2012 American Control Conference
Fairmont Queen Elizabeth, Montréal, Canada
June 27-June 29, 2012

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 a policy for scheduling decisions and a policy for control inputs 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 scheduling decisions can be determined off-line. We highlight how the protocol obtained from the rollout algorithm can be implemented in a distributed way in broadcast networks. Moreover, it follows by construction of rollout algorithms that our proposed scheduling method can outperform any periodic scheduling of transmissions.

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, and highway traffic control. 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 Institute for Systems and Robotics, Instituto Superior Técnico, 1046-001 Lisboa, Portugal, and the Faculty of Science and Technology, University of Macau, Taipa, Macau. 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) and also supported by project FCT [PEst-OE/EEI/LA0009/2011], and by project FCT OBSERVFLY (PTDC/EEA-ACR/72853/2006)

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 problem 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. 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 the 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 propose sub-optimal strategies (see, e.g., [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 swichted system assuming that full state feedback is available. A class of suboptimal strategies are proposed to circumvent the combinatorial nature of the problem, which can be directly applied to networked control scheduling problems.

In the present paper, we tackle the problem of simultaneously designing a policy for scheduling decisions and a policy for control inputs so as to optimize a quadratic objective over a finite horizon. In the companion paper [7], we consider the case where a control law is given, and only the scheduling sequence is to be designed. We propose the use of rollout algorithms to the problem at hand, which, as explained in [8], 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.

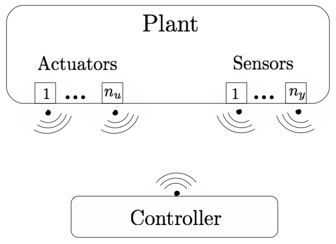
The proposed rollout algorithm results in a scheduling protocol in which sensors and controller arbitrate which one transmits at each step based on the current state estimate and on the associated error covariance matrix of a Kalman filter, which is iteratively updated based on the information sent over 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) broadcast networked control systems in which sensors can run an arbitration protocol; (ii) CAN-BUS wired networks, where by taking advantage of the arbitration field, we are still 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 with lookahead horizon one outperforms any periodic assignment as long as this assignment is used as the base policy.

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. Section IV contains concluding remarks and directions for future work.

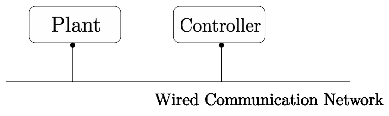
Notation We denote by I_n and O_n the $n \times n$ identity and zero matrices, respectively, and by $\text{diag}(A_1, \dots, A_n])$ a block diagonal matrix with blocks A_i. For a matrix A, $A^\intercal$ denotes its transpose.

II. Preliminaries and problem setup

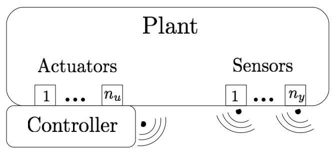
We consider a networked control system in which a plant communicates with a remote controller via a shared network that allows only one user to transmit at a given time. The controller receives measurement data from n_y sensor nodes and sends control values to n_u actuator nodes. We define a vector of controls u, and a vector of measurements y, which can be partitioned as $u = (u^1, \dots, u^{n_u})$, $y = (y^1, \dots, y^{n_y})$, where $u^i \in \mathbb{R}^{s_i}$ pertains to an actuator node $1 \leq i \leq n_u$, and $y^{i-n_u} \in \mathbb{R}^{s_i}$ pertains to a sensor node $n_u + 1 \leq i \leq n_u + n_y$. Note that, for convenience, we use the same index $1 \leq i \leq n_u + n_y$ to label both actuator and sensor nodes. With some abuse of terminology, we say that an actuator or a sensor node transmits when a transmission occurs either from the controller to an actuator, or from a sensor to the controller,



(a) Control and sensor scheduling in a broadcast network



(b) To sample or to control problem in a CAN-BUS network



(c) Sensor Scheduling

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

respectively. Transmission times are denoted by $t_k, k \in \mathbb{N}_0$, and are assumed to be evenly spaced for simplicity, i.e., $t_{k+1} - t_k = \tau_s, \forall_{k \in \mathbb{N}_0}$, for some sampling period τ_s. Figure 1 depicts three scenarios, which have a special interest to the present paper, and can be described as follows.

a) *Control and sensor scheduling in a broadcast network*. 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).
b) *To sample or to control in CAN-BUS networks*. 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. 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.
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.

