Resource Allocation and Scheduling for Energy Efficient Tracking

Praveen Bommannavar
Management Science and Engineering
Stanford University
Stanford, CA 94305 USA
bommanna@stanford.edu

John Apostolopoulos
Hewlett-Packard Labs
Palo Alto, CA 94304 USA
john_apostolopoulos@hp.com

Nicholas Bambos
Electrical Engineering and
Management Science and Engineering
Stanford University
Stanford, CA 94305 USA
bambos@stanford.edu

***Abstract*—We examine the problem of tracking the states of a collection of systems over a finite horizon in a power limited scenario. Specifically, each system has a sensor which can track a property of interest and has a fixed budget with which to make measurements and communicate to a fusion center. The state at each system varies independently according to a Markov model and the transitions between different states occur according known transition matrices. Different systems can have vastly different state evolution statistics. At each time step, the fusion center can request an update from any number of the systems, subject to the constraint that the corresponding sensor has not exhausted its budget to do so. These measurement updates are expensive and hence resource limited. After the fusion center receives all updates, it must estimate the state at each system with minimum error. We give an optimal policy for the fusion center to request updates from each sensor and also provide an optimal policy for the fusion center to allocate the measurement budget to each sensor before deployment, given the transition matrices corresponding to each system.**

***Index Terms*—Dynamic programming, Markov models, Resource allocation, Tracking**

I. Introduction

Due to the widespread availability of inexpensive wireless sensor units with small batteries, the deployment of large collections of sensors to perform tracking operations is becoming increasingly feasible. Applications abound, from surveillance to environmental studies and health care. In particular, we are motivated by the problem tracking geo-spatially varied environmental conditions, such as temperature, moisture, or the presence of pests, using wireless sensors over a time range of interest.

Mathematically, we model this as tracking the states of a collection of systems over a finite horizon in a power limited scenario. In our problem formulation, the states at the different systems vary according to a Markov process and sensors are each system can be 'pinged' by a fusion center to request updates. These updates are constrained by the limitation that they can only be requested a certain number of times over the horizon of interest because of power limitations. We discuss an optimal policy for both scheduling of updates as well as distribution of power among the systems of interest.

Two approaches are usually taken with respect to conserving the power used by the sensors. In our paper we suppose that there is a 'wake-up' mechanism by which the fusion center can ask sensor for updates. If listening for these updates is cheap relative to taking measurements and transmitting a response, this is a good assumption, as made in [1] and [2]. On the other hand, it might be the case that the sensors themselves are smart and know when to wake up to give updates to the fusion center, as in [3] and [4]. Our methodology can be extended to accommodate this setting as well. We discuss such extensions in Section V.

The applications in which sensor networks find practical use and also have unique research implications are many. Decentralized detection for the purpose of intrusion detection has been widely studied and is a domain in which resource consumption must be minimized [5]. The topics that have been tackled include protocols for transmission and coding as well as implementation issues such as sensor density and power usage [6], [7]. While some treatments of these topics are concerned with asymptotic results, we are interested in finding optimal strategies over short intervals of interest. Researchers have also taken interest in determining optimal strategies for information collection when information patterns are limited, that is one does not have access to all of the measurements being made [8].

In this paper we study the tradeoff of power usage versus accurate tracking of system state when there are many independent systems which must be tracked. We suppose that the sensors are utilized over a finite time horizon of interest. Additionally, we look at a two phase problem: in the first (offline) phase, resources are allocated to each system so that updates can be requested from them and in the second (online) phase a fusion center must decide when to request updates so as to maintain the best tracking precision. We take a dynamic programming approach to the problem using a structural approach as introduced in [10].

In Section II we present the mathematical details of our problem formulation and model system states as Markov chains evolving over the time horizon of interest. We also define our objective function and how estimation error is counted. Section III gives the algorithm used by the fusion to request updates from the sensors. In true backwards induction style, Section IV goes on to describe the offline procedure of allocating the power to sensors before deployment. Throughout the algorithmic developments, a small numerical example is given to verify some of the key claims. Finally Section V addresses extensions and future work, and we wrap up with concluding remarks in Section VI.

