

Scheduling Measurements and Controls over Networks - Part II: Rollout Strategies for Simultaneous Protocol and Controller Design

D. J. Antunes, W. P. M. H. Heemels, J. P. Hespanha, and C. J. Silvestre

Abstract— We consider a networked control system where a plant is connected to a remote controller via a shared network that allows only one user to transmit at a given time. At each transmission time, the controller decides between sampling one of the plant’s sensors or transmitting control data to the plant. We tackle the problem of simultaneously designing the scheduling sequence of transmissions and the control law so as to optimize a quadratic objective. Using the framework of dynamic programming, we propose a rollout strategy by which the scheduling and control decisions are determined at each transmission time as the ones that lead to optimal performance over a given horizon assuming that from then on controller and sensors transmit in a periodic order and the control law is a standard optimal law for periodic systems. We show that this rollout strategy results in a protocol where scheduling decisions are based on the state estimate and error covariance matrix of a Kalman estimator, and must be determined on-line. We contrast the solution to this problem with the solution to the seemingly similar sensor scheduling problem where optimal schedules can be determined off-line. We highlight how the protocol obtained from the rollout algorithm can be implemented in a distributed way both in wireless and wired networks. It follows by construction of rollout algorithms that our proposed scheduling method can outperform any periodic scheduling of transmissions. This fact is illustrated by a numerical example.

I. INTRODUCTION

Recent advances in communication and microprocessor technologies made possible several applications where a process is connected to a controller via a communication network through which it receives controls and transmits measurements. Applications of networked control systems include, e.g., energy building efficiency systems, remote surgery, highway traffic control, and platoon formation in traffic. In some of these applications, the communication protocol, e.g., the Ethernet, the CAN-BUS, or the Wireless 802.11, impose that only one user can transmit at a given time, which implies that controller and process must schedule their transmissions.

Duarte Antunes and Maurice Heemels are with the Hybrid and Networked Systems Group, Department of Mechanical Engineering, Eindhoven University of Technology, the Netherlands. {D.Antunes, M.Heemels}@tue.nl

João P. Hespanha is with the Dept. of Electrical and Computer Eng., University of California, Santa Barbara, CA 93106-9560, USA. hespanha@ece.ucsb.edu

Carlos Silvestre is with the Dep. of Electrical Eng. and Computer Science, Instituto Superior Técnico, ISR, 1046-001 Lisboa, Portugal. cjs@isr.ist.utl.pt

This work is supported by the Dutch Science Foundation (STW) and the Dutch Organization for Scientific Research (NWO) under the VICI grant Wireless controls systems: A new frontier in automation, by the European 7th Framework Network of Excellence by the project Highly-complex and networked control systems (HYCON2-257462)

In this paper, we consider a networked control system where a plant is connected to a remote controller via a shared network that allows only one user to transmit at a given time. At each transmission time, the controller decides between sampling one of the plant’s sensors or transmitting control data to the plant. This feedback setup in which the controller either samples or controls at each time step was proposed in [1], where the analysis is restricted to scalar systems. Optimal policies are computed for this class of systems in [1] and it is shown that closed-loop policies outperform in general open-loop policies. A related line of work is the sensor scheduling problem [2], [3], [4]. The pioneering work [2] considers a plant corrupted by Gaussian noise and addresses the problem of simultaneously choosing the control law and a scheduling sequence for sampling different sensors, so as to minimize the expected value of a quadratic function over a finite horizon. The fundamental difference between the present paper and [2] is that [2] assumes that the controller can update the plant’s input at every time step, while in present paper the control update may not be available at every time step due to constraints imposed by the network. In [2] it is proved that the problem can be decoupled into an optimal control problem and an optimal sensor scheduling problem, where the latter problem can be computed off-line and is combinatorial, which has prompted several researchers to proposed sub-optimal strategies (see [5] and references therein). Another line of related work results from the fact that the optimal scheduling problem for networked control systems can be put into the framework of optimal control for general switching systems. In [6], an optimal LQR-type control problem is considered for swiched system assuming that full state feedback is available. A class of suboptimal strategies are proposed to circumvent the combinatorial nature of the solution, which can be directly applied to networked control scheduling problems.

In the present paper, we tackle the problem of simultaneously designing the control law and the scheduling sequence of controller and sensors’ transmissions so as to optimize a quadratic objective over a finite horizon. In the companion paper [7], we consider the simpler case where a control law is given, and only the scheduling sequence is to be designed. Since the dimension of the search space of scheduling sequences grows exponentially with the length of the horizon, we propose the use of rollout algorithms to the problem at hand. As explained in [8], rollout algorithms consist of suboptimal strategies for dynamic programming problems in which the search for optimal decisions occurs

only along a lookahead horizon, assuming that from then on a base policy is used for which the cost to go is typically simple to determine. In our approach, we propose the base policy to be a periodic scheduling, in which nodes transmit following a prescribed order, and a corresponding standard optimal control law for the periodic system obtained from using this scheduling.

