac{1}{2}\alpha \Delta C + \frac{1}{2}\beta \bar{E} - \frac{1}{2}\eta \bar{H} + \frac{1}{2}\rho \bar{D} \tag{17}$$

where we further define the coverage time deviation $\Delta C$ as

$$\Delta C = \sum_{i \in S} \left[ \sum_{j,k} \pi_j p_{jk} (T_{jk,i} - \Phi_i T_{jk}) \right]^2$$

and the average exposure time $\bar{E}$ as

$$\bar{E} = \sum_i \bar{E}_i^2 = \sum_{i \in S} \left[ \frac{\sum_{j \neq i} p_{ij} R_{ji}}{1 - p_{ii}} \right]^2.$$

In addition, recall the formulas for the average entropy $\bar{H}$ and energy cost $\bar{D}$ in (4) and (5), respectively.

We will study the following performance aspects of the steepest descent algorithm: 1) the stability and adaptivity of the steepest descent algorithm (see Sections VII-A–C); 2) the tradeoff between different performance metrics (see Sections VII-D and E); 3) the performance of actual Markov chain simulations of the coverage schedules (see Section VI-F).

In our experiments, we use the four topologies shown in Fig. 1. Each PoI $i$ in the topologies is the center of the cell labeled with $i$, and its targeted share of coverage time $\Phi_i$ is

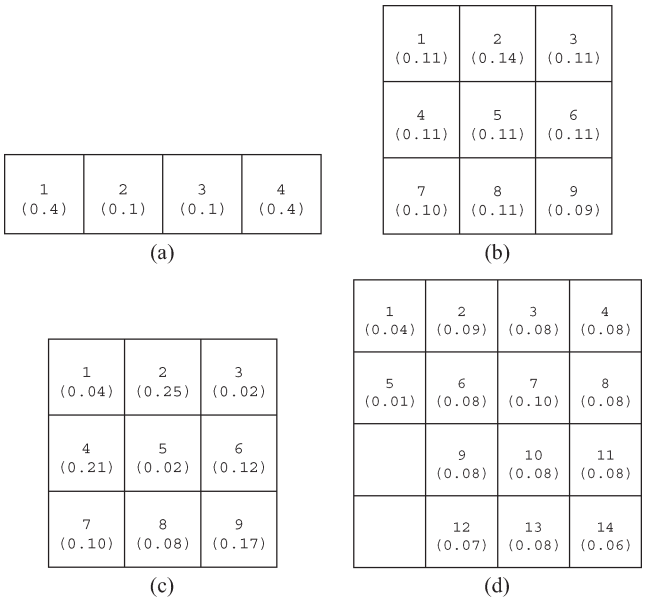



Fig. 1. Target per-PoI shares of coverage times $\Phi_i$ in four simulation topologies. (a) Topology 1 ($1 \times 4$). (b) Topology 2 ($3 \times 3$). (c) Topology 3 ($3 \times 3$). (d) Topology 4 ($4 \times 4$).

given in parentheses. We assume that, in traveling from $i$ to $j$, the sensor uses the straight-line path between $i$ and $j$.

### A. Existence of Numerous Local Optima

We characterize the search space by comparing the adaptive algorithm (i.e., the basic algorithm modified by variants V2 and V3 in Section V) and the perturbed algorithm (i.e., the basic algorithm modified by variants V2–V4 in Section V) under Topology 4 in Fig. 1. In computing the cost function, we illustrate the cases of considering the following three conditions: 1) $\bar{E}$ only $(\alpha = 0, \beta = 1, \eta = 0, \rho = 0)$; 2) both $\Delta C$ and $\bar{E}$ $(\alpha = 1, \beta = 0.0001, \eta = 0, \rho = 0)$; and 3) $\Delta C$ only $(\alpha = 1, \beta = 0, \eta = 0, \rho = 0)$ while having $\epsilon = 0.0001$ and $k = 10000$. The cumulative distribution functions (cdf's) of the achieved costs $U_\epsilon$ by the adaptive and perturbed algorithms are shown in Fig. 2.

Based on the figures, observe that the adaptive algorithm hits a large number of local optima in its search, from which it cannot discover a better cost $U$ in the computed descent direction. The presence of numerous local optima makes it essential to adopt noise in the search process, as detailed by the design of the perturbed algorithm in Section V. In Fig. 2, notice that the perturbed algorithm achieves much better performance than the adaptive algorithm, as evidenced by the extremely sharp rise of the cdf at the global optimum solution. Indeed, the perturbed algorithm computes a solution that is extremely close to, if not exactly the same as, the global optimum in all the runs of the experiment, irrespective of the initial search parameters.