II. Problem Formulation

We now describe our problem formulation in greater detail. The following model considers the set of states $\mathcal{X}$ in a Markov chain $\mathcal{M}$ to be an abstraction for some attribute of interest in a given system, such as spatial position, health state of a data warehouse or buffer in a queue. Generally speaking, we focus on properties that can be observed and whose evolution may be stochastically modeled as a Markov chain. Additionally we suppose that there is a distance metric which maps any pair of states to a real number.

In our model, we are interested in a scenario in which there are V systems $\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_V$ in which the state evolutions of each system i corresponds to a different transition matrix P_i. Each state evolves independently of the others and each of these systems has a sensor preallocated with a budget M_i with which to make and communicate a measurement to a centralized fusion center over the finite time horizon $k = 1, 2, \ldots, N$. At each time step k of the horizon, a decision must be made by the fusion center on whether or not to request the observed system state for each sensor. After receiving all updates, the center updates its knowledge of the states and makes estimates for those systems that did not send updates.

The objective of the policy we seek to develop is to minimize the error in estimating the state of each system, as measured by the distance metric d, over all time steps. The remainder of this section makes this more precise.

A. System dynamics, metric and communication

Let us focus on one of the V systems in our problem and drop its subscript for convenience. Consider a Markov chain $\mathcal{M}$ with finite state space $\mathcal{X}$, transition matrix P and associated measure $d : \mathcal{X} \times \mathcal{X} \to \mathbf{R}$ as in Fig. 1.

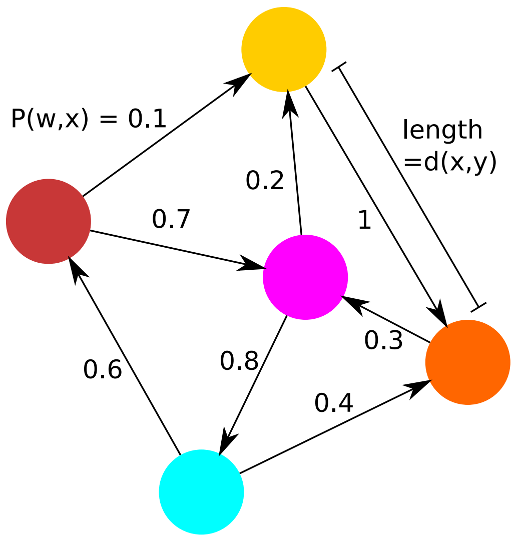

Fig. 1. Markov chain with transitions $P(\cdot,\cdot)$ and measure $d(\cdot,\cdot)$. Self loops are captured by outgoing edge probabilities summing to less than one.

The metric gives a sense of how close states are so that we can measure the effectiveness of an estimate of the true state. We assume that the process is known to start at initial state x_0 and we are interested in having an accurate estimate of the process over a finite horizon $k = 1, ..., N$.

The decision space is simply $u \in \{0, 1\}$ where 0 corresponds to no update being made from the system to the fusion center and 1 corresponds to an update being made. When an update is made, the state x_k of $\mathcal{M}$ is perfectly known by the center. Without an observation, on the other hand, an estimate $\hat{x}_k$ must be formed for the state given all observed information thus far. The number of times updates may be made is limited to $M \leq N$, where M is preallocated the system before deployment.

The cost of making estimate $\hat{x}_k$ at time k when the true state is actually x_k is $d(x_k, \hat{x}_k)$. If d is a metric, we have the important properties

1. $d(x, y) \geq 0 \quad \forall x, y \in S$
2. $d(x, x) = 0 \quad \forall x \in S$
3. $d(x, y) = d(y, x) \quad \forall x, y \in S$
4. $d(x, z) \leq d(x, y) + d(x, z) \quad \forall x, y, z \in S$

At each time k, the state of our decision making problem can be represented by $\{(r, s, t); x_{N-t-r}\}$ where r is the number of time slots that have passed since the last update, s is the number of opportunities remaining to make a transmission based on available power at the system, t is the number of time slots remaining in the problem, and x_{N-t-r} is the last observed state of $\mathcal{M}_i$. This will be formally stated in Section III.