The proposed rollout algorithm results in a scheduling protocol in which sensors and controller arbitrate which one transmits at each step based on the current state estimate and on the associated error covariance matrix of a Kalman filter, which is iteratively updated based on the information sent over the network. The resulting control law is a linear feedback of the state estimate. The dependency of the scheduling decisions not only on the error covariance matrix but also on the state estimate, leads to the interesting conclusion, that the scheduling must be performed on-line, i.e., in a closed-loop fashion. This is true not only for the rollout algorithms, but also for the optimal solution to the original quadratic cost problem. Note that this is in striking contrast with the seemingly similar sensor scheduling problem [2], where the scheduling can be determined off-line. We show that our proposed protocol can be implemented in a distributed way in two cases, which encompass many networked control scenarios of interest: i) wired and wireless networked control systems in which sensors are smart, meaning that they can run an arbitration protocol; (ii) CAN-BUS wired networks, where by taking advantage of the arbitration field, we will still be able to find a distributed solution to the problem in cases where there is only one network node (instead of possibly many sensor nodes) associated with the plant, which does not necessary possess computational resources to run a scheduling algorithm, transmits and receives control data from the remote controller. Another interesting feature of our rollout algorithm is that scheduling transmissions using state information according to a rollout strategy outperforms any periodic assignment as long as this assignment is used as the base policy. We illustrate this fact with a numerical example.

The remainder of the paper is organized as follows. Section II sets up the general networked control scheduling problem. Section III addresses rollout policies and establishes our main results. We provide an illustrative example in Section IV. Section V contains concluding remarks and directions for future work. In the Appendix one can find the proofs or the results.

Notation We denote by I_n and O_n the $n \times n$ identity and zero matrices, respectively, and by $\text{diag}([A_1 \dots A_n])$ a block diagonal matrix with blocks A_i . For a matrix A , A^T denotes its transpose. For dimensionally compatible matrices A and B , we define $(A, B) := [A^T B^T]^T$.

II. PRELIMINARIES AND PROBLEM SETUP

We consider a networked control system in which a plant communicates with a remote controller via a shared network that allows only one user to transmit at a given time. The controller receives data from m sensors, which acquire a scalar measure of the plants' state, and sends data to p

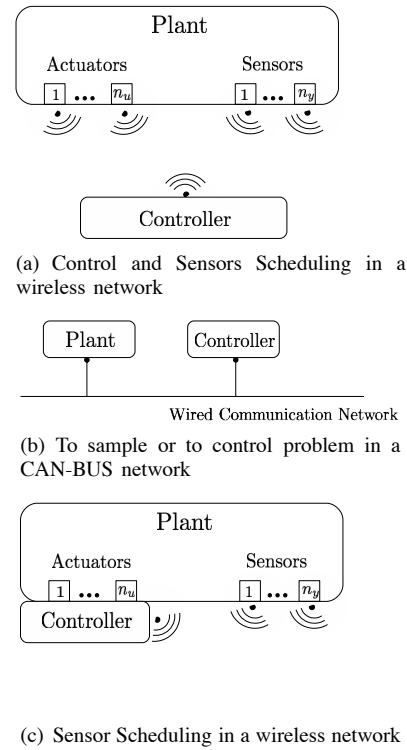


Fig. 1. Some Networked Control Scenarios modeled in our framework

actuators, which are associated with a scalar input of the plant. Transmission times are denoted by $t_k, k \geq 0$, and are assumed to be evenly spaced for simplicity, i.e., $t_{k+1} - t_k = \tau_s, \forall k \geq 0$, for some sampling period τ_s . Although, only one user can transmit through the network at a given time, it may be the case that at a transmission time the controller has access to measurements from more than one sensor, e.g., connected to the same adapter that transmits data over the network, or it may also occur be that the controller may update more than one actuator. To model these scenarios, we consider that at each transmission time, n_D options are available for data exchange between controller and plant, which can be encapsulated in the following n_D matrices

$$\Gamma_j = \text{diag}(a_1^j, \dots, a_p^j), \Omega_j = \text{diag}(b_1^j, \dots, b_m^j), \quad j \in \mathcal{M}, \quad (1)$$

where $a_i^j = 1$ if sensor i is sampled in option j , and $a_i^j = 0$ otherwise, and $b_i^j = 1$ if actuator i is sampled in option j , and $b_i^j = 0$ otherwise. The set $\mathcal{M} := \{1, \dots, n_D\}$ denotes the set of options. We introduce the scheduling sequence,

$$\sigma_k \in \mathcal{M}, \quad \text{for } k \in \mathcal{K}, \quad (2)$$

which indicates that at the time t_k , σ_k is the node that transmits, where $\mathcal{K} := \{0, \dots, k_F\}$, with $k_F \in \mathbb{N} \cup \{\infty\}$, denotes the set of the time instants of interest.