### B. Adaptivity and Stability of the Algorithm

In this set of experiments, we trace the evolution of the cost function $U$ as the transition matrix changes in each iteration of the steepest descent algorithm. We verify that steepest descent can progressively improve the transition matrix by reducing $U$ over the iterations. To demonstrate that the algorithm does not get stuck at a local minimum, we show convergence to the same transition matrix and $U$ value from different initial $p_{ij}$'s using the perturbed algorithm in Section V. To simplify the study, we have $\eta = \rho = 0$.

We study the stability and adaptivity of the algorithm under two settings, with $\epsilon = 0.0001$ and $\Delta t = 0.000001$ (for further results, see our technical report [24]).

*$\Delta C$ Only $(\alpha = 1, \beta = 0)$:* We evaluate the case when the cost function considers only the coverage time deviation metric. The results of the basic algorithm using Topology 3 are depicted in Fig. 3(a). In addition, we evaluate the perturbed algorithm with different initial values of $p_{ij}$'s (i.e., the initial $p_{ij}$'s are generated using different random seeds), and the results are shown in Fig. 3(b).

*Both $\Delta C$ and $\bar{E}$:* We further consider the case where the coverage time deviation metric and the exposure time metric have nonzero weights. The results for different $\alpha$ and $\beta$ values for the basic algorithm using Topology 1 are shown in Fig. 4.

*Observations:* Based on the simulation results, we deduce the following.

1) During the optimization, the cost function generally decreases until it reaches a stable value, as depicted in Figs. 3 and 4. However, the marginal reduction in the cost becomes smaller as the number of iterations becomes larger.
2) Using different random seeds to generate the initial $p_{ij}$'s does not affect the performance of the perturbed algorithm [see Fig. 3(b)], because it will converge to the same stable values of the cost function. However, the convergence and its time will be affected. Moreover, the perturbed algorithm can converge extremely close to the same optimal costs in all these situations, i.e., the algorithm is not trapped at a local minimum. This result also holds for more complex topologies.

### C. Effect of $\delta$ in the Perturbed Steepest Descent Algorithm

Recall that $\delta$ is the standard deviation of the zero-mean Gaussian noise used to perturb the $[D_P U]$ values during the steepest descent algorithm. In this set of experiments, we study the effect of $\delta$ on the performance of the transition matrix obtained at the end of the simulations.

While keeping $\epsilon = 0.0001$ and $\Delta t = 0.000001$, Fig. 5 shows the results when we consider only the coverage and exposure metrics, i.e., $\alpha = 1$, $\beta = 0.0001$, $\eta = 0$, and $\rho = 0$. Fig. 6 shows the results when we consider the coverage, exposure, and entropy metrics, i.e., $\alpha = 1, \beta = 0.0001, \eta = 0.001$, and $\rho = 0$. Fig. 7 shows the results when we consider the coverage, exposure, and energy efficiency metrics, i.e., $\alpha = 1$, $\beta = 0.0001$, $\eta = 0$, and $\rho = 0.001$.

*Observations:* As shown in the figures, the value of $\delta$ affects the performance of the transition matrix obtained at the end of the simulations. On the other hand, a larger $\delta$ does not necessarily improve the performance. For example, Fig. 5 shows that 0.01 is the best $\delta$ among all the simulated values, whereas in Fig. 6, $\delta = 100$ gives the best results.

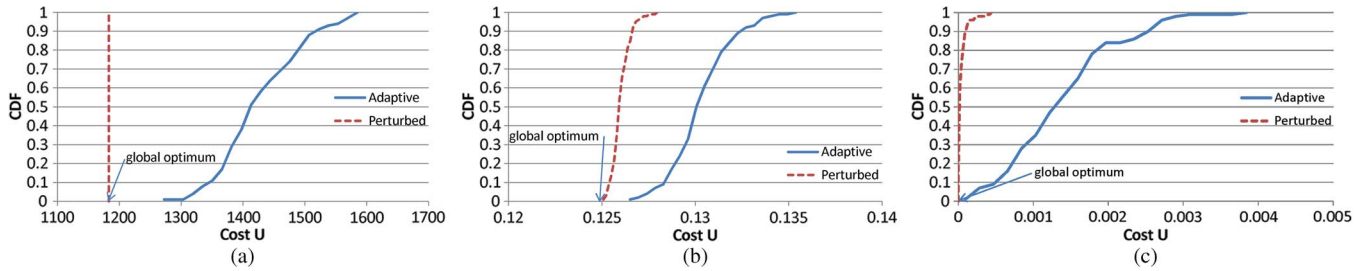



Fig. 2. CDFs of the achieved cost $U_\epsilon$ in a large number of runs by the adaptive and the perturbed algorithms. Topology 4. (a) $\bar{E}$ only ($\alpha = 0, \beta = 1$). (b) $\Delta C$ and $\bar{E}$ ($\alpha = 1, \beta = 0.0001$). (c) $\Delta C$ only ($\alpha = 1, \beta = 0$).