We seek a policy $\pi = \{\mu_k\}_{k=1}^{N-1}$ such that the actions $u_k = \mu_k((r, s, t), x_{N-t-r}) \in \{0, 1\}$ are chosen to minimize the cumulative estimation error at the fusion center. The policy π is admissible if it abides by the additional constraint that the number of times observations are made is no greater than M_i. Denote the class of admissible policies by Π. The goal of the fusion center is to find a policy $\pi^* \in \Pi$ to minimize

$$J_i = \mathbf{E}\left\{\sum_{k=1}^{N-1} d(x_k, \hat{x}_k)\right\}$$

for each $\mathcal{S}_i$. It should be noted that the estimate $\hat{x}_k$ depends on the action u_k because if $u_k = 1$ then $\hat{x}_k = x_k$ and there is no estimation error, while if $u_k = 0$ then the fusion center must make the best guess of the state that is possible with the known information.

Deciding on the distance metric is an issue of modeling and may be specific to the application at hand. We consider a few alternatives here:

1) Probability of Error: To recover a cost structure that results in the same penalty regardless of which state is chosen in error (probability of error criterion), we simply set the distance metric as

$$d(x, y) = \begin{cases} 0 & \text{if } x = y \\ 1 & \text{otherwise} \end{cases}$$

Such a choice maximizes the likelihood of estimating the correct state.

2) Euclidean distance: We may suppose that states correspond to physical locations - in this case, we may choose to let the distance $d(\cdot,\cdot)$ correspond to the Euclidean distance between states so that best estimates minimize the error as measured spatially.

Several other choices could also be made for a distance metric, such as the well known Metropolis distance or Chebyshev distance. In this paper, we are agnostic to the choice of metric, and the techniques outlined here can be utilized in a general setting.

B. Global loss criterion

Knowing the dynamics for the evolution of the processes at each system along with the decision making structure, we can now proceed a discussion of the pre-deployment phase of this problem. When presented with a collection of systems that are suitably described in this way, we must decide how much energy to allocate to each system's sensor. We further suppose that the total amount of energy that can be consumed is limited and make the simplifying assumption that the cost for any of these sensors to make a transmission to the fusion center is fixed. The situation can be pictorially represented as in Fig. 2.

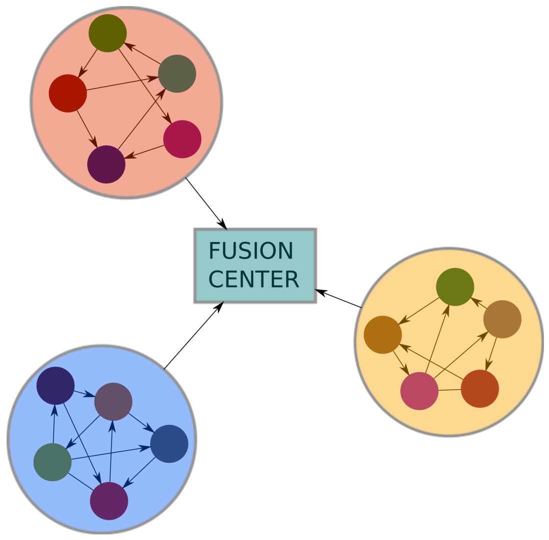

Fig. 2. Independent systems which transmit state information to the fusion center upon request.

Before deploying the sensors, the fusion center must allocate $M_1 \dots M_V$ transmissions to each of them. Given that the estimation errors for each sensor i depends on the number of allocated transmissions, M_i, we denote the resulting error $J_i(M_i)$, or as a vector, $\mathbf{J}(M_1, \dots, M_V)$. We can state our search for an allocation strategy as an optimization problem:

$$\min_{\sum M_i \leq M} \|\mathbf{J}\|_p$$

where $p \in [1, \infty]$. In this paper we present a way to optimally allocate transmissions for this entire range. Let us describe the range of values for p in greater detail:

1) $p = 1$: In the case of $p = 1$ we aim to minimize the sum total of the errors made at each sensor.