This modeling framework can be used to model several networked control scenarios of interest. Figure 1 depicts three scenarios, which have a special interest to the present paper, and can be described as follows.

- a) *Smart sensors in broadcast networks.* In this scenario we make two assumptions i) the sensors have enough

computational resources to run an arbitration protocol; and (ii) every node can listen to the network at every transmission time (although only one can transmit). A network node may transmit several measures and therefore we group sensors into n_y sets each associated with $s_i, 1 \leq i \leq n_y$ scalar measurements. Likewise, each node in the network may be associated with several inputs of the plant and therefore we group actuators into n_u sets each associated with $r_i, 1 \leq i \leq n_u$ inputs. At every transmission time either the controller transmits to an actuator node $1 \leq j \leq n_u$, which we label by options $1 \leq j \leq n_u$ or a sensor node $1 \leq i \leq n_y$ can transmit to the controller, which we label as options $n_u + 1 \leq j = n_u + i \leq n_u + n_y$. Then, the matrices (1) take the form

$$\Omega_i = \begin{cases} \text{diag}(0_{r_1}, \dots, 0_{r_{i-1}}, I_{r_i}, 0_{r_{i+1}}, \dots, 0_{r_{n_u}}), \\ \quad \text{if } 1 \leq i \leq n_u \\ 0, \text{ if } n_u + 1 \leq i \leq n_u + n_y, \end{cases} \quad (3)$$

and

$$\Gamma_i = \begin{cases} 0, \text{ if } 1 \leq i \leq n_u \\ \text{diag}(0_{s_1}, \dots, 0_{s_{i-1}}, I_{s_i}, 0_{s_{i+1}}, \dots, 0_{s_{n_u}}), \\ \quad \text{if } n_u + 1 \leq i \leq n_u + n_y. \end{cases} \quad (4)$$

- b) *To sample or to control over CAN-BUS.* There is only one network node (instead of possibly many sensor nodes) associated with the plant, which does not necessarily possess computational resources to run a scheduling algorithm. This node transmits and receives control data from the remote controller, to which is connect by a CAN-BUS. We shall see in sequel that by taking advantage of the arbitration field in CAN-BUS data messages, we can still run an arbitration algorithm in this setup. In terms of the matrices (1), this is a special case of case a), with $n_u = 1$ and $n_y = 1$, i.e., we can model this scenario with two options: 1) to sample, in which case $\Gamma_k = 0$, and $\Omega_k = I$; and 2) to transmit, in which case $\Gamma_k = I$, and $\Omega_k = 0$.
- c) *Sensor Scheduling.* The controller is collocated with the plant and can update the actuator at every transmission time t_k . Only the sensors need to be scheduled over the network. Grouping sensors in n_y set as in case a) options and the matrices Γ_j takes the form (4), and $\Omega_j = I_m, \forall j \in M$.

A. Problem Formulation

We consider a plant model that directly takes into account the communication constraints imposed by the network, and which takes the form

$$\begin{aligned} x[k+1] &= Ax[k] + B\Omega_{\sigma_k} u[k] + w_x[k], \quad k \geq 0 \\ y[k] &= \Gamma_{\sigma_k} Cx[k] + w_y[k] \end{aligned} \quad (5)$$

The vectors $w_x[k]$ and $w_y[k]$ are zero mean independent Gaussian processes characterized by the covariance matrices $E[w_x[k]w_x[k]^T] = \Phi_x$ and $E[w_y[k]w_y[k]^T] = \Phi_y$. The initial state $x[0]$ is assume to be a Gaussian variable, with $\mathbb{E}[x[0]] = \bar{x}_0$, and $\mathbb{E}[x[0]x[0]^T] = \Phi_0$. Note that (5) indeed capture the

fact that σ_k selects at each time step k which components of the input vector $u[k]$ can influence the plant model, and which components of the output vector $y[k]$ are available for feedback. Moreover, we assume that the actuation associated with actuator $1 \leq j \leq n_u$ is not available, which is already captured in the model (5).

We are interested in the problem

$$\min_{\sigma_k, u_k} \mathbb{E} \left[\sum_{k=0}^{k_F-1} x[k]^T Q x[k] + u[k]^T R u[k] + x[k_F]^T Q x[k_F] \right] \quad (6)$$

where the matrices Q and R are assumed to be positive definite. The set of admissible control laws $u[k]$ is the set of functions of the information available to the controller from listening the network up to time t_k , i.e., $\{y_\ell, u_\ell, \ell < k\}$.

