Performance analysis of different routing protocols in WSN for real-time estimation

Damiano Varagnolo, Phoebus Chen, Luca Schenato, Shankar S. Sastry

***Abstract*— In this paper we analyze the performance of two different routing protocols specifically designed for Wireless Sensor Networks (WSNs) for real-time estimation, control an monitoring. These protocols are designed to compensate for the lossy nature of the wireless links and the delay from sending messages over multiple hops from the sensors to the controller. The routing protocols are designed to reduce packet delay and packet loss using either retransmissions or multicasting, and for some routing topologies one protocol may be better than the other at reducing the worst case packet delay but may have a worse packet loss rate. Here, we apply mathematical tools to analytically compute the average real-time performance based on end-to-end packet delay statistics for two recently proposed routing strategies and we show that the performance is strongly related to the dynamics of the systems being estimated. This suggests that routing protocols are to be designed based on the specific real-time estimation and control application under consideration.**

***Index Terms*— packet drop, random delay, remote estimation, wireless sensor networks, routing, multipath**

I. Introduction

Wireless Sensor Networks (WSNs), i.e. large networks of smart devices that can sense and control the environment and can exchange information with their neighbors via wireless communication, are being employed for a number of very diverse applications. In particular, WSNs have started to be employed for real-time estimation, control and monitoring applications since they can be deployed and installed more rapidly and cheaply than standard wired networks. However, WSNs suffer the same problems in wireless communications such as time-varying channels and large packet loss probabilities. Moreover, this is exacerbated by the need to multi-hop messages through intermediate nodes to communicate with far away nodes or the base station. As a consequence, multi-hopping potentially increases the end-to-end packet loss rate and induces varying delay due to retransmission, multiple path routing, and out-of-order packet arrival. These problems pose two main challenges: the first challenge is how to design routing protocols which give rise to low end-to-end packet loss and small delay (latency), and the second challenge is how to design real-time estimation algorithms which can cope with random delay and packet loss. In the following, we briefly review the most relevant literature in these two areas and our contribution.

This work was supported in part by the Italian CaRiPaRo Foundation, by EU PF7 project "FeedNetBack", and by TRUST (Team for Research in Ubiquitous Secure Technology), which receives support from the National Science Foundation (NSF award number CCF-0424422) and the following organizations: AFOSR (#FA9550-06-1-0244), BT, Cisco, ESCHER, HP, IBM, iCAST, Intel, Microsoft, ORNL, Pirelli, Qualcomm, Sun, Symantec, Telecom Italia, and United Technologies.

L. Schenato and D. Varagnolo are with the Department of Information Engineering, University of Padova, Via Gradenigo, 6/b, Padova, Italy {schenato|damiano.varagnolo}@dei.unipd.it

P. Chen and S. Sastry are with the Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, USA. {phoebusc|sastry}@eecs.berkeley.edu

A. *WSN Routing for Estimation and Control*

WSN routing protocols developed for estimation and control need to provide good reliability with low latency. As a result, many protocols like to use a TDMA scheduling scheme, as opposed to CSMA schemes with randomized contention schemes to access the network. For instance, both Time Synchronized Mesh Protocol (TSMP) [1] and RT-Link [2] schedule link transmissions across the network to bound end-to-end latency of a packet. However, RT-Link is less robust to link failure than TSMP because RT-Link is a single-path routing protocol while TSMP is a mesh routing protocol.

Other WSN protocols designed for industrial control, such as SERAN [3], Breath [4], and DSF [5] use cluster-based routing to get higher reliability. Cluster-based routing is a form of constrained flooding, where copies of a packet are passed between groups of nodes. SERAN and Breath assume the independence of links, node wake up times, and random attempts to access the channel so that the Central Limit Theorem can be employed to get probabilistic guarantee on end-to-end delivery and reliability. DSF assumes independence of links and uses individual link probabilities to get end-to-end connectivity as a function of latency.

B. *Estimation and control subject to random delay and packet loss*

Recently, there has been a considerable effort in analyzing and designing estimation and control schemes in networked control systems (NCS) subject to packet loss and packet delay, as surveyed in [6] and [7] and the references therein. Most of the results are concerned with finding stability conditions for filtering and control, and in general very few results provide a quantitative measure of performance based on packet delay and loss statistics. Among these, in [8] the authors provided upper and lower bounds for the optimal mean square estimator in systems subject to packet loss but not to packet delay. On the other hand, Nilsson et al. in [9] considered designing an optimal LQG regulator when packets are subject to random packet delay with known statistics, but not to packet loss. Recently, in [10] we proposed different estimation algorithms with quantifiable performance if the packet delay statistics are known and i.i.d.

Another important related area of research addresses the problem of finding numerically efficient algorithms to compute the optimal mean square estimator subject to delayed measurements, as in [11] and [12]. These are general algorithms which require little memory and are also valid for time-varying dynamics and out-of-order packet arrival. However, they do not provide performance evaluation tools based on packet delay statistics, which is of primary concern in our work.

C. Contribution

In this paper, we will study the performance of real-time filtering running over two of the most promising routing protocols: Directed Staged Flooding (DSF) and Unicast Path Diversity (UPD), which is a specific implementation of a protocol based on TSMP [5]. In particular, we show how to derive the end-to-end packet loss latency and connectivity statistics in terms of λ_h, the probability that a packet sent from the sensor is delivered to the estimator with a delay τ less than h (i.e. $\lambda_h = \mathbb{P}[\tau \leq h]$). These statistics are used offline to compute the performance, in the sense of estimation error covariance, of a class estimators which use a Kalman-like filter with a buffer of dimension N that stores the measurements that arrive with different delays. These types of filters have been proposed in [10], and here we extend them to consider a shifted buffer, i.e. only measurements with a delay between between M and $M+N$, where M is the buffer shift. Through some numerical examples, we show that there is a trade off between performance, computational complexity and system dynamics, which might lead to regimes where one routing protocol is better than the other and vice versa. This implies that protocols must be chosen with the specific application in mind.