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

$$x_{k+1} = Ax_k + B\Omega_{\sigma_k} u_k + w_k, \quad k \in \mathbb{N}_0$$
$$y_k = \Gamma_{\sigma_k} C x_k + v_k \tag{1}$$

where $x_k, k \in \mathbb{N}_0$ is the state, $u_k = (u_k^1, \dots, u_k^m)$ encapsulates the control inputs and $y_k = (y_k^1, \dots, y_k^p)$, encapsulates the plant's measurements. The vectors w_k and v_k are zero mean independent Gaussian processes characterized by the covariance matrices $E[w_k w_k^\intercal] = \Phi_x$ and $E[v_k v_k^\intercal] = \Phi_y$. The initial state x_0 is assume to be a Gaussian variable, with $\mathbb{E}[x_0] = \bar{x}_0$, and $\mathbb{E}[(x_0 - \bar{x}_0)(x_0 - \bar{x}_0)^\intercal] = \Phi_0$. The scheduling sequence

$$\sigma_k \in \mathcal{M}, \quad k \geq 0,$$

takes values in a set $\mathcal{M} := \{1, \dots, n_D\}$. The matrices Ω_j, $j \in \mathcal{M}$ and Γ_j, $j \in \mathcal{M}$ are diagonal matrices with zero and one entries in the diagonal and therefore 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. For the scenarios a), b), and c) depicted in Figure 1 and described above these matrices are described as follows.

a) We set $n_D = n_u + n_y$ and the following matrices capture the fact that only one node $i \in \mathcal{M}$ can transmit at a given time ℓ ($\sigma_\ell = i$)

$$\Omega_i = \begin{cases} \text{diag}(0_{\sum_{j=0}^{i-1} s_j}, I_{s_i}, 0_{\sum_{j=i+1}^{n_u} s_j}), \\ \quad \text{if } 1 \leq i \leq n_u \\ 0, \text{ if } n_u + 1 \leq i \leq n_u + n_y, \end{cases} \tag{2}$$

and

$$\Gamma_i = \begin{cases} 0, \text{ if } 1 \leq i \leq n_u \\ \text{diag}(0_{\sum_{j=n_u+1}^{i-1} s_j}, I_{s_i}, 0_{\sum_{j=i+1}^{n_u+n_y} s_j}), \\ \quad \text{if } n_u + 1 \leq i \leq n_u + n_y. \end{cases} \tag{3}$$

b) We can model this scenario with $n_D = 2$ options: 1) to sample ($\sigma_k = 2$), in which case $\Gamma_2 = I_{n_u+n_y}$, and $\Omega_2 = 0_{n_u+n_y}$; and 2) to control ($\sigma_k = 1$), in which case $\Gamma_1 = 0_{n_u+n_y}$, and $\Omega_1 = I_{n_u+n_y}$.

c) We set $n_D = n_y$ and $i \in \mathcal{M}$ corresponds to a sensor node $1 \leq i \leq n_y$ transmitting. Thus, the matrices Γ_i, $i \in M$ are given by

$$\Gamma_i = \text{diag}(0_{\sum_{j=1}^{i-1} s_{j+n_u}}, I_{s_{i+n_u}}, 0_{\sum_{j=i+1}^{n_y} s_{j+n_u}}),$$

for $1 \leq i \leq n_y$. Since control actuation can be provided to the plant at every time instant we have

$$\Omega_j = I_m, \forall j \in M.$$

Consider the following cost

$$J(\underline{u}, \underline{\sigma}) = \mathbb{E}[\sum_{k=0}^{k_F-1} x_k^\intercal Q_{\sigma_k} x_k + u_k^\intercal R_{\sigma_k} u_k + x_{k_F}^\intercal \underline{Q}_{k_F} x_{k_F}] \tag{4}$$

for positive definite matrices Q_j, $j \in \mathcal{M}$, $\underline{Q}_{k_F}$, and R_j, $j \in \mathcal{M}$, where x_0 in the initial condition of (1), and $\underline{u} := (u_0, \dots, u_{k_F-1})$ and $\underline{\sigma} := (\sigma_0, \dots, \sigma_{k_F-1})$ are decision variables. Due to the network constraints, we have that $(\sigma_k, u_k) \in \mathcal{U}, k \in \mathcal{K}$, where

$$\mathcal{U} := \{(i, \Omega_i v) \mid i \in \mathcal{M}, v \in \mathbb{R}^m\}.$$