Remark 1: Note that we do not need to impose the restrictions that $u_j[k] = 0$ at times at which the actuator j does not receive an update, since its value at these times does not affect (5). In fact, since $u_j[k]$ is always weighted in the performance cost it is automatically set to zero at these times by the optimal and suboptimal controllers that we propose in the sequel (cf. Remark 7). Likewise, we do not need to assume that the controller does not know $y_i[k]$ at times at which sensor i does not send data to the controller. In fact, $y_i[k]$ is purely noise at these times, and it is automatically not taken into account by the optimal and suboptimal controllers that we propose in the sequel (cf. Remark 7).

III. MAIN RESULTS

In this section, we start by showing how to compute the cost (6) for a periodic base policy in subsection III-A, and then we present the scheduling protocol that a rollout policy with this periodic base policy leads to in Subsection III-B. In Subsection III-C, we address the implementation of this scheduling protocols in the scenario of Fig. 1.

A. Base Policy

To defined a periodic scheduling, we consider a set of h consecutive schedules for σ_k denoted by

$$(v_0, \dots, v_{h-1}), \quad (7)$$

where $v_\ell \in \mathcal{M}$, $0 \leq \ell \leq h-1$, which are periodically repeated as explained next. If we let $[k]_h$ denote the remainder after division of k by h , we have

$$\sigma_k = \theta_k^\kappa, \quad k \in \{0, \dots, k_F\}, \quad (8)$$

where

$$\theta_k^\kappa := v_{[k+\kappa]_h}, \quad k \geq 0, \quad (9)$$

for some $0 \leq \kappa \leq h-1$ that characterizes the initial condition of the periodic scheduling θ_k^κ .

Once the periodic scheduling is fixed, we can obtain the optimal standard solution to the problem (5), (6), which we use as the base policy for the control law. This optimal

solution, which can be found e.g., in [9] is summarized in the next Proposition. Let

$$\begin{aligned} F_j(P) &:= A^\top P A + Q \\ &\quad - A^\top P B \Omega_j (R + \Omega_j B^\top P B \Omega_j)^{-1} \Omega_j B^\top P A \\ G_j(P) &:= -(R + \Omega_j B^\top P B \Omega_j)^{-1} \Omega_j B^\top P A \\ H_j(N) &:= A N A^\top + \Phi_x \\ &\quad - A N C^\top \Gamma_j (\Phi_y + \Gamma_j C N C^\top \Gamma_j)^{-1} \Gamma_j C N A^\top \\ M_j(N) &:= -A N C^\top \Gamma_j (\Phi_y + \Gamma_j C N C^\top \Gamma_j)^{-1} \end{aligned} \quad (10)$$

Proposition 2: The solution to the problem (5), (6) in the case where the scheduling is periodic and described by (8) is given by

$$u[k] = K_k \hat{x}[k] \quad (11)$$

where $\hat{x}[k]$ is obtained from

$$\begin{aligned} \hat{x}[k+1] &= (A + B \Omega_{\theta_k^\kappa} K_k + L_k \Lambda_{\theta_k^\kappa} C) \hat{x}[k] - L_k y[k] \\ u[k] &= K_k \hat{x}[k], \quad \hat{x}[0] = \bar{x}_0 \end{aligned} \quad (12)$$

and the gains K_k and L_k , $k \in \mathcal{K}$ are obtained from

$$\begin{aligned} P_K &= Q, \quad P_{k-1} = F_{\theta_k^\kappa}(P_k), \quad k_F - 1 \geq k \geq 1 \\ K_{k-1} &= G_{\rho_{k-1}}(P_k) \end{aligned} \quad (13)$$

and

$$\begin{aligned} N_0 &= \Phi_0, \quad N_{k+1} = H_{\theta_k^\kappa}(N_k), \quad 0 \leq k \leq H-1 \\ L_k &= M_{\theta_k^\kappa}(Q_k) \end{aligned} \quad (14)$$

Moreover, the optimal cost (6) is given by $J^{\text{base}}(\hat{x}[0], \kappa)$, where

$$\begin{aligned} J^{\text{base}}(\hat{x}, \Phi_0, \kappa) &= \hat{x}^\top P_0 \hat{x} + \text{tr}(P_0 \Phi_0) + \sum_{k=0}^{k_F-1} \text{tr}(P_{k+1} \Phi_x) + \\ &\quad \sum_{k=0}^{k_F-1} \text{tr}(N_k A^\top P_{k+1} B \Omega_{\theta_k^\kappa} (R + \Omega_{\theta_k^\kappa} B^\top P_{k+1} B \Omega_{\theta_k^\kappa})^{-1} \Omega_{\theta_k^\kappa} B^\top P_{k+1} A). \end{aligned} \quad (15)$$