2) $p \in (1, \infty)$: In the case of $p \in (1, \infty)$ we apply an exponent, p, to the sum of errors at each sensor and try to minimize this quantity. A common choice is $p = 2$, which corresponds to a sum of squared errors.

3) $p = \infty$: In the case of $p = \infty$ we aim to minimize the maximum error of estimating the state out of all the sensors.

Before considering this optimization however, we focus on the problem of update scheduling between the fusion center and a particular sensor, given that the preallocation step has already been done.

C. Implementation considerations

In certain applications, a number of concepts outlined in the modeling phases here would be applicable with slight modification. For instance, it could be the case that the Markov chains corresponding to state evolution at each system are unknown. One might initially deploy sensors to learn the parameters associated with each system and then use them to determine optimal tracking policies.

Additionally, the idea of being limited on the amount of power usage can map directly to being limited on the number of sensors one is willing to deploy. We can imagine that if all sensors are the same, having more sensors deployed to a system can yield a greater number of opportunities to request updates from the corresponding system.

For simplicity, we proceed without considering the special modifications needed in these settings while noting that the corresponding extensions can be made with little effort.

III. Update Scheduling

Let us now zoom into the problem of optimally scheduling the allotted M_i updates at sensor i with the receiver. We proceed with a proposition:

Proposition 1. *The policies for requesting updates from each sensor are independent; that is, the policy for requesting and update from $\mathcal{S}_i$ can be formulated independently of the policy for another system $\mathcal{S}_j$.*

This follows immediately from the facts that the preallocated resources are not shared and that the Markov chains $\mathcal{M}_i$ and $\mathcal{M}_j$ are independent of each other. Knowing this, we can design the policy π_i to be agnostic to the presence of other systems.

A. Dynamic Programming

We use a dynamic programming approach to obtain an optimal policy corresponding to each sensor [11]. Before presenting our algorithm for determining π^*, however, we first develop some important notation. In order to proceed, we must begin by determining several quantities offline. Let $\mathbf{d}(w)$ be the vector of distances of each state from w. Then we proceed by cataloging the quantities

$$\begin{aligned} w_r^*(x) &= arg\min_{w\in S}\left\{\sum_{y\in S}\mathbf{P}[x_r = y|x_0 = x]d(y,w)\right\} \\ &= arg\min_{w\in S}\{(P^r\mathbf{d}(w))(x)\} \\ e_r^*(x) &= (P^r\mathbf{d}(w_r^*(x)))(x) \end{aligned}$$

for $r = 1, ..., N$. The values $w_r^*(x)$ and $e_r^*(x)$ correspond to the optimal estimate and estimation error, respectively, when we must determine the current state given that r time steps ago we observed that the state was x. There may, in some cases, be an efficient way to determine these quantities, but in general we must do this by simply cataloging these quantities offline through brute force. This may be done with relative ease if the state space is of tractable size or if the system's transition matrix displays certain sparsity (if the state is changing at a bounded rate then we may narrow it down to a sparse set of states).

Now we proceed to construct the solution using backwards induction. We begin with $t = 1$, which corresponds to one unit of time remaining in the problem, and then continue for $t = 2, 3, ...$ until we are able to determine a recursion. As we build backwards in time (and forward in t), we let s vary and keep track of the cost $J_{r,s,t}(x)$ where x is a state of the Markov chain. This is depicted graphically in Fig. 3, where the index r has been omitted. A given state (s, t) can transition to $(s-1, t-1)$ or $(s, t-1)$. The transition represents whether an observation was made or not - if so, then s is decremented, otherwise it remains the same. In the special case of $s = t$, the only sensible policy is to always use an observation, and in the case of $s = 0$, the only admissible policy is not to make an observation. This is also shown in Fig. 3.