Since we will be interested in networked control problems in which the controller has access to the information previously transmitted in the network we define the information vector $\mathcal{I}_k$ available to the controller at time k as

$$\mathcal{I}_k := \{\Gamma_{\sigma_\ell} y_\ell, \Omega_{\sigma_\ell} u_\ell, \sigma_\ell | 0 \leq \ell < k\} \tag{5}$$

A policy

$$\pi = \{(\mu_0^\sigma(I_0), \mu_0^u(I_0)), (\mu_1^\sigma(I_1), \mu_1^u(I_1)), \dots, \}$$

for

$$(\sigma_k, u_k) = (\mu_k^\sigma(I_k), \mu_k^u(I_k)), \quad k \in \mathcal{K} \tag{6}$$

is called admissible if $(\mu_k^\sigma(I_k), \mu_k^u(I_k)) \in \mathcal{U}$ for every $k \in \mathcal{K}$ and for every I_k, $k \in \mathcal{K}$. Let J_π denote the cost (4) when the policy (6) is taken. The problem we tackle in this paper is to find a policy π to minimize J_π. Since this problem is in general hard, we shall proposed rollout suboptimal policies to be addressed in the next section.

To guarantee that $(\sigma_k, u_k) \in \mathcal{U}$ for a given $\sigma_k, k \in \mathcal{K}$, and for an optimal control policy for $u_k, k \in \mathcal{K}$ we assume the following.

Assumption 1: The matrices R_j, $j \in \mathcal{M}$ are assumed to be diagonal.

In fact, if Assumption (1) holds then the components of $u_k, k \in \mathcal{K}$ that do not influence (1) are set to zero by an optimal policy since they are weighted in (4).

Remark 2: The problem set-up considered here has some differences from the problem formulation in the companion paper [7], in which we consider the problem of designing only the scheduling decisions σ_k, $k \in \mathcal{K}$. In fact, in [7], we consider (i) a model with no disturbances; (ii) at a given time $k \in \mathcal{H}$ the components of the control input u_k that are not updated are set to the corresponding components of the previous control value u_{k-1} instead of being set to zero; (iii) infinite horizon problems ($k_F = \infty$). We can model hold operation of the control input with the following model

$$\begin{bmatrix} x_{k+1} \\ \hat{u}_k \end{bmatrix} = \begin{bmatrix} A & B(I - \Omega_k) \\ 0 & (I - \Omega_k) \end{bmatrix} \begin{bmatrix} x_k \\ \hat{u}_{k-1} \end{bmatrix} + \begin{bmatrix} B\Omega_k \\ \Omega_k \end{bmatrix} u_k$$

for some auxiliary state $\hat{u}_k, k \geq 0$. The same ideas in the present paper can then be applied to this model. However, considering analogous versions of the infinite horizon problem, such as average cost and discounted cost problems (cf. [8]) warrants further research.

III. Main Results

In this section, we start by showing how to compute the cost (4) 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 define a periodic protocol, we consider a set of h consecutive schedules σ_k denoted by

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

where $v_\ell \in \mathcal{M}, \ell \in \mathcal{H}$, $\mathcal{H} := \{0, \dots, h-1\}$, which are periodically repeated as explained next. If we let $\lfloor k \rfloor_h$ denote the remainder after division of k by h, we have

$$\sigma_k = \theta_k^\kappa, \quad k \in \mathcal{K}, \tag{8}$$

where

$$\theta_k^\kappa := v_{\lfloor k+\kappa \rfloor_h}, \, k \geq 0, \tag{9}$$

for some $\kappa \in \mathcal{H}$ that characterizes the initial condition of the periodic scheduling θ_k^κ.