□

B. Rollout Policy

We propose to choose at each iteration the control law and the node to transmit as the ones that leads to optimal performance over a fixed lookahead horizon, assuming that from then on a periodic base policy is used. In other words, at each iteration ℓ , $0 \leq \ell \leq k_F - 1$, the schedules

$$\sigma_\ell, \sigma_{\ell+1}, \dots, \sigma_{\ell+H-1}$$

and the controls

$$u_\ell, u_{\ell+1}, \dots, u_{\ell+H-1}$$

are assumed to be free variables, where H denotes the length of the lookahead horizon, while

$$\sigma_{\ell+H}, \sigma_{\ell+H+1}, \dots$$

and

$$u_{\ell+H}, u_{\ell+H+1}, \dots$$

are fixed and follow a periodic policy as in (8), (9), and (24), (12). The free scheduling variables are denoted by $\nu = (\nu_0, \dots, \nu_{H-1})$, i.e.,

$$\sigma_k = \nu_{k-\ell}, \quad \text{for } k \in \{\ell, \dots, \ell + H - 1\} \quad (16)$$

and the fixed scheduling variables can be written as

$$\sigma_k = \theta_{k-(\ell+H)}^\kappa, \quad \text{for } k \in \{\ell + H, \dots, k_F - 1\}. \quad (17)$$

Note that at time $\ell + H$ the base policy is assumed to start at an initial schedule ν_κ , determined by κ . We consider that $\kappa \in \{0, \dots, h-1\}$ is also a decision variable, and the decision set is denoted by $\mathcal{I} := \mathcal{M}^H \times \{0, \dots, h-1\}$, i.e., $(\nu, \kappa) \in \mathcal{I}$. The length of the lookahead horizon is a fixed constant, but naturally needs to be adapted when the iteration step is close to the terminal step time k_F , i.e.,

$$H(\ell) := \min(H_c, k_F - 1 - \ell), \quad (18)$$

where $1 \leq H_c \leq k_F - 1$ is a constant. The dependency of H on ℓ is omitted hereafter. The process is restarted at each step, in a similar fashion as in Model Predictive Control (MPC) [10].

As we shall see in the sequel this procedure boils down to the following protocol.

Protocol 1: Given the data $\{y_k, u_k, k < \ell\}$ obtained by listening to the network, at each iteration ℓ choose

$$\sigma_\ell = \bar{\nu}_{\ell 0}, \quad (19)$$

where $\bar{\nu}_{\ell 0}$ is the first entry of the vector $\bar{\nu}_\ell = (\bar{\nu}_{\ell 0}, \dots, \bar{\nu}_{\ell H-1})$ obtained from

$$(\bar{\nu}_\ell, \bar{\kappa}_\ell) = \underset{(\nu, \kappa) \in \mathcal{I}}{\text{argmin}} \hat{x}_\ell^\top P_{\nu, \kappa, \ell} \hat{x}_\ell + \delta(\nu, \kappa, \ell) + \beta(\nu, \kappa, \ell, N_\ell), \quad (20)$$

and the quantities in (20) are determined as follows.

- At iteration ℓ , compute $P_{\nu, \kappa, \ell} = \tilde{P}_0$, from

$$\tilde{P}_H = \bar{P}_H, \quad \tilde{P}_j = F_{\nu_j}(\tilde{P}_{j+1}), \quad H-1 \geq j \geq 0, \quad (21)$$

where \bar{P}_H is obtained from

$$\bar{P}^{k_F} = Q, \quad \bar{P}_j = F_{\theta_j^\kappa}(\bar{P}_{j+1}), \quad k_F - 1 \geq j \geq H. \quad (22)$$

Moreover, compute

$$\delta(\nu, \kappa, \ell) = \sum_{j=0}^{H-1} \text{tr}(\tilde{P}_j \Phi_x) + \sum_{j=H}^{k_F-1} \text{tr}(\bar{P}_j \Phi_x).$$

- The N_ℓ and \hat{x}_ℓ are obtained recursively by at each iteration $k \leq \ell$ updating the following iterations based on past schedules

$$N_0 = \Phi_0, \quad N_{k+1} = H_{\sigma_k}(N_k), \quad k < \ell, \quad (23)$$

and

$$\hat{x}[k+1] = (A + B \Omega_{\sigma_k} K_k + L_k \Lambda_{\sigma_k} C) \hat{x}[k] - L_k y_k \quad (24)$$

where $L_k = I_{\sigma_k}(N_k)$, $K_k = G_{\sigma_k}(\tilde{P}_1)$, and \tilde{P}_1 is the matrix obtained by running (21), (22) at iteration $k < \ell$.