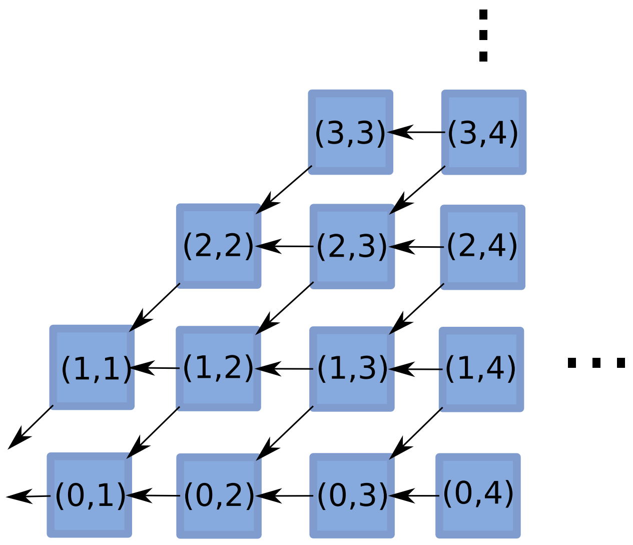

Fig. 3. Admissible transitions for backwards induction. The pair (s,t) represents the number of observations and time steps, respectively.

For $t = 1$, we can either have $s = 0$ or $s = 1$. These costs, respectively, are (in vector form)

$$\begin{aligned} J_{(r,0,1)} &= e_r^* \\ J_{(r,1,1)} &= 0 \end{aligned}$$

since not having an observation means we need to make a best estimate, and having an observation leads to zero cost.

Moving on to $t = 2$, the values of s can range from $s = 0$, $s = 1$ or $s = 2$. For $s = 0$ we have

$$J_{(r,0,2)} = e_r^* + e_{r+1}^*$$

since we would need to make an optimal estimate with no further information for the next two time slots. If $s = 1$, there are two choices: use an opportunity to make an observation so that $u = 1$, or do not observe, in which case $u = 0$. These choices can be denoted with superscripts above the cost function for each stage:

$$\begin{aligned} J_{(r,1,2)}^{(0)}(x) &= e_r^*(x) + J_{(r+1,1,1)}(x) = e_r^*(x) \\ J_{(r,1,2)}^{(1)}(x) &= 0 + \sum_{y\in S} P[x_{N-2} = y|x_{N-2-r} = x]e_1^*(y) \end{aligned}$$

For $u = 0$, we accrue error for the current time slot and no error afterwards. When an observation is made, no error is accrued for the current time slot $N - 2$, but there is error in the next time slot, which depends on the current observation. In vector form, we may write

$$\begin{aligned} J_{(r,1,2)}^{(0)} &= e_r^* + J_{(r+1,1,1)} = e_r^* \\ J_{(r,1,2)}^{(1)} &= P^r e_1^* \end{aligned}$$

We now introduce some new notation:

$$\begin{aligned} \Delta_{(r,1,2)} &= J_{(r,1,2)}^{(0)} - J_{(r,1,2)}^{(1)} \\ &= e_r^* - P^r e_1^* \end{aligned}$$

so that if $\Delta_{(r,1,2)}(x) \leq 0$, then we should not make an observation, whereas we should make an observation if $\Delta_{(r,1,2)}(x) > 0$. We proceed now by defining sets $\tau_{(r,1,2)}$ and $\tau_{(r,1,2)}^c$ such that

$$\begin{aligned} x \in \tau_{(r,1,2)}^c &\Leftrightarrow \Delta_{(r,1,2)}(x) \leq 0 \\ x \in \tau_{(r,1,2)} &\Leftrightarrow \Delta_{(r,1,2)}(x) > 0 \end{aligned}$$

and we also define an associated vector $\mathbf{1}_{(r,1,2)} \in \{0, 1\}^S$ which specifies the desired policy, as described in Section III.B:

$$\mathbf{1}_{(r,1,2)}(x) = \begin{cases} 1 & \text{if } x \in \tau_{(r,1,2)}^c \\ 0 & \text{otherwise} \end{cases}$$

Moving on to $s = 2$, we have $J_{(r,2,2)} = 0$, since there are as many opportunities to observe the process as there are remaining time slots. Due to space limitations, we suffice it to say that this pattern can be continued and a recursive solution emerges.