Once the scheduling decisions $\sigma_\kappa, \kappa \in \mathcal{H}$ are fixed, we can obtain the optimal control input $u_\kappa, \kappa \in \mathcal{H}$ that minimizes problem (1), (4), as summarized in the next proposition (cf. [9]). Let

$$\underline{B}_j := B_j \Omega_j, \quad \underline{C}_j := \Gamma_j C_j$$

and

$$\begin{aligned} F_j(P) &:= A^\intercal P A + Q_j \\ &\quad - A^\intercal P \underline{B}_j (R_j + \underline{B}_j^\intercal P \underline{B}_j)^{-1} \underline{B}_j^\intercal P A \\ G_j(P) &:= -(R_j + \underline{B}_j^\intercal P \underline{B}_j)^{-1} \underline{B}_j^\intercal P A \\ H_j(N) &:= A N A^\intercal + \Phi_x \\ &\quad - A N \underline{C}_j^\intercal (\Phi_y + \underline{C}_j N \underline{C}_j^\intercal)^{-1} \underline{C}_j N A^\intercal \\ M_j(N) &:= -A N \underline{C}_j^\intercal (\Phi_y + \underline{C}_j N \underline{C}_j^\intercal)^{-1}. \end{aligned} \tag{10}$$

Proposition 3: The optimal policy for the control input $u_k, k \in \mathcal{K}$ when the scheduling $\sigma_k, k \in \mathcal{K}$ is described by (8) is given by

$$u_k = K_k \hat{x}_k, k \in \mathcal{K} \tag{11}$$

where the state estimates $\hat{x}_k, k \in \mathcal{K}$, are obtained from

$$\begin{aligned} \hat{x}_{k+1} &= (A + B\Omega_{\theta_k^\kappa} K_k + L_k \Gamma_{\theta_k^\kappa} C)\hat{x}_k - L_k y_k \\ u_k &= K_k \hat{x}_k, \qquad \hat{x}_0 = \bar{x}_0, \quad k \geq 0 \end{aligned} \tag{12}$$

and the gains K_k and L_k, $k \in \mathcal{K}$ can be determined by

$$\begin{aligned} P_{k_F} &= \underline{Q}_{k_F}, \quad P_k = F_{\theta_k^\kappa}(P_{k+1}), \\ K_k &= G_{\rho_k}(P_{k+1}), \quad k \in \mathcal{K} \end{aligned} \tag{13}$$

and

$$\begin{aligned} N_0 &= \Phi_0, \quad N_{k+1} = H_{\theta_k^\kappa}(N_k), \\ L_k &= M_{\theta_k^\kappa}(Q_k), \quad k \in \mathcal{K}. \end{aligned} \tag{14}$$

Moreover, the optimal cost (4) is given by $J_{\text{base},\kappa}(\hat{x}_0, \Phi_0)$, where

$$\begin{aligned} J_{\text{base},\kappa}(\bar{x}_0, \Phi_0) &= \hat{x}^\intercal P_0 \hat{x} + \text{tr}(P_0 \Phi_0) + \sum_{k=0}^{k_F-1} \text{tr}(P_{k+1}\Phi_x) + \\ &\sum_{k=0}^{k_F-1} \text{tr}(N_k A^\intercal P_{k+1} \underline{B}_{\theta_k^\kappa} (R + \underline{B}_{\theta_k^\kappa}^\intercal P_{k+1} \underline{B}_{\theta_k^\kappa})^{-1} \underline{B}_{\theta_k^\kappa}^\intercal P_{k+1} A). \end{aligned} \tag{15}$$

and $\bar{x}_0$, Φ_0 are the initial state estimate and associated error covariance matrix of (1).

□

Note that, under Assumption 1, the control policy (11), where K_k are described by (13), is in fact admissible.

B. Rollout Policy

We propose to choose at each iteration the scheduling decision and the control input 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, $k \in \mathcal{K}$, 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 (11), (12). The free scheduling variables are denoted by $\nu = (\nu_0, \dots, \nu_{H-1})$, i.e.,