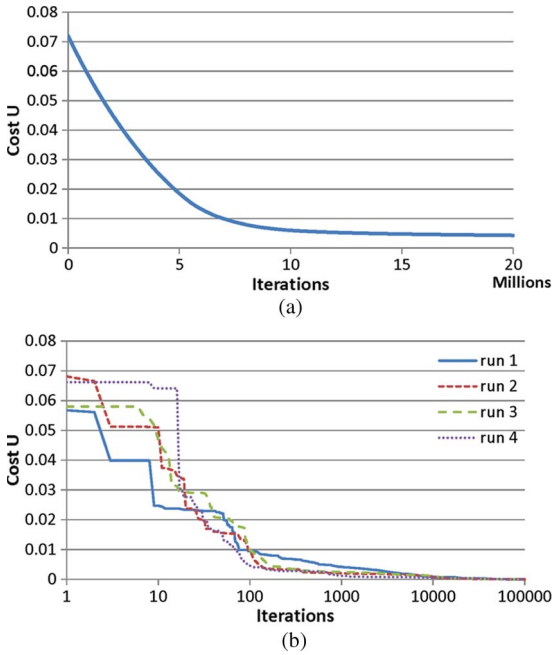



Fig. 3. Performance of the (a) basic and (b) perturbed algorithms with different initial $p_{ij}$'s: Cost function value $U$ as a function of the iteration number. ($\alpha = 1$, $\beta = 0$, Topology 3.) (a) Basic algorithm. (b) Perturbed algorithm with different initial $p_{ij}$'s.

### D. Tradeoff Between the $\delta C$ and $\bar{E}$ Metrics

We study the characteristics of the transition matrix $\{p_{ij}\}$ and the steady-state distribution $\pi_i$ of the Markov chain when we vary $\alpha$ and $\beta$ in (17) using Topology 1 in Fig. 1. To simplify the comparison, we have $\eta = \rho = 0$. We use $\epsilon = 0.0001$ and $\Delta t = 0.000001$ and vary $\beta$ from 1 to 0.0000001 while keeping $\alpha = 1$. We also study the following two extreme cases: 1) $\alpha = 1$ and $\beta = 0$ (we are concerned with only the $\Delta C$ metric) and 2) $\alpha = 0$ and $\beta = 1$ (we are concerned with only the $\bar{E}$ metric). The results for $\bar{C}_i$ and $\bar{E}_i$ [as defined in (2) and (3)] are shown in Tables I and II, respectively.

*Observations:* Based on the simulation results, we deduce that, when we reduce $\beta$, $\bar{C}_i$ (the fraction of time PoI $i$ is covered) will more closely approximate the target share of coverage time, whereas $\bar{E}_i$ (the average exposure time of $i$) grows, as shown in Tables I and II. This is because, upon reducing $\beta$, we are less concerned with the exposure time of the PoI and are more focused on the target coverage time. We can also observe that $\bar{C}_i$ and $\bar{E}_i$ are not significantly changed when $\beta$ is large. It is because, for Topology 1, the magnitude of $\bar{E}$ is significantly larger than $\Delta C$. Hence, even if $\beta$ (weight of $\bar{E}$) is reduced from 1 to 0.01, the exposure time component still dominates the cost function, and the resulting $p_{ij}$ and $\pi_i$ do not significantly change. We notice that, as the size of the topology grows, $\bar{E}$ grows and we will need to further reduce $\beta$ before the coverage time component in $U$ will have an observable effect on $p_{ij}$ and $\pi_i$.

### E. Interactions Between Performance Metrics

In general, the four performance metrics that we consider are interdependent. In this set of experiments, we illustrate their interactions by varying the weight on one performance metric while keeping the weights of the other metrics constant. The averaged results of 20 simulation runs are shown in Fig. 8. Notice that, because the figure plots the values of different metrics, the $y$-axis for unfairness is shown on the *right* vertical axis instead of the left vertical axis.

Fig. 8(a) shows that, as we increase the weight on matching, its cost drops in general. The effect is more obvious when the weight is high. However, when the weight of entropy is really small, its cost may rise as its weight initially increases. It is because, when the weight is small, it has little influence on the overall cost function. As we put more weight on matching, the performance of both unfairness and entropy gets worse because to improve matching, the mobile node picks its next destination with transition probabilities that match the target allocation in steady state, and the probabilities are unlikely to be uniform, which benefits the unfairness and entropy metrics.

