

Lowering Traffic without Sacrificing Performance in Networked Control Systems

Pau Martí, Manel Velasco, José Yépez and Enric X. Martín
Automatic Control Department
Technical University of Catalonia
Pau Gargallo 5, 08028 Barcelona, Spain
{pau.marti,manel.velasco,jose.yepez,enric.xavier.martin}@upc.edu

Abstract

In Networked Control Systems (NCS), the amount of control data exchanged between sensors, controllers and actuators nodes highly depends on the control performance specifications given to each networked control loop. The periodic execution of each loop helps meeting the control specifications while imposing a static network traffic. This paper presents an alternative execution mechanism for each networked control loop that permits to dynamically lower the traffic while ensuring the same or better control performance than the achieved by the periodic case. Simulation results illustrate the theoretical analysis.

I. Introduction

In NCS, control loops are deployed in physically distributed sensors, controllers and actuators that exchange control data over a network. In particular, each networked closed loop execution, which involves sampling, control algorithm computation, and actuation, requires sending a sensor message from sensor to controller node, and a control message from controller to actuator node. The standard design and implementation approach to NCS consists in the periodic execution of control loops, which facilitates meeting control performance specifications at the expenses of making an static and often abusive use of the bandwidth [1].

This paper presents a novel execution approach for networked control loops that permits alleviating the bandwidth demand imposed by the periodic approach without scarifying control performance. Rather than using a time-triggered paradigm, the execution rule for each networked control loop is event-triggered. By using an optimal control setting, the novel execution rule triggers control loops' executions guaranteeing that the resulting traffic rate in the worst case scenario will be lower bounded by the rate of the periodic case.

In particular, for a single control loop, at each execution, the execution rule determines when the next execution will occur. In the worst case scenario, it will occur after the sampling period (used in the periodic approach) has elapsed. Otherwise, it will occur later on, thus lowering the traffic rate. In the first case, the performance delivered for that execution will be the same as the one delivered by the periodic approach. In the second case, even executing later than the time marked by the periodic case, a performance improvement is achieved. In summary, at each loop activation, the execution rule acting at the sensor node looks for the furthest next loop activation time that can be applied considering that the resulting performance does not decrease with respect to the performance given by a periodic controller.

Event-driven control has been proved to be a promising technique for lowering the computational demand of controllers [2]. And for NCS, event-triggered sampling has been shown to be much more efficient than time-triggered sampling schemes in terms of resource usage of the communication system [3]. Although recent research has applied event-triggering techniques to NCS (e.g., [4], [5], or [6]) none of these works has focused on deriving an execution rule that allows decreasing the network traffic while maintaining or improving control performance with respect to the one delivered by periodic optimal controllers. Complementary to this paper, it is interesting to stress that [5] or [7] presented approaches to NCS aimed at maximizing control performance rather than minimizing bandwidth consumption, as it is done in this paper. In [5] or [7] spare bandwidth is used for executing additional control updates apart from the periodic ones in such a way that control performance is increased. Hence, the resulting traffic in the network follows an aperiodic pattern, like the traffic originated by the approach presented in this paper.

Simulation results show that the approach is effective at lowering the network traffic without sacrificing performance. In addition, they show that the effectiveness of the presented approach is highly influenced by several key parameters. Finally, since the execution rule modifies

messaging timing constraints at run-time, schedulability issues are also discussed.

The rest of this paper is organized as follows. Section II formulates the problem to be solved. Section III develops the theory that solves the problem and discusses implementation issues. Section IV presents simulation results and Section V concludes the paper.

II. Problem formulation

The networked control system considered in this paper consists of $i = 1, \dots, n$ control loops, each one controlling a plant. Henceforth, the index i is dropped in the plant, controller, and cost whenever not required. Each plant is described by a continuous-time linear system

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) \end{aligned} \quad (1)$$

where $x \in \mathbb{R}^{n \times 1}$ is the plant state, $u \in \mathbb{R}$ is the control signal (input), $A \in \mathbb{R}^{n \times n}$ and $B \in \mathbb{R}^{n \times 1}$ are the system and input matrices respectively, and $C \in \mathbb{R}^{1 \times n}$ is the output matrix. Let

$$u(t) = u_k = Lx(t_k) = Lx_k \quad \forall t \in [t_k, t_{k+1}[\quad (2)$$

be the control updates given by a linear feedback controller L designed in the discrete-time domain using only samples of the state at discrete instants $t_0, t_1, \dots, t_k, \dots$. We refer to t_k as the controller activation times. Between two consecutive control updates, $u(t)$ is held constant. In periodic sampling we have $t_{k+1} = t_k + h$, where h is the period of the controller.