II. Networking protocols for WSN: UPD and DSF

This section provides brief descriptions and Markov Chain models of two mesh routing protocols designed to provide high reliability for industrial control applications. For more details and examples, see [5].

A. Unicast Path Diversity

Dust Networks, Inc. proposed Unicast Path Diversity (UPD) over Time Synchronized Mesh Protocol (TSMP) [1], which exploits frequency, time, and space diversity for reliable networking in sensor networks. UPD is a many-to-one, multi-path routing protocol where each node in the network has multiple parents and the routing graph has no cycles. The links selected for routing are bidirectional, hence every link transmission can be acknowledged. If a packet transmission is not acknowledged, it is queued in the node for retransmission. To schedule the network, time is divided into time slots, and grouped into superframes. At each time slot, pairs of nodes are scheduled for transmitting a packet on different frequencies. The superframe containing the schedule of transmissions is repeated over time. Our model uses frequency hopping to justify the assumption that links are independent over retransmissions.

To model UPD to calculate λ_h, we construct a general Mesh TDMA Markov Chain (MTMC) model for UPD that assumes knowledge of the routing topology, schedule, and all the link probabilities. MTMC models single packet transmission in the network without the effects of queuing.

1) Mesh TDMA Markov Chain Model: Let us represent the routing topology as a graph $G = (\mathcal{V}, \mathcal{E})$, and denote a node in the network as $i \in \mathcal{V} = 1, \ldots, N$, and a link in the network as $l \in \mathcal{E} \subset \{(i,j) \mid i,j \in \mathcal{V}\}$, where $l = (i,j)$ is a link for transmitting packets from node i to node j. Time h will be measured in units of time slots, and let H denote the number of time slots in a superframe. The link success probability for link $l = (i,j)$ at time slot h is denoted $p_l^{(h)}$, or $p_{ij}^{(h)}$. We set $p_l^{(h)} = 0$ when link l is not scheduled to transmit at time h.

For a packet originating from a source node a routed to a sink node b, we wish to compute λ_h, the probability the packet reaches b at or before time h has elapsed. This is done by a time-varying, discrete-time Markov chain.

Definition 1 (MTMC Model): Let the set of states in the Markov chain be the nodes in the network, $\mathcal{V}$. The transition probability from state i to state j at time h is simply $p_{ij}^{(h)}$, with $p_{ii}^{(h)} = 1 - \sum_{j \neq i} p_{ij}^{(h)}$. Let $P^{(h)} = [p_{ij}^{(h)}]^T \in [0,1]^{N \times N}$ be the column stochastic transition probability matrix for a time slot and $P^{(\underline{H})} = P^{(H)} P^{(H-1)} \ldots P^{(1)}$ be the transition probability matrix for a repeating superframe. Assume

$$P^{(H+d)} = P^{(cH+d)}, \qquad \forall c, d \in \mathbb{Z}_+ \tag{1}$$

meaning that the link probabilities in a time slot do not vary over superframes.

A packet originating at node a is represented by $\mathbf{p}^{(0)} = \mathbf{e}^{[a]}$, where $\mathbf{e}^{[a]}$ is an elementary vector with the a-th element equal to 1 and all other elements equal to 0. Then,

$$\begin{aligned} \lambda^h = P^{(h)} \cdots P^{(2H+1)} \underbrace{P^{(2H)} P^{(2H-1)} \cdots P^{(H+1)}}_{P^{(\underline{H})}} \cdot \\ \underbrace{P^{(H)} P^{(H-1)} \cdots P^{(1)}}_{P^{(\underline{H})}} \mathbf{p}^{(0)} \end{aligned} \tag{2}$$

represents the probability distribution of the packet over the nodes at time h. ■

B. Directed Staged Flooding

Directed Staged Flooding (DSF) uses simple constrained flooding for one-to-many or one-to-one routing. Unlike UPD, DSF provides increased end-to-end connectivity with less latency by multicasting packets instead of using acknowledgments and retransmissions.

Like UPD, DSF also assumes that the nodes follow a TDMA routing schedule. During a transmission each node transmits to a subset of its neighboring nodes. Furthermore, we group the nodes along the end-to-end transmission path such that a packet is modeled as being passed between groups of nodes, and we call each group of nodes a *stage*. For instance, looking at the node topology for one time slot on the right of Figure 1, each column of nodes is a stage. For simplicity, in this paper we assume nodes are not shared between stages, although this is not required (see [5]).

We use a Directed Staged Flooding Markov Chain (DSFMC) model to find λ_h. As with UPD, we build the

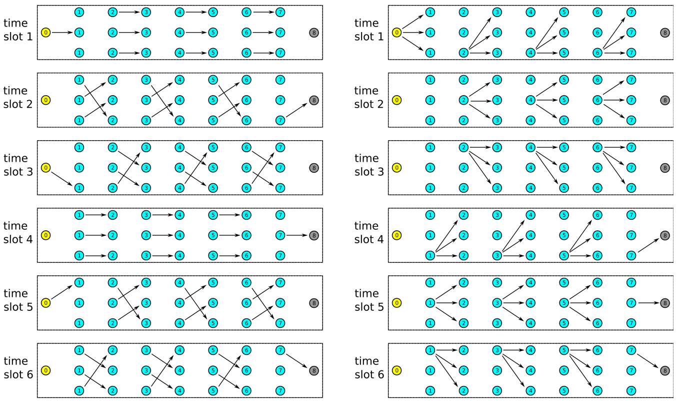

Fig. 1. (left) UPD and (right) DSF schedules for routing on a grid of width 3, used in the calculations for the graph in Figure 2.