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

and the fixed scheduling variables can be written as

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

Note that at time $\ell + H$ the base policy is assumed to start at an initial schedule v_κ, determined by κ. We consider that $\kappa \in \mathcal{H}$ is also a decision variable, and the decision set is denoted by $\mathcal{I} := \mathcal{M}^H \times \mathcal{H}$, 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,

$$H(\ell) := \min(H_c, k_F - 1 - \ell), \tag{18}$$

where $1 \leq H_c \leq k_F - 1$ is a constant[1] 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: At each time iteration ℓ take the scheduling decision σ_ℓ as

$$\sigma_\ell = w_0^\ell, \tag{19}$$

where w_0^ℓ is the first entry of the vector $w^\ell = (w_0^\ell, \dots, w_{H-1}^\ell)$ obtained from

$$(w^\ell, \kappa^\ell) = \text{argmin}_{(\nu,\kappa)\in\mathcal{I}} \hat{x}_\ell^\intercal P_{\nu,\kappa,\ell} \hat{x}_\ell + \delta(\nu, \kappa, \ell) + \beta(\nu, \kappa, \ell, N_\ell), \tag{20}$$

[1]In the companion paper [7] we consider the case $H = 0$ for which only the variable κ in (17) is a decision variable. For simplicity we do note consider this case in the present paper.

where

- $P_{\nu,\kappa,\ell} = \tilde{P}_0$ is computed at each iteration ℓ from
$$\tilde{P}_H = \bar{P}_H, \quad \tilde{P}_j = F_{\nu_j}(\tilde{P}_{j+1}), \quad 0 \leq j \leq H-1, \tag{21}$$
where $\bar{P}_H$ is obtained from
$$\bar{P}^{k_F} = \underline{Q}_{k_F}, \quad \bar{P}_j = F_{\theta^{\kappa}_{j-H}}(\bar{P}_{j+1}), \quad H \leq j \leq k_F - 1. \tag{22}$$
Moreover, the function δ is given by
$$\delta(\nu,\kappa,\ell) = \sum_{j=1}^{H-1} \text{tr}(\tilde{P}_j)\Phi_x) + \sum_{j=H}^{k_F} \text{tr}(\bar{P}_j \Phi_x).$$
- The N_ℓ and $\hat{x}_\ell$ are obtained recursively by at each iteration $0 \leq k < \ell$ updating the following iterations based on past scheduling decisions
$$N_0 = \Phi_0, \quad N_{k+1} = H_{\sigma_k}(N_k), \quad 0 \leq k < \ell, \tag{23}$$
and for $0 \leq k < \ell$,
$$\hat{x}_{k+1} = (A + B\Omega_{\sigma_k} K_k + L_k \Lambda_{\sigma_k} C)\hat{x}_k - L_k y_k, \tag{24}$$
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, $0 \leq k < \ell$.
- The function β is given by
$$\beta(\nu,\kappa,\ell,N_\ell) = \text{tr}(N_\ell \tilde{P}_0)$$
$$+\sum_{j=0}^{H-1} \text{tr}(\tilde{N}_j (A^\intercal \tilde{P}_{j+1} \underline{B}_{\nu_j} (R + \underline{B}^\intercal_{\nu_j} \tilde{P}_{j+1} \underline{B}_{\nu_j})^{-1} \underline{B}^\intercal_{\nu_j} \tilde{P}_{j+1} A)$$
$$+\sum_{j=H}^{k_F-1} \text{tr}(\bar{N}_j (A^\intercal \bar{P}_{j+1} \underline{B}_{\theta^{\kappa}_j} (R + \underline{B}^\intercal_{\theta^{\kappa}_{j-H}} \bar{P}_{j+1} \underline{B}_{\theta^{\kappa}_{j-H}})^{-1} \underline{B}^\intercal_{\theta^{\kappa}_{j-H}} \bar{P}_{j+1} A).$$
where $\bar{N}_k$ and $\tilde{N}_k$ are obtained at iteration ℓ from
$$\tilde{N}_{j+1} = H_{\nu_j}(\tilde{N}_j), 0 \leq j \leq H-1, \quad \tilde{N}_0 = N_\ell$$
$$\bar{N}_{j+1} = H_{\theta^{\kappa}_{j-H}}(\bar{N}_j), H \leq j \leq k_F - 1, \quad \bar{N}_H = \tilde{N}_H$$

Moreover, the control law u_ℓ is given by
$$u_\ell = K_\ell \hat{x}_\ell, \tag{25}$$
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). □

Again, note that, under Assumption 1, the control policy (25) is in fact admissible.