Fig. 8(b) shows that, as we increase the weight on unfairness, its cost drops in general, whereas those of matching and energy increase. It is because to reduce unfairness, the mobile node has to more frequently visit all of the PoIs. Fig. 8(c) shows that, as we increase the weight on entropy, generally, its cost drops, as well as that of unfairness, whereas those of matching and energy increase. It is because, to improve entropy, the sensor picks its next destination with a closer to uniform distribution over all the possible PoIs. Fig. 8(d) shows that, as we increase the weight on energy, its cost drops in general, whereas those of unfairness and entropy increase because, to improve energy use, the sensor has to move less and visit closer PoIs with higher probability.

### F. Performance of Actual Markov Chain Simulations

We evaluate the performance of actual sensor schedules, because they are controlled by the Markov chain with the

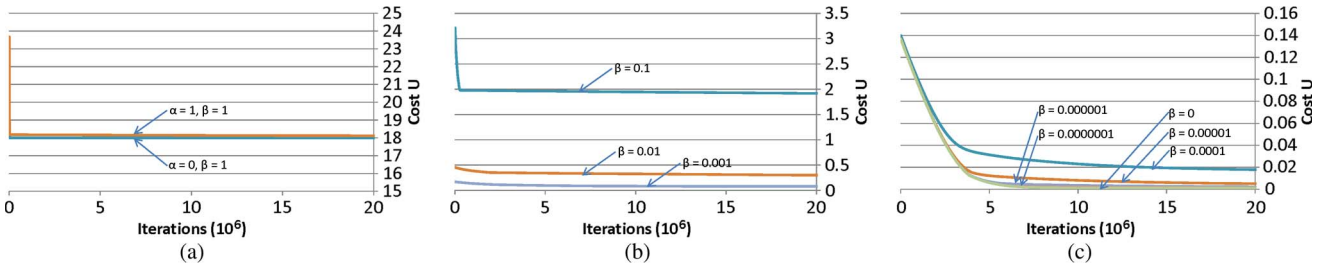



Fig. 4. Cost function value $U$ in terms of the iteration number. ($\epsilon = 0.0001$, $\Delta t = 0.000001$, Topology 1.) (a) $\beta = 1$. (b) $\alpha = 1$. (c) $\alpha = 1$.

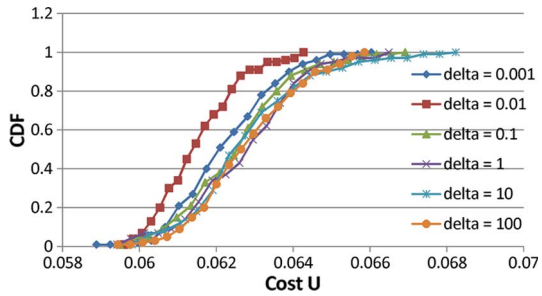



Fig. 5. CDFs of the achieved cost $U_\epsilon$ in 100 simulation runs by the perturbed algorithm with different $\delta$ values. ($\alpha = 1$, $\beta = 0.0001$, $\eta = 0$, $\rho = 0$, Topology 3.)

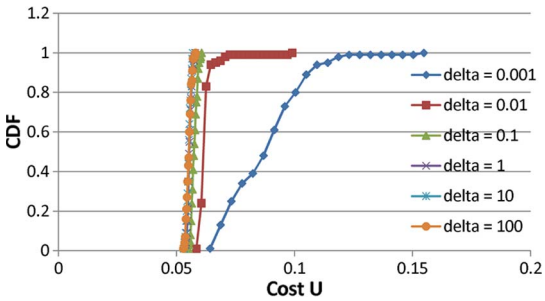



Fig. 6. CDFs of the achieved cost $U$ in 100 simulation runs by the perturbed algorithm under different $\delta$ values. ($\alpha = 1$, $\beta = 0.0001$, $\eta = 0.001$, $\rho = 0$, Topology 3.)

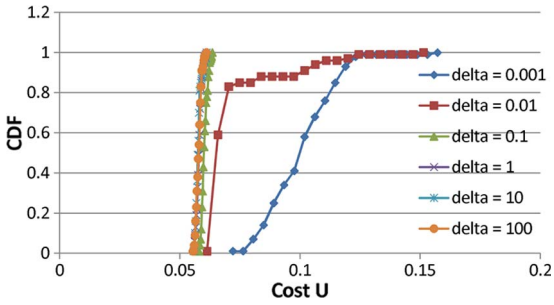



Fig. 7. CDFs of the achieved cost $U$ in 100 simulation runs by the perturbed algorithm under different $\delta$ values. ($\alpha = 1$, $\beta = 0.0001$, $\eta = 0$, $\rho = 0.001$, Topology 3.)

transition matrices found by the steepest descent algorithm. We verify that the cost function that was predicted by steepest descent, indeed, reflects the realized $\Delta C$ and $\bar{E}$ metrics when the computed transition probabilities are applied in practice.