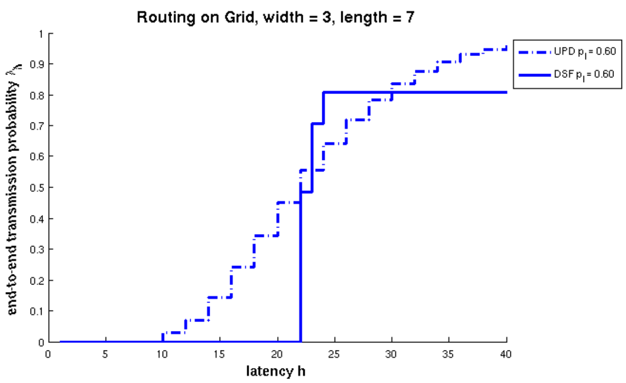

Fig. 2. *Calculated* end-to-end connectivity as a function of latency using the routing schedules described in Figure 1, where all the links have probability $p_l = 0.6$.

model assuming we are provided with a routing schedule, the way nodes are grouped into stages, and all the link probabilities. Our DSFMC model of DSF requires the *sets* of link transmissions between *distinct* pairs of stages to be independent. Like UPD, DSF uses frequency hopping over time to help justify this assumption. However, the model allows the link transmissions between the *same* pair of stages to be correlated. Our DSFMC model also assumes that all nodes in one stage transmit their copy of the packet before the nodes in the next stage transmit their copy of the packet.

1) Directed Staged Flooding Markov Chain Model: As before, we represent the routing topology as a graph $G = (\mathcal{V}, \mathcal{E})$ and denote a node in the network as $i \in \mathcal{V} = 1, \ldots, N$ and a link in the network as $l \in \mathcal{E} \subset \{(i,j) \mid i,j \in \mathcal{V}\}$, where $l = (i,j)$ is a link for transmitting packets from node i to node j. Because each link is used only once when transmitting a single packet, the link success probability for link $l = (i,j)$ is treated as being time-invariant and is denoted p_l, or p_{ij}.

Unlike the MTMC model, in the DSFMC model a state in the Markov chain at a stage represents the *set* of nodes in the stage that successfully received a copy of the packet. The transition probabilities between the states depend on the joint probability of successful link transmissions between stages. Below is the DSFMC model for the special case where the links are all independent.

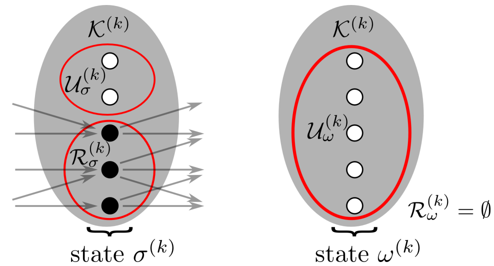

Fig. 3. Mapping of states to nodes that received a packet in the DSFMC model. On the left is an example of a state $\sigma^{(k)}$ and on the right is the state $\omega^{(k)}$ where no packets have been received.

Definition 2 (DSFMC Model): Let's assume we have a routing topology with $K+1$ stages $0, \ldots, K$. Each stage k has N_k nodes, and the set of 2^{N_k} possible states in stage k is represented by the set of numbers $\mathcal{S}^{(k)} = \{0, \ldots, 2^{N_k} - 1\}$. Let $\mathcal{K}^{(k)}$ be the set of nodes in stage k and for each state $\sigma^{(k)} \in \mathcal{S}^{(k)}$, let $\mathcal{R}_\sigma^{(k)} \subset \mathcal{K}^{(k)}$ be the set of nodes that have received a copy of the packet and $\mathcal{U}_\sigma^{(k)} = \mathcal{K}^{(k)} \backslash \mathcal{R}_\sigma^{(k)}$ be the set of nodes that have not received a copy of the packet (See Figure 3). Let $\omega^{(k)}$ denote the state where no nodes received a copy of the packet in stage k.

The conditional probability of the next state $\mathbf{X}^{(k+1)}$ being state $\sigma^{(k+1)}$ given that the current state $\mathbf{X}^{(k)}$ is $\sigma^{(k)}$ can be expressed as

$$\mathbb{P}(\mathbf{X}^{(k+1)} = \sigma^{(k+1)} | \mathbf{X}^{(k)} = \omega^{(k)}) = \begin{cases} 1 & : \quad \sigma^{(k+1)} = \omega^{(k+1)} \\ 0 & : \quad \text{otherwise} \end{cases}$$

if $\sigma^{(k)} \neq \omega^{(k)}$

$$\mathbb{P}(\mathbf{X}^{(k+1)} = \sigma^{(k+1)} | \mathbf{X}^{(k)} = \sigma^{(k)}) =$$

$$\left(\prod_{\substack{u \in \mathcal{U}_\sigma^{(k+1)} \\ i \in \mathcal{R}_\sigma^{(k)}}} (1 - p_{iu}) \right) \prod_{r \in \mathcal{R}_\sigma^{(k+1)}} \left(1 - \prod_{i \in \mathcal{R}_\sigma^{(k)}} (1 - p_{ir}) \right) \tag{3}$$

The transition probability matrices between stage k and $k+1$ are $P^{(k+1)} \in [0,1]^{N_{k+1} \times N_k}$, where the entry in position $(\sigma^{(k+1)}, \sigma^{(k)})$ of the matrix is $\mathbb{P}(\mathbf{X}^{(k+1)} = \sigma^{(k+1)} | \mathbf{X}^{(k)} = \sigma^{(k)})$.