As stated next, this protocol does in fact correspond to the rollout algorithm described above. Moreover, for $H = 1$, we can establish that it outperforms the corresponding periodic base policy. Let $J_{\text{rollout}}(\bar{x}_0, \Phi_0)$ be the cost (4) when the scheduling sequence σ_k and the control law u_k for the system (1) are as described by Protocol 1, which depends on the initial state estimate $\bar{x}_0$ and associated error covariance matrix Φ_0 for (1).

Theorem 4: The rollout scheduling algorithm with the base policy described in Subsection III-A is determined by the Protocol 1. Moreover, in the case where $H = 1$, the following holds for every $\bar{x}_0 \in \mathbb{R}^n$, $\Phi_0 \in \mathbb{R}^{n \times n}$.
$$J_{\text{rollout}}(\bar{x}_0, \Phi_0) \leq \min_{\kappa \in \mathcal{H}} J_{\text{base},\kappa}(\bar{x}_0, \Phi_0) \tag{26}$$
□

We restrict $H = 1$ to obtain (26) since for this case one can apply directly analogous arguments to [8, Prop. 6.3.1, and p.338] to establish (26). Establishing if (26) holds for $H > 1$ warrants further research.

We provide next further important comments on Protocol 1 and Theorem 4, concerning: (i) the fact that Protocol 1 is a closed-loop policy for the scheduling decisions; (ii) real-time implementation.

1) Closed-loop policy for the scheduling decisions: The scheduling decision (20) depends in general on the state estimation $\hat{x}_\ell$ at time ℓ, on the error covariance matrix Φ_ℓ at time ℓ, and on the time ℓ. Since $\hat{x}_\ell$ is updated taking into account the measurements y^j_k (cf. (24)), which in turn are corrupted by noise, the scheduling sequence of the rollout algorithm cannot in general to determined a priori, i.e., it must be determined on-line, or in a closed-loop fashion. We shall see in the sequel that for the special case of the sensor scheduling problem, one can determine the scheduling decisions a priori. The rollout policy for the control law, given by (25), is a closed-loop policy, since it depends on $\hat{x}_\ell$. Note also that 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 decisions.

2) Real-time implementation: The Protocol 1 can be 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 (20) is available at time $t_{\ell-1}$ and therefore computations can be done between time $t_{\ell-1}$ and t_ℓ. In this way, computational delays can be avoided 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.

Since, for long horizons k_F, the computation requirements can be too demanding, one may need to consider less computationally demanding policies. Note that the scheduling decision (20) depends in general on the state estimation $\hat{x}_\ell$ at time ℓ, on the error covariance matrix Φ_ℓ at time ℓ, and on the time ℓ. Removing the dependency from ℓ may be obtained by considering a infinite horizon discounted problem formulation [8], since in this case rollout policies lead to stationary polices, and this is a topic for future work. Finding heuristic rules for scheduling in the space $(\hat{x}_\ell, \theta_\ell)$ that approximate the rule (20) is a topic for future research.

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 $\hat{x}_\ell$, and also Q_ℓ, at time t_ℓ. In fact, knowing the data sent over the network at times $0 \leq k < \ell$, nodes can locally iterate (24) and (23). Moreover, since the knowledge of $\hat{x}_\ell$, and Q_ℓ, at time ℓ summarizes the information a node 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 an arbitration field is available, which is used in the CAN-BUS messages, plays an essential role.

2) To sample or to control in CAN-BUS networks: Here, sensors are no longer assumed to be able to 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 matrices $P_{\nu,\kappa,\ell}$, obtained from (21) do not depend on ν and κ, due to $\Lambda_j = I_m$, for every $1 \leq j \leq n_y$, and therefore (20) does not depend on $\hat{x}_k$, i.e., the optimal schedules at time ell only depend on the error covariance matrix N_ℓ, which can be computed off-line 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 off-line.

IV. 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. We also addressed the relation of the work with the sensor scheduling problem.

There are several directions for future work, from which we mention: i) considering an infinite horizon discounted problem formulation in which case rollout policies are expected to lead to stationary policies, ii) considering less computationally demanding policies using heuristic rules guided by the rule (20), as explained in Subsection III-C.2; iii) extending Theorem 4 to lookahead horizons larger than one.

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," Mar. 2012, to be presented at 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*. Academic Press, 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.