We vary the weights $\alpha$ and $\beta$ in (17) and use steepest descent to compute the optimal transition matrix. A matrix that was generated by each iteration of the steepest descent algorithm

TABLE I
$\bar{C}_i$ FOR $\epsilon = 0.0001$, $\Delta t = 0.000001$, TOPOLOGY 1

| $\alpha : \beta$ | $C_i$ 1 | 2 | 3 | 4 | $\Delta C$ |
|---|---|---|---|---|---|
| 0:1 | 0.214 | 0.286 | 0.286 | 0.214 | 0.138 |
| 1:1 | 0.250 | 0.250 | 0.250 | 0.250 | 0.090 |
| $1 : 10^{-2}$ | 0.254 | 0.246 | 0.246 | 0.254 | 0.085 |
| $1 : 10^{-4}$ | 0.369 | 0.130 | 0.131 | 0.370 | 0.004 |
| $1 : 10^{-6}$ | 0.399 | 0.101 | 0.101 | 0.399 | $4.0 \times 10^{-6}$ |
| 1:0 | 0.4 | 0.1 | 0.1 | 0.4 | 0 |

TABLE II
$\bar{E}_i$ FOR $\epsilon = 0.0001$, $\Delta t = 0.000001$, TOPOLOGY 1

| $\alpha : \beta$ | $E_i$ 1 | 2 | 3 | 4 | $\bar{E}$ |
|---|---|---|---|---|---|
| 0:1 | 9.001 | 9.001 | 9.001 | 9.001 | 324.1 |
| 1:1 | 8.992 | 9.040 | 9.004 | 8.960 | 323.9 |
| $1 : 10^{-2}$ | 8.255 | 9.836 | 9.835 | 8.254 | 329.7 |
| $1 : 10^{-4}$ | 29.770 | 62.513 | 62.099 | 29.524 | 9522.1 |
| $1 : 10^{-6}$ | 48.264 | 118.937 | 118.898 | 48.290 | 32944.1 |
| 1:0 | 6.944 | 13742.767 | 13742.768 | 6.944 | $3.8 \times 10^8$ |

is used to drive a corresponding Markov chain simulation of the mobile sensor schedule, and the $\Delta C$ and $\bar{E}$ values are measured. Each simulation is repeated ten times to obtain the reported average. The 25th and 75th percentiles of the measured values are reported as error bars where they are significant. We study the case in which both $\Delta C$ and $\bar{E}$ are considered ($\alpha = 1, \beta = 0.001$). The results are shown in Fig. 9 for Topology 2. The figure shows that the measured $U$ in the simulations gives a very close match with the computed $U$ by the steepest descent, although the match is not exact because of the simplifying assumption in computing $\bar{E}$ (see Section III-A). See our technical report [24] for further results.

## VIII. Conclusion

As discussed, the main contribution of this paper is the computational framework for the search of the optimal transition probabilities of a Markov chain to minimize some cost function (or, equivalently, the optimization of some utility function) with application to the scheduling of mobile sensor coverage. The key feature of the cost function is that it contains *conflicting* or *antagonistic* criteria such as the coverage and exposure time metrics. The novelty of our method of solution, through *steepest descent*, is its *simplicity* and the *comprehensiveness* of the search space—we look for optimal transition rates in the space of *all* transition probability matrices. Underlying this approach is the explicit expression of the cost (or utility) function in terms of the transition probabilities. The further advantage of