The initial state $\mathbf{X}^{(0)}$ is the state $\sigma^{(0)}$ corresponding to $\mathcal{R}_\sigma^{(0)} = \{a\}$, where a is the node sending the initial packet. Then, the probability distribution $\mathbf{p}^{(k)} \in [0,1]^{N_k}$ of the state at stage k is

$$\mathbf{p}^{(k)} = \underbrace{P^{(k)} \ldots P^{(2)} P^{(1)}}_{P^{(\underline{k})}} \mathbf{p}^{(0)} \tag{4}$$

■

Let us assume the transmissions of nodes within a stage must be scheduled in separate time slots so they do not interfere with each other. We can obtain the probability that a copy of the packet is at a node i at time h directly from our model by translating h to k from the relation $h = \sum_{i=0}^{k-1} N_i$ and looking at $\sum_{\{\sigma^{(k)} | i \in \mathcal{R}_\sigma^{(k)}\}} \mathbb{P}(\mathbf{p}^{(k)} = \sigma^{(k)})$.

C. UPD and DSF Comparison

UPD has the potential to deliver packets from the source to the sink in a shorter period of time than DSF, but the packet delivery time has a larger variance. Also, because $\lim_{h\to\infty}\lambda_h = 1$ for UPD and λ_h for DSF is a fixed value less than 1 after the last stage transmits (assuming $p_l \neq 1$), UPD can always provide better end-to-end connectivity at high latencies h. DSF tends to perform well when there are very poor links in the network. Figure 2 compares λ_h for UPD and DSF under the schedules in Figure 1, where all links have probability $p_l = 0.6$.

III. Minimum variance estimators subject to packet loss and delay

Consider the following discrete time linear stochastic plant:

$$x_{t+1} = Ax_t + w_t \tag{5}$$
$$y_t = Cx_t + v_t, \tag{6}$$

where $t \in \mathbb{N}$, $x, w \in \mathbb{R}^n$, $y \in \mathbb{R}^m$, $A \in \mathbb{R}^{n\times n}$, $y \in \mathbb{R}^m$, $C \in \mathbb{R}^{m\times n}$, (x_0, w_t, v_t) are Gaussian, uncorrelated, white, with mean $(\bar{x}_0, 0, 0)$ and covariance (P_0, Q, R) respectively. We also assume that the pair (A, C) is observable, $(A, Q^{1/2})$ is reachable, and $R > 0$.

Measurements are time-stamped, encapsulated into packets, and then transmitted through a digital communication network (DCN), whose goal is to deliver packets from a source to a destination. Time-stamping of measurements is necessary to reorder packets at the receiver side since they can arrive out of order. The arrival process is modeled via the random variable γ_k^t defined as follows (from now on k will indicate a time):

$$\gamma_k^t = \begin{cases} 1 & \textit{if } y_k \text{ received before or at time } t,\ t \geq k \\ 0 & \textit{otherwise} \end{cases} \tag{7}$$

We also define the packet delay $\tau_k \in \{\mathbb{N}, \infty\}$ for observation y_k as follows:

$$\tau_k = \begin{cases} \infty & \textit{if } \gamma_k^t = 0, \forall t \geq k \\ t_k - k & \textit{otherwise}, \text{ where } t_k = \min\{t \mid \gamma_k^t = 1\} \end{cases} \tag{8}$$

where t_k is the arrival time of observation y_k at the estimator. If the delay of the arriving packets is bounded, i.e. if there exists $\bar{N}$ such that $\gamma_k^t = \gamma_k^{t+1}$ for $t - k + 1 \geq \bar{N}$, then it has been shown in [10] that the minimum variance estimator $\hat{x}_{t|t}^t = \mathbb{E}[x_t \,|\, \text{arrived measurements}] = \mathbb{E}[x_t \,|\, \gamma_1^t, .., \gamma_t^t, \tilde{y}_1^t, .., \tilde{y}_t^t]$ (where $\tilde{y}_k^t = \gamma_k^t y_k$) and its corresponding prediction error covariance $P_{t+1|t}^t = \mathbb{E}[(x_{t+1} - A\hat{x}_{t|t}^t)(x_{t+1} - A\hat{x}_{t|t}^t)^T \,|\, \gamma_1^t, \ldots, \gamma_t^t]$ is given by a time-varying Kalman filter with a buffer of size $\bar{N}$ whose equations are:

$$\begin{aligned}
\hat{x}_{t-\bar{N}|t-\bar{N}}^t &= \hat{x}_{t-\bar{N}|t-\bar{N}}^{t-1}, \quad P_{t-\bar{N}+1|t-\bar{N}}^t = P_{t-\bar{N}+1|t-\bar{N}}^{t-1} \\
\hat{x}_{k|k}^t &= A\hat{x}_{k-1|k-1}^t + \gamma_k^t K_k^t(\tilde{y}_k^t - CA\hat{x}_{k-1|k-1}^t) \\
K_k^t &= P_{k|k-1}^t C^T (CP_{k|k-1}^t C^T + R)^{-1} \\
P_{k+1|k}^t &= AP_{k|k-1}^t A^T + Q - \gamma_k^t A K_k^t C P_{k|k-1}^t A^T
\end{aligned}$$

where $k = t - \bar{N} + 1, \ldots, t$, and $\hat{x}_{h|h}^t = \bar{x}_0, P_{h|h-1}^t = P_0$ for $h \leq 0$. Because the error covariance $P_{t+1|t}^t$ depends on the packet arrival sequence γ_k^t, it is time-varying and does not converge to a steady state, unlike the standard Kalman filter with no packet loss. Moreover, it requires the inversion of up to $\bar{N}$ matrices at every time step t and might be too expensive for on-line implementation. Also, the buffer size $\bar{N}$ needed for the optimal estimator might be too large. Although in theory even very old measurements help reduce the estimation error, in practice their contribution is marginal. Based on these considerations, in the next section we propose a filter which does not require any matrix inversion and whose buffer size can be reduced to trade off performance with computational complexity.