For each networked control loop, rather than applying periodic control, we are interested in achieving a non-periodic execution pattern based on

- enlarging at each controller execution the current sampling interval
- while providing the same or better control performance than the periodic case
- assuming that the controller gain L designed for the periodic scenario is kept constant in the new non-periodic execution approach,
- and considering that the forthcoming activations after the current one will be periodic.

Figure 1 illustrates the problem to be solved. The upper line exemplifies the periodic execution of a control loop, with $t_{k+1} - t_k = h$ where $k = 0, \dots, \infty$, whose control performance is evaluated using a cost function J evaluated from the current time ($t_0 = 0$) to infinity, and labeled as J_0^∞ . The lower line exemplifies the proposed approach. Let $\tau = t_1 - t_0$ denote the time interval from the current time t_0 to the next activation time t_1 . During the τ time interval, the current execution of the control loop is taking place. From t_1 to infinity, control loop executions are assumed to be periodic with the same period as before, that is, assuming that $t_{k+1} - t_k = h$ with $k = 1, \dots, \infty$. For this approach, J_0^τ denotes the cost function J evaluated during τ , and J_τ^∞ denotes the

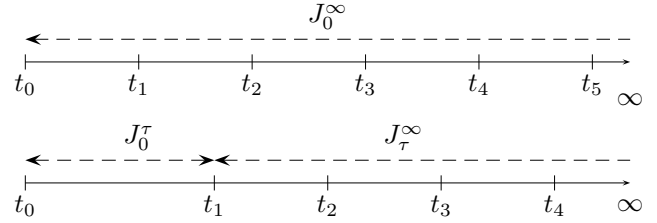


Fig. 1: Periodic execution pattern (upper line) versus non-periodic execution pattern (lower line)

remaining cost J evaluated from t_1 to infinity due to the assumed forthcoming periodic executions. To simplify, we use

$$J(\tau) = J_0^\tau + J_\tau^\infty \quad (3)$$

to denote the evaluation from the current time $t_0 = 0$ to infinity of the cost function J using the non-periodic execution pattern of the presented approach, illustrated in the lower line of Figure 1.

For a given control loop, the problem to be solved at each control loop execution is to find the longest τ value such that

$$J(\tau) \leq J_0^\infty \quad \text{with} \quad \tau \geq h. \quad (4)$$

III. Execution rule

The execution rule solving the problem formulated in the previous section is described next. First, the theoretical analysis is presented, and implementation issues are then discussed.

A. Problem reformulation

Finding the longest τ that satisfies (4) can be transformed into an optimization problem that simplifies the search for τ . The optimization problem can be formulated as

$$\text{maximize} \quad \tau \quad (5)$$

$$\text{subject to} \quad \frac{dJ(\tau)}{d\tau} = 0 \quad (6)$$

$$J_0^\infty - J(\tau) \geq 0 \quad (7)$$

$$\tau > h \quad (8)$$

$$x(\tau) = \Phi(\tau)x_0 + \Gamma(\tau)u_0 \quad (9)$$

where

$$u_0 = Lx_0 \quad (10)$$

and

$$\Phi(\tau) = e^{A\tau}, \quad \Gamma(\tau) = \int_0^\tau e^{A(\tau-s)} ds B, \quad \tau \in [t_k, t_{k+1}). \quad (11)$$

The objective function (5) indicates that at each closed-loop execution we should find the longest τ whose cost $J(\tau)$ is equal or lower than the cost J_0^∞ given by the periodic execution approach, constraint specified by (7). Noting that the objective function (5) is continuous, to simplify the search, constraint (6) reduces the constraint

set to a set of isolated points (critical points) that may contain a maximum and a minimum for J . This helps evaluating faster if there is any τ satisfying (7), and ensures that the problem is solvable. Note that the feasible domain given by restrictions (6)-(9) may be an empty set. In this case, we specify $\tau = h$. Finally, constraint (8) ensures that the next activation time will occur later than the time specified by h , and constraint (9) further shapes the search for τ because it is restricted to the specific closed-loop dynamics into consideration.

B. An explicit solution

This section presents a solution to the optimization problem (5)-(9) for the case where the cost function J and controller gain L are the standard ones used in an optimal control setting.

Hence, for each control loop (1), control performance is measured by an infinite-horizon continuous-time quadratic cost function

$$J = \int_0^\infty [x^T(t)Q_c x(t) + u^T(t)R_c u(t) + 2x^T(t)N_c u(t)] dt \quad (12)$$

where the weighting matrices Q_c and R_c are symmetric positive semi-definite matrices [8].