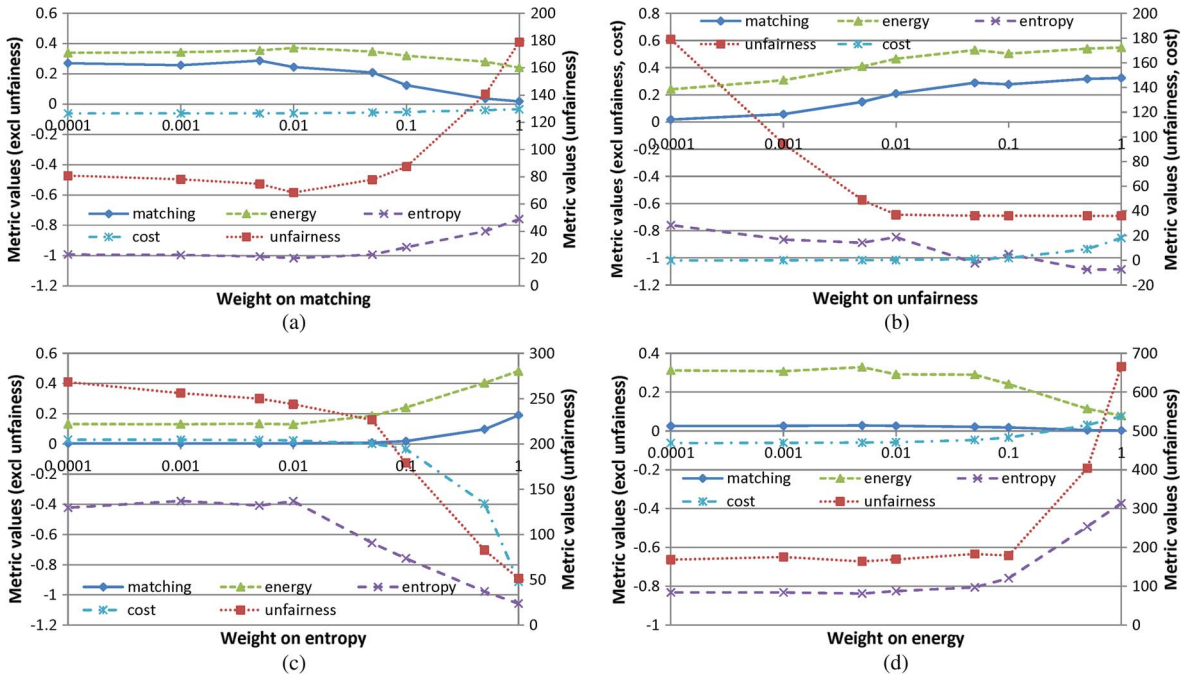



Fig. 8. Changes of different metrics in the cost function as we vary the weight of one metric while keeping the other three metrics constant. ($\alpha = 1$, $\beta = 0.0001$, $\rho = 0.1$, $\eta = 0.1$, $\delta = 10$, and $k = 10\,000$, unless stated otherwise in each subfigure, Topology 1.) (a) Varying $\alpha$. (b) Varying $\beta$. (c) Varying $\eta$. (d) Varying $\rho$.

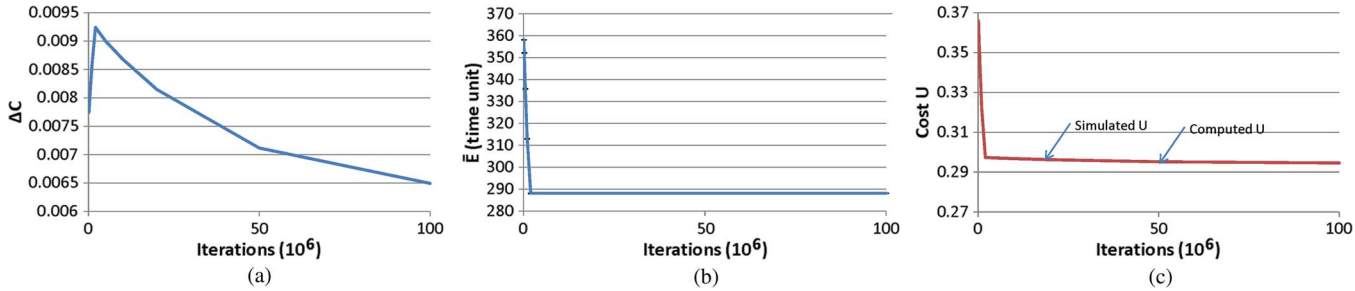



Fig. 9. Performance of the algorithm. (a) $\Delta C$. (b) $\bar{E}$. (c) Overall cost function $U$ as a function of the iteration number. ($\alpha = 1$, $\beta = 0.001$, Topology 2.)

our approach is the ease for individual users to adjust the relative importance of the various criteria suitable for different applications. One challenge of the comprehensive optimization, however, is the complexity of the solution space empirically evidenced by a large number of local optima. We successfully overcome the challenge by incorporating an adaptive time step and noise in the optimization.

## REFERENCES

[1] P. McKenney, "Stochastic fairness queuing," in *Proc. IEEE INFOCOM*, Mar. 1990, pp. 733–740.

[2] A. K. Parekh and R. G. Gallager, "A generalized processor sharing approach to flow control in integrated services networks: The single-node case," *IEEE/ACM Trans. Netw.*, vol. 1, no. 3, pp. 344–357, Jun. 1993.

[3] C. A. Waldspurger and W. E. Weihl, "Lottery scheduling: Flexible proportional-share resource management," in *Proc. 1st Symp. Oper. Syst. Des. Implementation*, 1994, pp. 1–11.

[4] C. A. Waldspurger, "Lottery and stride scheduling: Flexible proportional-share resource management," Ph.D. dissertation, Mass. Inst. Technol., Cambridge, MA, Sep., 1995.

[5] J. C. R. Bennett and H. Zhang, "$WF_2Q$: Worst case fair weighted fair queuing," in *Proc. IEEE INFOCOM*, 1996, pp. 120–128.