IV. Estimators with shifted buffer and constant gains

In this section we propose a suboptimal estimator design strategy which does not require any matrix inversion and has a buffer with length smaller than the maximum packet delay $\bar{N}$ in the WSN. Since we want to quantify the performance of the estimator, we need to specify the statistics of the arrival process. We assume that the packet arrival process at the estimator is stationary and i.i.d. with the following probability function:

$$\mathbb{P}[\tau_t \leq h] = \lambda_h \tag{9}$$

where $t \geq 0$, $0 \leq \lambda_h \leq 1$ is non-decreasing in $h = 0, 1, 2, \ldots, \bar{N}$, and τ_t was defined in Equation (8). Equation (9) corresponds to the probability that a packet sampled h time steps ago has arrived at the estimator. Although in reality packet arrivals might not be i.i.d. because of correlation in packet delays, the i.i.d. assumption allows us to explicitly compute the performance of the proposed estimators and to find the optimal gains within this class.

Starting from the buffer of the optimal filter described in Section III, we consider the subset of the measurements inside the same with time delays in $M, \ldots, M+N$ (the subset will be called *shifted buffer*), where $M = 0, \ldots, \bar{N}$ is the starting point of the shifted buffer, and $N = 1, \ldots, \bar{N} - M - 1$ is its length (an example is shown in Figure 4). The estimation scheme has the following structure:

$$\begin{aligned}
&\tilde{x}_{t-N-M|t-N-M}^t = \hat{x}_{t-N-M|t-N-M}^{t-1} \\
\hat{x}_{k|k}^t &= A\hat{x}_{k-1|k-1}^t + \gamma_k^t \tilde{K}_{t-k}(\tilde{y}_k^t - CA\tilde{x}_{k-1|k-1}^t), \\
&\qquad k = t-N-M+1, \ldots, t-M \\
\tilde{x}_{t|t}^t &= A^M \tilde{x}_{t-M|t-M}^t
\end{aligned} \tag{10}$$

which mimics the time-varying estimator with the buffer in the previous section, but with gains $\{\tilde{K}_h\}_{h=M}^{N+M-1}$ not depending on the packet arrival sequence γ_k^t, unlike the gains $\{K_k^t\}$ of the optimal filter of Section III.

The performance of this new estimator is measured in terms of its prediction error covariance $\tilde{P}_{t+1|t} = \mathbb{E}[(x_{t+1} - A\tilde{x}_{t|t}^t)(x_{t+1} - A\tilde{x}_{t|t}^t)^T \,|\, \gamma_1^t, ..., \gamma_t^t]$. Obviously we must have $P_{t+1|t}^t \leq \tilde{P}_{t+1|t}^t$ for every sequence γ_k^t since the filter in the previous section is the minimum variance linear filter. Just like $P^t_{t+1|t}$, the prediction error covariance $\tilde{P}^t_{t+1|t}$ is a random variable since it depends on the specific realization of the arrival process γ^t_k. Therefore, we are interested in computing the expected prediction error covariance with respect to all possible realizations of γ^t_k, i.e. $\overline{P}^t_{t+1|t} = \mathbb{E}_\gamma[\tilde{P}^t_{t+1|t}] = \overline{P}^t_{t+1|t}(\tilde{K}, N, M)$, where we made explicit the dependence on the gains $\tilde{K} = (\tilde{K}_M, \ldots, \tilde{K}_{M+N-1})$, the length of the buffer N, and its initial position M. The following theorem provides stability conditions for the proposed filter.

Theorem 1: Consider the following modified Algebraic Riccati Equation:

$$P = \Phi_\lambda(P) = APA^T + Q - \lambda APC^T(CPC^T + R)^{-1}CPA^T \quad (11)$$

and the gain $K_P = g(P) = PC^T(CPC^T + R)^{-1}$. If A is unstable, then there exists a unique positive semidefinite solution if and only if $\lambda > \lambda_c$ where:

- λ_c depends only on the pair (A, C);
- λ_c satisfies the following inequalities (where the $\sigma^u_i(A)$'s are the unstable eigenvalues of A):

$$p_{min} = \frac{1}{\prod_i |\sigma^u_i(A)|^2} \leq 1 - \lambda_c \leq \frac{1}{\max_i |\sigma^u_i(A)|^2} = p_{max}$$

- $p_{min} = 1 - \lambda_c$ if C is rank one;
- $p_{max} = 1 - \lambda_c$ if C is invertible.