- The function β can be computed by

$$\beta(\nu, \kappa, \ell, \ell) = \text{tr}(\tilde{N}_\ell \tilde{P}_0) + \sum_{j=0}^{H-1} \text{tr}(\tilde{N}_j (A^\top \tilde{P}_{j+1} B \Omega_{\nu_j} (R + \Omega_{\nu_j} B \tilde{P}_{j+1} B \Omega_{\nu_j})^{-1} \Omega_{\nu_j} B^\top \tilde{P}_{j+1} A) + \sum_{j=H}^{k_F-1} \text{tr}(\tilde{N}_j (A^\top \tilde{P}_{j+1} B \Omega_{\theta_j^\kappa} (R + \Omega_{\theta_j^\kappa} B \tilde{P}_{j+1} B \Omega_{\theta_j^\kappa})^{-1} \Omega_{\theta_j^\kappa} B^\top \tilde{P}_{j+1} A).$$

where \tilde{N}_k and \tilde{N}_k are obtained at iteration ℓ from

$$\tilde{N}_{j+1} = H_{\nu_j}(\tilde{N}_j), 0 \leq j \leq H-1, \quad \tilde{N}_0 = N_\ell$$

$$\tilde{N}_{j+1} = H_{\theta_j^\kappa}(\tilde{N}_j), H \leq j \leq k_F-1, \quad \tilde{N}_H = \tilde{N}_H$$

Moreover, the control law u_ℓ is given by

$$u_\ell = K_\ell \hat{x}_\ell, \quad (25)$$

where $K_\ell = G_{\sigma_\ell}(\tilde{P}_1)$, \tilde{P}_1 is the matrix obtained by running (21), (22) at iteration ℓ , and $\hat{x}[\ell]$ is obtained from (24). \square

As stated next, this protocol does in fact correspond to the rollout algorithm described above, and it always outperforms the corresponding periodic base policy.

Theorem 3: The rollout scheduling algorithm with the base policy described in Subsection III-A is determined by the Protocol 1. Moreover

$$J^{\text{dyn}}(\hat{x}, \Phi) \leq \min_{0 \leq \kappa \leq k-1} J^{\text{base}}(\hat{x}, \Phi, \kappa) \quad (26)$$

where $J^{\text{dyn}}(x)$ is the cost (6) when the scheduling sequence σ_k and the control law u_k for the system (5) are as described by Protocol 1, starting with an initial state estimate $x_0 = \hat{x}$ and associated covariance matrix $\Phi_0 = \Phi$. \square

We state next four important remarks about the Protocol 1.

Remark 4: (closed-loop policy)

A very important feature of the rollout policy just described is that since the scheduling selection (20) depends on the state estimation $\hat{x}[\ell]$ at time t_ℓ , updated taking into account measurements $y_j[k]$ (cf. (24)), which in turn are corrupted by noise. Thus, the scheduling sequence of the rollout algorithm cannot be determined a priori, or in an open-loop fashion, as it is the case, for the sensor scheduling problem, discussed in the sequel. The rollout policy for the control law, given by (25), is a closed-loop policy, since it depends on $\hat{x}[\ell]$. As we shall see, this is also the case for the sensor scheduling problem.

Remark 5: (the case of a optimal policy)

The dimension of the decision set \mathcal{I} , given by $n_D^H \times h$, grows exponentially with the length of the lookahead horizon H , and therefore the length of the lookahead horizon can only correspond to a few schedules, to make the computation of Protocol 1 viable. However, from a theoretical point of view, if we make the lookahead horizon sufficiently large to cover the entire horizon, i.e., making $H_c > k_F$ in (18), then the rollout policy degenerates into the search for an optimal policy. Thus, also the search for an optimal policy leads to a closed loop policy both for the control law and for the scheduling sequence.

Remark 6: (Implementation) The Protocol 1 is run in the nodes of the network in a distributed way, as we shall see in Subsection III-C for the scenarios of Fig. 1. An important observation, is that every information that the nodes require to compute the arbitration function (20) is available at time $t_{\ell-1}$ and therefore computations can be done between time $t_{\ell-1}$ and t_ℓ , in a way that there are no delays at time t_ℓ . The feasibility of this implementation depends on the computational capacity of the nodes, i.e., whether they can perform these calculations in a time interval no larger than one sampling interval h .

Remark 7: (connection to Remark 1) The equations (24) and (25). confer what was stated intuitively in Remark 7. In fact, if an actuator $1 \leq i \leq m$ does not receive an update at time t_ℓ then the corresponding entry $b_i^{\sigma_k}$ of the matrix Ω_{σ_ℓ} is zero, and from the structure of the gains $K_\ell = G_{\sigma_\ell}(\tilde{P}_1)$ where G is defined in (10) we also have that $u_j[k] = 0$. Likewise, if a sensor $1 \leq i \leq n_y$ does not transmit at time t_ℓ then the corresponding entry $a_i^{\sigma_k}$ of the matrix Γ_{σ_ℓ} will be zero, and from the structure of the gains $L_\ell = I_{\sigma_\ell}(N_\ell)$ where I is defined in (10) we have that $y_j = 0$ is not taken into account in (24).