[6] N. Bisnik, A. Abouzeid, and V. Isler, "Stochastic event capture using mobile sensors subject to a quality metric," in *Proc. MOBICOM*, Sep. 2006, pp. 98–109.

[7] Z. Butler and D. Rus, "Event-based motion control for mobile sensor networks," *IEEE Pervasive Comput.*, vol. 2, no. 4, pp. 34–42, Oct.–Dec. 2003.

[8] P. Ogren, E. Fiorelli, and N. Leonard, "Cooperative control of mobile sensor networks: Adaptive gradient climbing in a distributed environment," *IEEE Trans. Autom. Control*, vol. 49, no. 8, pp. 1292–1302, Aug. 2004.

[9] R. Murray, R. Janke, W. E. Hart, J. W. Berry, T. Taxon, and J. Uber, "Sensor network design of contamination warning systems: A decision framework," *J. Amer. Water Works Assoc.*, vol. 100, no. 11, pp. 97–109, 2008.

[10] A. Ostfeld, J. G. Uber, and E. Salomons, "Battle of water sensor networks: A design challenge for engineers and algorithms," in *Proc. 8th Symp. Water Distrib. Syst. Anal.*, 2006, pp. 1–3.

[11] M. J. Stealey, A. Singh, M. A. Batalin, B. Jordan, and W. J. Kaiser, "NIMS-AQ: A novel system for autonomous sensing of aquatic environments," in *Proc. IEEE/RSJ Int. Conf. Robot. Autom.*, 2008, pp. 621–628.

[12] W. K. Hastings, "Monte Carlo sampling methods using Markov chains and their applications," *Biometrika*, vol. 57, no. 1, pp. 97–109, Apr. 1970.

[13] N. A. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equation of state calculations by fast computing machines," *J. Chem. Phys.*, vol. 21, no. 6, pp. 1087–1092, Jun. 1953.

[14] R. L. Cruz, "Quality-of-service guarantees in virtual circuit switched networks," *IEEE J. Sel. Areas Commun.*, vol. 13, no. 6, pp. 1048–1056, Aug. 1995.

[15] L. Koralov and Y. G. Sinai, *Theory of Probability and Random Processes*. New York: Springer-Verlag, 2007.

[16] C. D. Meyer, "The role of the group generalized inverse in the theory of finite Markov chains," *SIAM Rev.*, vol. 17, no. 3, pp. 443–464, Jul. 1975.

[17] J. G. Kemeny and J. L. Snell, *Finite Markov Chains*. New York: Van Nostrand, 1960.
[18] P. J. Schweitzer, "Perturbation theory and finite Markov chains," *J. Appl. Prob.*, vol. 5, no. 2, pp. 401–413, Aug. 1968.
[19] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by Simulated Annealing," *Science, New Ser.*, vol. 220, no. 4598, pp. 671–680, May 1983.
[20] S. Geman and C. R. Hwang, "Diffusion for global optimization," *SIAM J. Control Optim.*, vol. 24, pp. 1031–1043, 1986.
[21] B. Hajek, "Cooling schedules for optimal annealing," *Math. Oper. Res.*, vol. 13, no. 2, pp. 311–329, May 1988.
[22] P. Brémaud, *Markov Chains: Gibbs Fields, Monte Carlo Simulations and Queues*. Berlin, Germany: Springer-Verlag, 1999.
[23] R. L. Yang, "Convergence of the simulated annealing algorithm for continuous global optimization," *J. Opt. Theory Appl.*, vol. 104, no. 3, pp. 691–716, Mar. 2000.
[24] C. Y. T. Ma, D. K. Y. Yau, N. K. Yip, N. S. V. Rao, and J. Chen, "Stochastic steepest descent optimization of multiple-objective mobile sensor coverage," Purdue Univ., West Lafayette, IN, Dec. 2011.

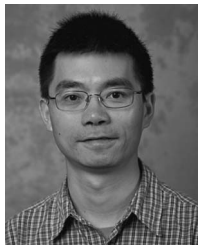
**Chris Y. T. Ma** received the B.Eng. (first-class honors) degree in computer engineering and the M.Phil. degree in computer science and engineering from the Chinese University of Hong Kong, Shatin, Hong Kong, SAR, in 2004 and 2006, respectively, and the Ph.D degree in computer science from Purdue University, West Lafayette, IN, in 2010.

He is currently a Postdoctoral Researcher with the Advanced Digital Sciences Center, Singapore. His research interests include the performance and security study of wireless networks, mobile sensor networks, and smart grids.