If A is strictly stable, then there always exists a unique positive semidefinite solution. Consider also the class of filters defined by Equation (10), and suppose the packet arrival process is i.i.d. and following the probability function (9). If $\lambda_{M+N-1} < \lambda_c$ then $\lim_{t\to\infty} \sup_t \overline{P}^t_{t+1|t}(\tilde{K}, N, M) = \infty$ for any choice of the gains $\tilde{K}$. If $\lambda_{M+N-1} > \lambda_c$, then consider the following semipositive definite matrices:

$$\begin{array}{l} V_{M+N-1} = \Phi_{\lambda_{M+N-1}}(V_{M+N-1}) \\ V_k = \Phi_{\lambda_{k+1}}(V_{k+1}), \quad k = M+N-2, \ldots, M \\ V_k = AV_{k+1}A^T + Q = \Phi_0(V_{k+1}), \quad k = M-1, \ldots, 0 \end{array} \quad (12)$$

and the gains $\tilde{K}^*_k = g(V_k), k = M+N-1, \ldots, M$. Then:

$$\begin{array}{l} \lim_{t\to\infty} \overline{P}^t_{t+1|t}(\tilde{K}^*, N, M) = V_0(N, M) \\ \lim_{t\to\infty} \overline{P}^t_{t+1|t}(\tilde{K}, N, M) \geq V_0(N, M), \quad \forall \tilde{K} \end{array}$$

Finally $V_0(N, M) \geq V_0(N+1, M)$.

Proof: The proof is a straightforward application of the results presented in [10] and is therefore omitted. ■

The previous theorem states that if the packet arrival probability for the last slot in the buffer λ_{M+N-1} is sufficiently high, then there exists a stable estimator within the class of filters proposed in this section. The theorem also shows how to find the best estimator in terms of minimum variance within this class. The best expected prediction error covariance $V_0(N, M)$ is a function of the buffer length N and initial position M. The memory and computational complexity for such estimators do not depend on M. Therefore, we would like to find the best M which minimizes $V_0(N, M)$. Unfortunately it is not possible to guarantee that there exits M^* such that $V_0(N, M^*) \leq V_0(N, M)$, and indeed this is actually false in general. To overcome this limitation, we will consider a cost function which is linear and positive in $V \geq 0$, i.e. a function $f : \mathbb{R}^{n\times n} \to \mathbb{R}^+$. Some examples are $f(V) = \mathrm{trace}(V)$ and $f(V) = z^T V z$, where $z \in \mathbb{R}^n$. Using this cost function we will compute the optimal shifted buffer M for any fixed N as:

$$M^*(N) = \arg\min_M f\left(V_0(N, M)\right)$$

and the corresponding minimum cost $v^*(N) = \min_M f(V_0(N, M))$. Since M is an integer, it is not possible to find the minimum in closed form. Therefore, we need to explicitly compute $f(V_0(N, M))$ for all M. However, this can be done off-line and then be used for on-line estimation.

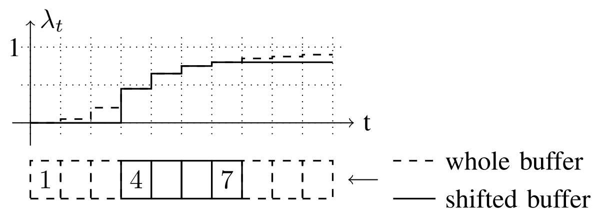

Fig. 4. Example of shifted buffer with $M = 3$ and $N = 4$; the elements of the buffer and the λ_t's used in Equation (11) are plotted with a continuous line. The dashed λ_t function refers to the trivial buffer with $M = 0$ and $N = \bar{N}$.

V. Estimation performance under UPD and DSF routing protocols

In this section we apply the tools developed in the previous section to the protocols proposed in Section II to evaluate their performance in a typical application of target tracking. A popular model for the dynamics of a moving target is given by a double integrator subject to white noise, i.e. $\ddot{\xi}_x(t) = w_x(t)$ where ξ_x is the position of the moving target along the x-axis and $w_x(t)$ is continuous time white noise with zero mean and variance q. We also assume that the position is measured through noisy sensors, i.e. $y_x(t) = \xi_x(t) + v(t)$, where $v(t)$ is zero mean white noise with variance I. The dynamics along the y-axis is modeled similarly and the noises are assumed to be uncorrelated along the two axes. The discretized dynamics of the moving target with period T can be written in state space form as follows:

$$x_{t+1} = \begin{bmatrix} G & 0 \\ 0 & G \end{bmatrix} x_t + w_t, \quad y_t = \begin{bmatrix} H & 0 \\ 0 & H \end{bmatrix} x_t + v_t$$

$$G = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix}, \; H = \begin{bmatrix} 1 & 0 \end{bmatrix}, \; S = \begin{bmatrix} \frac{T^3}{3} & \frac{T^2}{2} \\ \frac{T^2}{2} & T \end{bmatrix}$$

where $x^T = [\xi_x \, \dot{xi}_x \, \xi_y \, \dot{xi}_y]$, w_t and v_t are white Gaussian noise with covariance $Q = q \begin{bmatrix} S & 0 \\ 0 & S \end{bmatrix}$ and $R = rI$ respectively, and I is the identity matrix. It is easy to verify that $(A, Q^{1/2})$ is reachable, and (A, C) is observable. Also the critical packet arrival probability defined in the previous section is $\lambda_c = 0$, since the eigenvalues are all ones. The ratio q/r regulates the importance of measurements with respect to the state dynamics. Without loss of generality, we fix the variable $q = 1$ and we evaluate the performance of the estimators as a function of the variable r. The performance is evaluated in term of the mean square prediction error on the position of the moving target $v^*(N) = f(V_0(N, M^*(N)) = z^T V_0(N, M^*(N))z$, where $z^T = [1\,0\,1\,0]$ and $V_0(N, M)$ is the expected prediction error covariance of the estimator with a shifted buffer with size N and initial position M defined in the previous section.