B. Solution

We construct an optimal policy by storing for each (r, s, t) a subset of $\mathcal{X}$, denoted by $\tau^c_{(r,s,t)}$, which is the set of last observed states for which we do not use an opportunity to view the process when we are at stage (r, s, t). That is, if the last observed state x was seen r time slots ago, it is in the set $\tau^c_{(r,s,t)}$, there are s opportunities remaining to make observations and there are t time slots remaining in the horizon then we should not make an observation at this time and simply make an estimate $w^*_r(x)$. On the other hand, if $x \in \tau_{(r,s,t)}$ then we should make an observation at stage (r, s, t) and accrue zero cost for that stage.

More precisely, an optimal policy π^* is given by

$$u_{(r,s,t)}(x) = \begin{cases} 0 & \text{if } x \in \tau^c_{(r,s,t)} \\ 1 & \text{otherwise} \end{cases}$$

Let us introduce three vector valued functions: $F_{(r,s,t)}, \Delta_{(r,s,t)} \in \mathbf{R}^S$ and $\mathbf{1}_{(r,s,t)} \in \{0,1\}^S$. We fill in values for these functions by using the following recursions:

$$\begin{aligned} F_{(r,s,t)} &= F_{(r+1,s-1,t-1)} \\ &\qquad + P^r diag(\mathbf{1}_{(1,s-1,t-1)})\Delta_{(1,s-1,t-1)} \\ \Delta_{(r,s,t)} &= e^*_r + F_{(r+1,s,t-1)} - F_{(r,s,t)} \\ &\qquad + diag(\mathbf{1}_{(r+1,s,t-1)})\Delta_{(r+1,s,t-1)} \\ \mathbf{1}_{(r,s,t)}(x) &= \begin{cases} 0 & \text{if } \Delta_{(r,s,t)}(x) > 0 \\ 1 & \text{otherwise} \end{cases} \end{aligned}$$

for $1 < s < t < N$ and $1 \leq r \leq N - t + 1$. We also have the boundary conditions

$$F_{(r,t,t)} = 0, \quad F_{(r,1,t)} = P^r \sum_{j=1}^{t-1} e^*_j, \quad \Delta_{(r,t,t)} = e^*_r$$

These recursions allow us to determine the sets $\tau^c_{(r,s,t)}$ for s, t, r in the bounds specified, which in turn defines our optimal policy. Specifically, we assign

$$x \in \tau^c_{(r,s,t)} \Leftrightarrow \Delta_{(r,s,t)}(x) \leq 0$$

We conclude by giving expressions for the cost-to-go from any particular state when a particular action $u \in \{0, 1\}$ is taken. The superscripts denote whether or not an observation will be made in the current stage.

$$\begin{aligned} J^{(0)}_{(r,s,t)} &= e^*_r + F_{(r+1,s,t-1)} + diag(\mathbf{1}_{(r+1,s,t-1)})\Delta_{(r+1,s,t-1)} \\ J^{(1)}_{(r,s,t)} &= F_{(r,s,t)} \end{aligned}$$

Observe that $\Delta_{(r,s,t)}$ is the difference between these two quantities. Hence, $\Delta_{(r,s,t)}$ functions as a method of determining whether or not to make an observation in the current time step. Fig. 4 shows a numerical illustration of the effect of having more observation update opportunities on the number of states in the state space inducing an update. The example features a state space with $|\mathcal{X}| = 9$ and $N = 10$, where the plot corresponds to the first time step of the horizon.

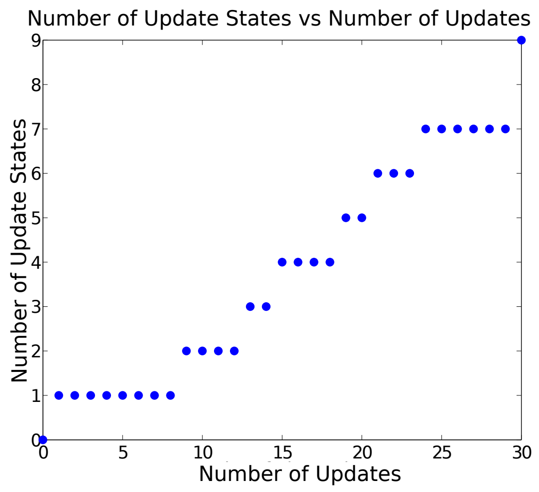

Fig. 4. Number of states resulting in an update as a function of update opportunities. Note that as updates become available, more states result in an update request at time $t = 10$.