**David K. Y. Yau** (M'10) received the B.Sc. (first-class honors) degree in computer science from the Chinese University of Hong Kong, Shatin, Hong Kong, SAR, and the M.S. and Ph.D. degrees in computer science from the University of Texas, Austin.

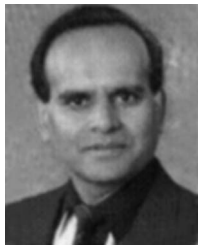
He is currently a Distinguished Scientist with the Advanced Digital Sciences Center, Singapore, and an Associate Professor of computer science with Purdue University, West Lafayette, IN. His research interests include protocol design and implementation, wireless and sensor networks, network security, network incentives, and smart grids.

Dr. Yau was the recipient of a Faculty Early Career Development (CAREER) Award from the National Science Foundation for his research on quality-of-service provisioning. He was an Associate Editor for the IEEE/Association for Computing Machinery TRANSACTIONS ON NETWORKING from 2004 to 2009. He served as the Vice General Chair in 2006, a Technical Program Committee (TPC) Cochair in 2007, and the TPC Area Chair in 2011 of the IEEE International Conference on Network Protocols; a TPC Cochair in 2006 and a Member of the Steering Committee from 2007 to 2009 of the IEEE International Workshop Quality of Service; and a TPC Track Cochair (Network/Web/Peer-to-Peer Protocols and Applications) of the 2012 IEEE International Conference on Distributed Computing Systems.



**Nung Kwan Yip** received the B.Sc. degree in electrical engineering from Purdue University, West Lafayette, IN, in 1991 and the Ph.D. degree in mathematics from Princeton University, Princeton, NJ, in 1996.

He was a Courant Instructor with New York University, New York, and the Van Vleck Assistant Professor with the University of Wisconsin, Madison. Since 1999, he has been with the Department of Mathematics, Purdue University. He has also made long-term visits to the Institute of Pure and Applied Mathematics, Los Angles, CA; the Institute for Mathematics and Its Applications, Minneapolis, MN; and the Max Planck Institute for the Mathematics in the Sciences, Leipzig, Germany. His research interests include partial differential equations, calculus of variations, and probability theory. His recent works include mathematical modeling in materials science and stochastic optimization in computer sensing networks.



**Nageswara S. V. Rao** (F'08) received the B.Tech. degree in electronics and communications engineering from the National Institute of Technology, Warangal, India, in 1982, the M.E. degree in computer science and automation from the Indian Institute of Science, Bangalore, India, in 1984, and the Ph.D. degree in computer science from Louisiana State University, Baton Rouge, in 1988.

From 1988 to 1993, he was an Assistant Professor with the Department of Computer Science, Old Dominion University, Norfolk, VA. In 1993, he joined the Computer Science and Mathematics Division, Oak Ridge National Laboratory, Oak Ridge, TN, where he is currently a Corporate Fellow. Since 2008, he has been on assignment with the U.S. Missile Defense Agency as the Technical Director of the C2BMC Knowledge Center. He has published more than 250 technical conference proceedings and journal papers on sensor networks, information fusion, and high-performance networking.

Dr. Rao received the 2005 IEEE Technical Achievement Award for his contributions to the information fusion area.

**Jiming Chen** (M'08–SM'11) received the B.Sc. and Ph.D. degrees in control science and engineering from Zhejiang University, Hangzhou, China, in 2000 and 2005, respectively.

He was a Visiting Researcher with the National Institute for Research in Computer Science and Control (INRIA), Paris, France, in 2006; the National University of Singapore, Singapore, in 2007; and the University of Waterloo, Waterloo, ON, Canada, from 2008 to 2010. He is currently a Full Professor with the Department of Control Science and Engineering and the Coordinator of the Networked Sensing and Control Group, State Key Laboratory of Industrial Control Technology, Zhejiang University. He is a Guest Editor for *Computer Communication* (Elsevier), *Wireless Communication and Mobile Computer* (Wiley), and the *Journal of Network and Computer Applications* (Elsevier). His research interests include estimation and control over sensor networks, target tracking in sensor networks, energy harvesting for rechargeable sensor networks, and mobile social networks.

Dr. Chen currently serves as an Associate Editor for several international journals, including the IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS. He is a Guest Editor for the IEEE TRANSACTIONS ON AUTOMATIC CONTROL. He also served as a Cochair of the 2011 IEEE GLOBECOM Ad hoc and Sensor Network Symposium, a General Symposia Cochair of the 2009 and 2010 Association for Computing Machinery International Wireless Communications and Mobile Computing Conference, a Publicity Cochair of the Eighth IEEE International Conference on Mobile Ad Hoc and Sensor Systems in 2011, the Seventh IEEE International Conference on Distributed Computing in Sensor Systems in 2011, and the 32nd IEEE International Conference on Distributed Computing Systems in 2012.