First we compare the best achievable performance of estimators with constant gains as a function of the ratio q/r for the two protocols UPD and DSF. The performance of the filters with constant gains and a shifted buffer are computed using the end-to-end packet delay statistics shown in Figure 2. In particular, from the figure we see that $\lambda_h^{UPD} = 0$ for $h < 10$ and $\lambda_h^{UPD} \approx 1$ for $h > 40$. Therefore $\bar{N}^{UPD} = 40$, i.e. almost all packets arrive with a delay between 10 and 40 time steps. On the other hand, $\lambda_h^{DSF} = 0$, for $h < 22$ and $\lambda_h^{DSF} = \lambda_{h+1} = 0.81$ for $h \geq 24$, which implies that $\bar{N}^{DSF} = 24$ and that with probability $p_{loss}^{DSF} = 1 - 0.81 = 0.19$ some packets are lost and all the others arrive with a delay between 22 and 24.

Figure 5 compares the best achievable performance, measured in terms of the prediction error $v^*(N) = z^T V_0(\bar{N}, 0)z$, as a function of the ratio q/r between the process noise and the measurement noise. It shows that UPD always perform better that DSF for our given routing topology and link probabilities. This is to be expected for the two extreme regimes, i.e. for large q/r an for small q/r. In fact, for large q/r the old measurements do not help reduce estimation error since x_t changes rapidly from one time step to the next. Since UPD delivers some packets with much smaller delay than DSF, it should perform better. In the other regime, i.e for small q/r, the state x_t does not change very rapidly. Here, the old measurements help to reduce the estimation error. Therefore, the only relevant parameter is the end-to-end packet loss probability, which for UPD is almost zero, while for DSF is about $p_{loss} = 0.19$. The fact that UPD performs better that DSF also for all other values of q/r, is not obvious since λ_h^{DSF} is sometimes greater than λ_h^{UPD}. For this reason the mathematical tools developed in Section IV are particularly relevant.

We now evaluate the trade off between estimation and computational complexity by adopting a measurement buffer of size $N < \bar{N}$, which is shown in Figure 6. This figure shows the performance of the filters as a function of the buffer length N for three different noise ratio q/r regimes: small, medium, and high. The buffer shift $M^*(N)$ has been chosen optimally for each buffer length. As expected, the performance for DSF becomes constant for buffers of length $N > 3$ since all packets arrives with delay $h \in \{22, 23, 24\}$. On the other hand, the performance of UPD should continue to improve until $N = 30$, the range of delay of the packets that arrive at the receiver, while it appears to be constant for $N > 30$. Indeed the performance improves monotonically

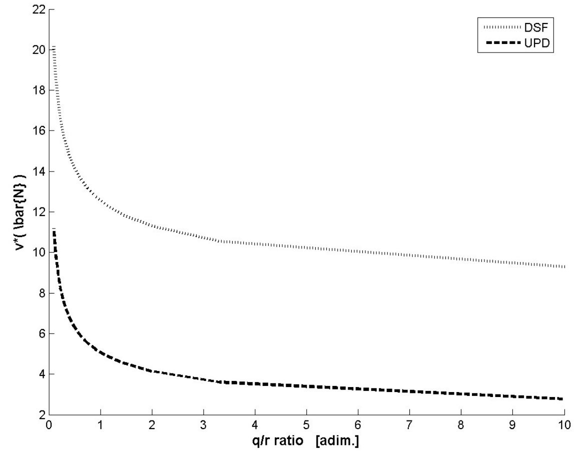

Fig. 5. Estimation error cost $v^*(\bar{N})$ as a function of the ratio q/r with $q = 1$ for the UPD and DSF protocols.

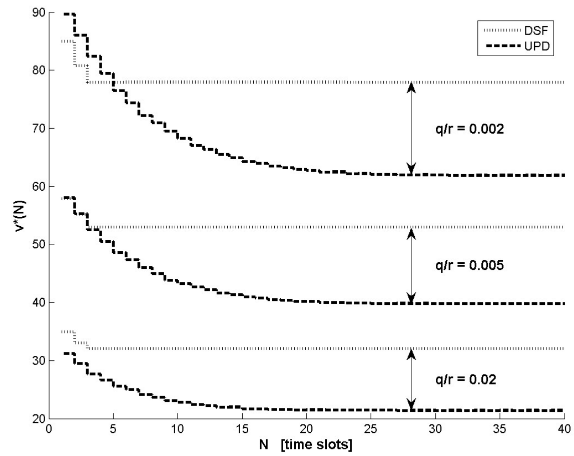

Fig. 6. Estimation error cost $v^*(N)$ as a function of the buffer length N considering optimally buffer shift $M^*(N)$ evaluated for three different ratio of q/r.

until $N = 30$, but the improvement after $N > 20$ is so small that it is irrelevant to use longer buffers. These curves can be used by the control designer to choose the proper buffer size to tune the computational complexity and performance. Note that if q/r is sufficiently small, then using DSF and a very small buffer performs better than UPD. Therefore, it is not possible to claim that UPD is always superior to DSF, but it depends on several factors such as the ratio of q/r, the dynamics given by the matrix A, and the size of the buffer N.