C. Implementation of the Protocol 1 in the scenarios of Fig. 1

1) *Distributed implementation in broadcast networks with smart sensors:* The key assumption that every node can listen to the network at each transmission time, allows every node to compute \hat{x}_ℓ , and also Q_ℓ , at time t_ℓ . In fact, knowing the data sent over the network at times $\{t_k < t_\ell, k \geq 0\}$, nodes can locally iterate (24) and (23). Moreover, since the knowledge of \hat{x}_ℓ , and also Q_ℓ , at time t_ℓ is all one needs to run the Protocol (1), each node can implement the same rollout algorithm independently and therefore determine which node is the next to transmit. At each transmission step, the node that gains arbitration simply transmits and the other nodes do not. This implementation does not assume anything from the network, e.g., whether it is wired or wireless, as opposed to e.g., the solution proposed in [11] for CAN-BUS networks and in Subsection III-C.2 for the scenario b) in Fig.1, where the assumption that we have available an arbitration field that is used in the network messages plays an essential role.

2) *To sample or to control in CAN-BUS networks:* Here, sensors are no longer assumed to be smart, i.e., they cannot run an arbitration algorithm. Still, in the case where there is only one network node, corresponding to the plant, that transmits the measurement information to the controller and receives the actuation data from the controller, and the communication network is the CAN-BUS, we can still implement the Protocol 1 as follows. At every time t_k the plant simply transmits a message which encapsulates the data from the measurements. Therefore only the controller runs the arbitration Protocol 1, and decision are just to sample or to control. To sample the controller does not transmit. To control the controller can use the arbitration field in the CAN-BUS message it sends to gain priority over the sensor, and therefore the plant receives the control update.

3) *Sensor scheduling problem:* For this special case the computation of the matrices $\tilde{P}_j, 0 \leq j \leq H$ and $P_{\nu, \kappa, \ell}$ obtained from (21) do not depend on ν , and therefore (20) does not depend on $\hat{x}[k]$, i.e., the optimal schedules at time t_ℓ only depend on the error covariance matrix N_ℓ , which can be computed offline from the recursion (23). Note also that the rollout policy (when H is fixed and typically much smaller than k_F) in this case can be viewed as a suboptimal strategy to determine sensor schedules, which can be computed offline.

IV. ILLUSTRATIVE EXAMPLE

Consider the problem of controlling an inverted pendulum by a remote controller, which is connected to the pendulum system through a communication network as the CAN-BUS, Wireless 802.11, or the Ethernet. Thus, the controller has to choose at each transmission time either to send a control action (torque) for the pendulum or obtain a measurement, which is assumed to be a measurement of the displacement angle. Control decisions at time t_k correspond to $\sigma_k = 1$, and measurement decision to $\sigma_k = 2$. We assume that transmissions occur with a sampling period of $\tau_s = 0.1$. A discretized model of a linearization of the inverted pendulum about the unstable equilibrium point is given by (5), with

$$A = e^{A_P \tau_s}, \quad B = \int_0^{\tau_s} e^{A_P s} B_P ds, \quad C = [1 \ 0],$$

where

$$A_P = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad B_P = \begin{bmatrix} 0 \\ 1 \end{bmatrix},$$

and $\Gamma_1 = 0, \Gamma_2 = 0, \Omega_1 = 1, \Omega_2 = 0$. The noise signals in (5) are characterized by

$$\Phi_0 = 0.1I_2, \quad \Phi_x = 0.001I_2, \quad \Phi_y = 0.001$$

The performance criterion is given,

$$\mathbb{E} \left[\sum_{k=0}^{k_F-1} (Cx[k])^2 + u[k]^2 dt + (Cx[k_F])^2 \right] \quad (27)$$

with $k_F = 100$ and the controller aims at finding simultaneously the scheduling sequence σ_k and the control law u_k that minimize (27).

Consider for the base policy a periodic scheduling law that alternates between sampling and control, i.e, with period 2. We can compute from Proposition 2 the cost (27) when this periodic scheduling and the optimal control law given in Proposition 2 are applied to (5) with the matrices described above. However, we have no closed-form solution to compute the cost (27) when the scheduling is obtained by running the Protocol 1. Thus, we estimate it by running Monte Carlo simulations of the protocol. The results for the base and rollout policies are shown in Table I. The cost for the base policy can also be confirmed by running Monte Carlo simulations. The values in Table I corroborate Theorem 3, i.e., the cost obtained with the rollout policy is less than the one obtained with the base policy. In fact, a significant improvement of about 22 percent of the base policy cost is achieved.