C. Convexity/Monotonicity

The cost-to-go function J has several interesting properties, but the most important for our analysis will be *convexity* and *monotonicity*:

Proposition 2. *$J_{(r,s,t)}(x)$ is convex and monotonically decreasing in s for fixed r, t, x.*

Due to space limitations, we give intuition behind this proposition but leave the details to [12]. As more opportunities are given to update the system, the value of each additional update decreases. Put another way, as fewer update opportunities are available for a particular sensor, the greater the marginal loss in performance when another update is taken away. Numerical examples of such a curve are given in Fig. 4.

IV. Resource Allocation

The previous section detailed an optimal update request policy from the fusion center to a particular sensor *given an allocated budget of update opportunities*. We now take a macro view of the system and discuss the allocation of update actions. In a real world setting, this may correspond to having a number of systems for which we must decide the number of sensors to be assigned. Our analysis and intuition can be utilized with minimal modification in this modified scenario.

Returning to our problem formulation, we remind ourselves that our goal is to choose integers M_i according to the optimization problem

$$\min_{\sum M_i \leq M} \|\mathbf{J}\|_p \tag{1}$$

with $p \in [1, \infty]$.

Proposition 3. *A solution to the preallocation problem is given by a greedy policy.*

Let us consider three cases - in all three, we shall find it instructive to refer to the numerical data in Fig. 5, which has a simulated cost-to-go curve for each of two sensors.

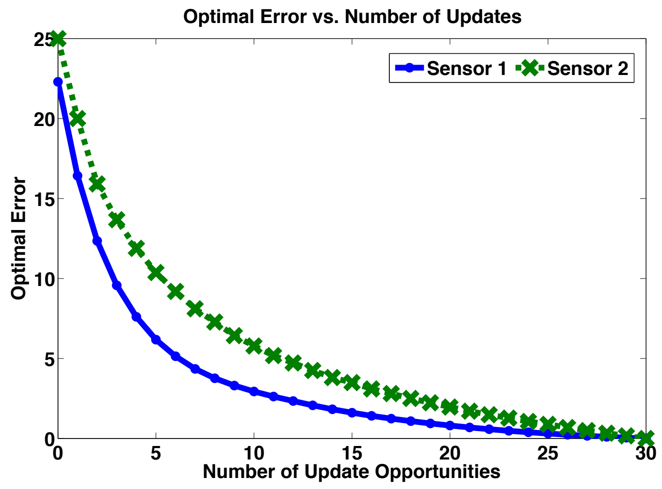

Fig. 5. Cost-to-go for each of two sensors with fixed $r = 1, t = 30$. Both curves are monotonically decreasing and convex.

1) $p = 1$: In the case of $p = 1$ we propose a greedy policy in which each additional update opportunity is given to the sensor which reduces cost the most. For the curves in Fig. 4, then, sensor 1 should be given the first update allocation. Then, sensor 2 would be given the next update allocation, because moving from $M_2 = 0$ to $M_2 = 1$ produces a larger decrease in overall cost than $M_1 = 1$ to $M_1 = 2$. Continuing in this way until the limit of M is reached results in an optimal preallocation.

2) $p \in (1, \infty)$: For $p \in (1, \infty)$ we can simple take the cost-to-go curve of Fig. 4 and apply the exponent p to each point, and continue with the greedy policy detailed in the $p = 1$ case.

3) $p = \infty$: When $p = \infty$, we can continue using a greedy approach, however instead of choosing the sensor whose first measurement update opportunity most reduces error, we must choose the sensor with the maximum error. The first measurement opportunity in our ongoing example would be given to sensor 2 because when $M_1, M_2 = 0$, sensor 2 has the highest cost-to-go. Then, the second opportunity should be given to sensor 1, since it has the highest cost-to-go when $M_1 = 0$ and $M_2 = 1$.