Finally, in Figure 7 we show the optimal buffer shift $M^*(N)$ as a function of the buffer length. As one would expect, the optimal $M^*_{DSF}(N)$ is around 22, which is the minimum delay experienced by the measurements. For $N > 3$, the performance becomes constant as explained above, and the optimal $M^*(N)$ is not unique, since any buffer which includes packet with delay $h = 22, 23, 24$ would perform optimally. Therefore, after $N = 3$, every point between the 2

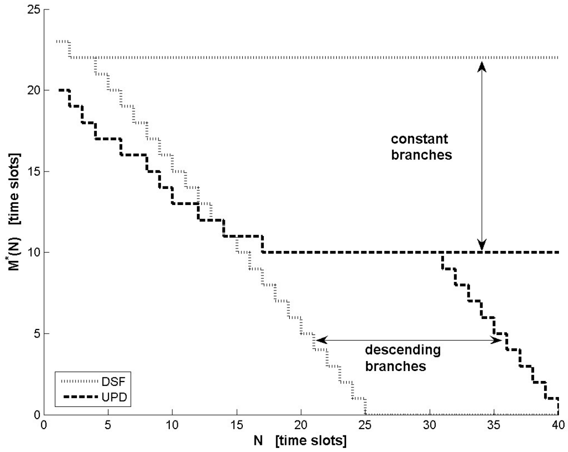

Fig. 7. Optimal buffer shift $M^*(N)$ as a function of the buffer length N for $q = 1$ and $r = 0.005$.

branches labeled as "descending" and "constant" and relative to the DSF curve is equivalent to the others. More interesting is the curve for UPD, since it is not clear for small N which is the optimal buffer shift M^*. In fact if M is chosen to be small, then only a small fraction of the measurements will be used since λ_M is small. On the other hand, if M is large so that λ_M is large, then the packets used for the estimator have a large delay $\tau \approx M$ and therefore provide little information about current target state x_t. The curve of M^* for UPD shows how the optimal shift M^* initially becomes smaller and smaller as N increases, indicating that the buffer grows mostly leftward, in order to add packets with smaller delays. However, it stops when $M^* = 10$ since $\lambda_h^{UPD} = 0$ for $h < 10$, i.e. no packets arrive with delay smaller than 10. After this point, the buffer grows to the right and include the small fraction of packets that arrive with larger delays; once the growth to the right is anymore feasible, we encounter a situation similar to the DSF case; each point between the descending and the constant branches of the UPD curve is equivalent to the others.

VI. Conclusion

In this paper we apply recent tools developed for evaluating the performance of filters when measurements are subject to packet loss and random delay [10] to two different WSN communication protocols specifically designed for real-time monitoring and tracking [5]. We also propose a new set of estimators with constant gains and a shifted buffer, which allows the design to trade off computational complexity and performance. In particular, we show that unless all the packet delay probabilities λ_h^a of a communication protocol a are greater than the relative λ_h^b's of another protocol b, it is not possible to claim that one protocol is better than the other in absolute terms. The performance of a communication protocol for a NCS depends on the ratio between process noise and measurement noise q/r, the dynamics of the system A, and the buffer length N. Nonetheless, the tools developed in this paper can be readily used by a control engineer to compare protocols for a specific application.

There are still several research avenues which deserve to be explored. The first is that the performance was evaluated in terms of estimation error covariances averaged over all possible packet delay realizations, while it would be important also to know the spreading of these covariance along a typical realization. This spreading is directly related to the jitter experienced by the estimation error which is known to give rise to poor control performance. A second research direction is to extend this work to handle non-i.i.d packet arrival processes. In fact, correlated delays between consecutive packets is very typical in most WSN communication protocols.

References

[1] Dust Networks, Inc., "Technical overview of time synchronized mesh protocol (TSMP)," http://www.dustnetworks.com/docs/TSMP_Whitepaper.pdf, 2006.

[2] A. Rowe, R. Mangharam, and R. Rajkumar, "RT-Link: A time-synchronized link protocol for energy-constrained multi-hop wireless networks," in *3rd Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks (SECON)*, vol. 2, 2006, pp. 402–411.

[3] A. Bonivento, C. Fischione, L. Necchi, F. Pianegiani, and A. Sangiovanni-Vincentelli, "System level design for clustered wireless sensor networks," *IEEE Transactions on Industrial Informatics*, vol. 3, no. 3, pp. 202–214, Aug. 2007.

[4] P. G. Park, C. Fischione, A. Bonivento, K. H. Johansson, and A. Sangiovanni-Vincentelli, "Breath: a self-adapting protocol for wireless sensor networks in control and automation," submitted for publication.

[5] P. Chen and S. Sastry, "Latency and connectivity analysis tools for wireless mesh networks," in *Proceedings of the First International Conference on Robot Communication and Coordination (ROBOCOMM)*, October 2007.

[6] J. P. Hespanha, P. Naghshtabrizi, and Y. Xu, "A survey of recent results in networked control systems," *Proceedings of the IEEE*, vol. 95, pp. 138–162, 2007.

[7] L. Schenato, B. Sinopoli, M. Franceschetti, K. Poolla, and S. Sastry, "Foundations of control and estimation over lossy networks," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 163–187, January 2007.

[8] B. Sinopoli, L. Schenato, M. Franceschetti, K. Poolla, M. Jordan, and S. Sastry, "Kalman filtering with intermittent observations," *IEEE Transactions on Automatic Control*, September 2004.

[9] J. Nilsson, B. Bernhardsson, and B. Wittenmark, "Stochastic analysis and control of real-time systems with random time delays," *Automatica*, vol. 34, no. 1, pp. 57–64, January 1998.

[10] L. Schenato, "Optimal estimation in networked control systems subject to random delay and packet loss," in *Proc. of the 45th IEEE Conference on Decision and Control*, December 2006.

[11] Y. Bar-Shalom, H. Chen, and M. Mallick, "One-step solution for the general out-of-sequence measurement problem in tracking," *IEEE Transactions on Aerospace and Electronics Systems*, vol. 40, no. 1, pp. 27–37, 2004.

[12] K.-S. Zhang, X. R. Li, and Y.-M. Zhu, "Optimal update with out-of-sequence observations for distributed filtering," *IEEE Transactions on Signal Processing*, vol. 53, no. 6, pp. 1992–2004, 2005.