Periodic	Rollout
88.4	68.7

TABLE I

PERFORMANCE OF BASE (PERIODIC) AND ROLLOUT POLICIES FOR THE PENDULUM EXAMPLE

V. CONCLUSIONS

In this paper we explored the use of rollout algorithms to the problem of simultaneous designing a control law and a scheduling transmission sequence for networked control systems. The rollout algorithm that we proposed determines that at each step, the transmission decision should be taken as the one that leads to optimal performance over a finite lookahead horizon assuming that from then on a fixed periodic base policy is used. Simulation results show that rollout algorithms can outperform periodic schedulings, while keeping computations within reasonable limits. We also addressed the relation of the work with the sensor scheduling problem.

APPENDIX

Proof: (of Theorem 3) The first part follows by construction of the rollout algorithm, and we omit the derivation. To prove the second part, let

$$J^{\text{rollout}}(\hat{x}[\ell], N_\ell, \ell) = \hat{x}_\ell^\top P_{\nu, \kappa, \ell} \hat{x}_\ell + \delta(\nu, \kappa, \ell) + \beta(\nu, \kappa, \ell, N_\ell), \quad (28)$$

denote the estimate of the cost to go at a given step ℓ , i.e., the minimum cost one obtains by optimizing (20). Moreover, let σ_k^{dyn} denote the scheduling sequence obtained by running the Protocol 1 and x^{dyn} denote the state $x[k]$ when this protocol is utilized. It is clear that

$$J^{\text{rollout}}(\hat{x}[0], N_0, \ell) \leq \min_{\kappa} J^{\text{base}}(\hat{x}[0], N_0, 0, \kappa) \quad (29)$$

since we can choose v_ℓ equal to the base policy in the optimization in (28). It is also clear that

$$J^{\text{rollout}}(\hat{x}[\ell], N_\ell, \ell) \geq \mathbb{E}[x[\ell]^\top Q x[\ell] + J^{\text{rollout}}(\hat{x}[\ell+1], N_{\ell+1}, \ell+1)] \quad (30)$$

since at iteration $\ell+1$ the optimization on v^ℓ takes into account the policy obtained at time step ℓ corresponding to the optimization in v^ℓ . By iteratively replacing the left hand side of (30) into the right hand side we obtain

$$J^{\text{rollout}}(\hat{x}[0], N_0, 0) \geq \sum_{k=0}^{k_F-1} x^{\text{dyn}}[k]^\top Q x^{\text{dyn}}[k] + x^{\text{dyn}}[k_F]^\top Q x^{\text{dyn}}[k_F] = J^{\text{dyn}}(\hat{x}[0], N_0). \quad (31)$$

Using (31) and (28) we obtain the desired result. \blacksquare

REFERENCES

- [1] O. Imer and T. Basar, "To measure or to control: optimal control with scheduled measurements and controls," in *American Control Conference, 2006. ACC '06.*, Jul 2006.
- [2] L. Meier, J. Peschon, and R. Dressler, "Optimal control of measurement subsystems," *Automatic Control, IEEE Transactions on*, vol. 12, no. 5, pp. 528 – 536, Oct 1967.

- [3] J. L. Ny, E. Feron, and M. A. Dahleh, "Scheduling kalman filters in continuous time," in *American Control Conference, St Louis, MO, Jun. 2009*, jun 2009.
- [4] W. Zhang, M. P. Vitus, J. Hu, A. Abate, and C. J. Tomlin, "On the optimal solutions of the infinite-horizon linear sensor scheduling problem," in *49th IEEE Conference on Decision and Control December 15-17, 2010*, 30 2010-july 2 2010, pp. 396 –401.
- [5] M. Vitus, W. Zhang, A. Abate, J. Hu, and C. Tomlin, "On efficient sensor scheduling for linear dynamical systems," in *American Control Conference (ACC), 2010*, 30 2010-july 2 2010, pp. 4833 –4838.
- [6] W. Zhang, J. Hu, and J. Lian, "Quadratic optimal control of switched linear stochastic systems," Aug. 2010, submitted to *Systems and Control Letters*.
- [7] D. Antunes, W. Heemels, J. P. Hespanha, and C. Silvestre, "Scheduling measurements and controls over networks - part i: Rollout strategies for protocol design," Sep. 2011, submitted to the *American Control Conference (ACC)*, 2012.
- [8] D. P. Bertsekas, *Dynamic Programming and Optimal Control*. Athena Scientific, 2005.
- [9] K. Astrom, *Introduction to stochastic control theory*. Elsevier, 1970.
- [10] J. Maciejowski, *Predictive Control with Constraints*. Prentice Hall, 2001.
- [11] G. Walsh, H. Ye, and L. Bushnell, "Stability analysis of networked control systems," *Control Systems Technology, IEEE Transactions on*, vol. 10, no. 3, pp. 438 –446, may. 2002.