The greedy policy is shown in Table I for $p = 1$ and $p = \infty$. At each step of the allocation process, one must decide which of the two sensors from Fig. 5 is to be allocated the next available update opportunity.

	1	2	3	4	5	6	7	8	9	10
$p = 1$	S_1	S_2	S_1	S_1	S_2	S_2	S_2	S_2	S_2	S_2
$p = \infty$	S_2	S_1	S_2	S_1	S_2	S_2	S_1	S_2	S_2	S_1

TABLE I
ALLOCATION OF GREEDY POLICY FOR $p = 1$ AND $p = \infty$.

One can observe that when $p = 1$ the assignment of resources happens in more of a block-like fashion, whereas $p = \infty$ induces a more 'back and forth' assignment strategy. By inspecting the curves and considering the nature of these two norms, it is unsurprising that the assignment of resources takes such a structure.

V. Concluding Remarks and Future Directions

In this paper, we have introduced the problem of allocating measurement budgets to a set of sensors and constructing a optimal policy for the fusion center to request these measurement updates. We have seen that an optimal policy can be constructed for each sensor independently of the other sensors and that a dynamic programming approach can be used to this end. Once a cost-to-go curve is found for each sensor, a greedy approach can be used to assign a limited collection of measurements to the sensors. In practical settings, this can correspond to deciding on how *many* identical sensors to deploy to a particular system.

Future work in this rich area can consider different (partial) information patterns, smart sensors and correlated system states. Another modification can be made by supposing that the sensors are 'smart' in that *they* decide when to transmit updates to the fusion center. We can finally suppose that different systems' states are correlated, changing the scheduling policy for update requests as well as the preallocation policy of measurement updates.

References

[1] L. Yang et al., A multi-modality framework for energy efcient tracking in large scale wireless sensor networks, in *Proc. 2006 IEEE Int. Conf. Networking, Sensing Control*, Apr. 2006, pp. 916 - 921.
[2] R. Gupta and S. R. Das, Tracking moving targets in a smart sensor network, in *Proc. 58th IEEE Vehicular Technology Conf., Oct. 2003*, vol. 5, pp. 3035 - 3039.
[3] N. Vasanthi and S. Annadurai, Energy saving schedule for target tracking sensor networks to maximize the network lifetime, in *Proc. 1st Int. Conf. Communication System Software Middleware*, Jan. 2006, pp. 1 - 8.
[4] C. Gui and P. Mohapatra, Virtual patrol: A new power conservation design for surveillance using sensor networks, in *Proc. IEEE/ACM Information Processing in Sensor Networks (IPSN)*, 2005, pp. 246253.
[5] S. Appadwedula, V.V. Veeravalli and D.L. Jones. "Decentralized Detection with Censoring Sensors," in *IEEE Transactions on Signal Processing*, 56(4): 1362-1373, April 2008.
[6] J.-F. Chamberland and V.V. Veeravalli. "How Dense Should a Sensor Network be for Decentralized with Detection Correlated Observations?" in *IEEE Transactions on Information Theory*, 52(11):5099-5106, November 2006.
[7] J. Fuemmeler and V.V. Veeravalli, "Smart Sleeping Policies for Energy Efficient Tracking in Sensor Networks," in *IEEE Transactions on Signal Processing*, vol. 56 no. 5: pp. 2091-2102, May 2008.
[8] S. Appadwedula, V.V. Veeravalli and D.L. Jones. "Decentralized Detection with Censoring Sensors," in *IEEE Transactions on Signal Processing*, 56(4): 1362-1373, April 2008.
[9] S. Appadwedula, V.V. Veeravalli, and D.L. Jones, "Energy Efficient Detection in Sensor Networks," in *IEEE Journal on Selected Areas in Communications*, vol. 23 no. 4, pp. 693-702, April 2005.
[10] O.C. Imer. Optimal Estimation and Control under Communication Network Constraints. Ph.D. Dissertation, UIUC, 2005.
[11] D. P. Bertsekas, *Dynamic Programming and Optimal Control*. Belmont, MA: Athena Scientic, 1995.
[12] P. Bommannavar and N. Bambos, Power Limited Tracking in Sensor Networks. Technical Report, Stanford University, 2012.