The evaluation of cost (12) varies if a periodic or an aperiodic execution approach is adopted. For periodic execution of a control loop, if the gain L in (2) is designed using linear quadratic (LQR) control for a given sampling period h , the evaluation of cost (12) is [8]

$$J_0^\infty = x_0^T S(h) x_0 \quad (13)$$

where $S(h)$ is the solution to the algebraic Riccati equation, and x_0 is a given initial state. For aperiodic execution, and using the same gain L as before, the evaluation of cost (12) can be specified in terms of J_0^τ and J_τ^∞ as in (3), being

$$J_0^\tau = \int_0^\tau [x^T(t)Q_c x(t) + u^T(t)R_c u(t) + 2x^T(t)N_c u(t)] dt \quad (14)$$

and

$$J_\tau^\infty = x^T(\tau)S(h)x(\tau). \quad (15)$$

Looking at the optimization problem (5)-(9), the first step is to find the critical points of $J(\tau)$, that is, to solve constraint (6). This requires expanding the derivative of $J(\tau)$ as

$$\begin{aligned} \frac{dJ(\tau)}{d\tau} &= \frac{d}{d\tau} \left[\int_0^\tau [x^T(t)Q_c x(t) + u^T(t)R_c u(t) + 2x^T(t)N_c u(t)] dt + x^T(\tau)Sx(\tau) \right] \\ &= x^T(\tau)Q_c x(\tau) + u^T(\tau)R_c u(\tau) \\ &\quad + 2x^T(\tau)N_c u(\tau) + \frac{d}{d\tau} [x^T(\tau)Sx(\tau)]. \quad (16) \end{aligned}$$

Since the control signal is held constant over consecutive control updates, in (16) the input $u(\tau)$ can be replaced by u_0 . The last summand of (16) will require computing the

variation of $x(\tau)$ with respect to τ . By considering (9) and the Leibniz integration rule, this variation is

$$\begin{aligned} \frac{d}{d\tau} x(\tau) &= \frac{d}{d\tau} [\Phi(\tau)x_0 + \Gamma(\tau)u_0] \\ &= \frac{d}{d\tau} \left[e^{A\tau}x_0 + \int_0^\tau e^{As}B ds u_0 \right] \\ &= Ae^{A\tau}x_0 + Bu_0 + A \int_0^\tau e^{As}B ds u_0 \\ &= A\Phi(\tau)x_0 + Bu(\tau) + A\Gamma(\tau)u_0 \\ &= A[\Phi(\tau)x_0 + \Gamma(\tau)u_0] + Bu(\tau) \\ &= Ax(\tau) + Bu(\tau). \quad (17) \end{aligned}$$

Noting again that in the previous expression $u(\tau) = u_0$, from (17), we can expand the last summand of (16) as

$$\begin{aligned} \frac{d}{d\tau} (x^T(\tau)Sx(\tau)) &= \frac{d}{d\tau} x^T(\tau)Sx(\tau) + x^T(\tau)S \frac{d}{d\tau} x(\tau) \\ &= [Ax(\tau) + Bu_0]^T Sx(\tau) \\ &\quad + x^T(\tau)S [Ax(\tau) + Bu_0] \\ &= x^T(\tau)A^T Sx(\tau) + u_0^T B^T Sx(\tau) \\ &\quad + x^T(\tau)SAx(\tau) + x^T(\tau)SBu_0 \\ &= x^T(\tau) [A^T S + SA] x(\tau) \\ &\quad + 2x^T(\tau)SBu_0. \quad (18) \end{aligned}$$

Taking into account that S is symmetric, substituting (18) into (16) we obtain

$$\begin{aligned} \frac{dJ(\tau)}{d\tau} &= x^T(\tau)Q_c x(\tau) + u_0^T R_c u_0 + 2x^T(\tau)N_c u_0 \\ &\quad + 2x^T(\tau)SAx(\tau) + 2x^T(\tau)SBu_0 \\ &= x^T(\tau)\bar{Q}x(\tau) + u_0^T \bar{R}u_0 + 2x^T(\tau)\bar{N}u_0 \\ &= \begin{bmatrix} x(\tau) & u_0 \end{bmatrix} \begin{bmatrix} \bar{Q} & \bar{N} \\ \bar{N}^T & \bar{R} \end{bmatrix} \begin{bmatrix} x(\tau) \\ u_0 \end{bmatrix} \quad (19) \end{aligned}$$

where

$$\bar{Q} = [2SA + Q_c], \quad \bar{N} = [N_c + SB], \quad \bar{R} = R_c.$$

Finally, making (19) equal to zero, we obtain the equation

$$\begin{bmatrix} x(\tau) & u_0 \end{bmatrix} \begin{bmatrix} \bar{Q} & \bar{N} \\ \bar{N}^T & \bar{R} \end{bmatrix} \begin{bmatrix} x(\tau) \\ u_0 \end{bmatrix} = 0 \quad (20)$$

whose solutions are the critical points given by constraint (6).

To solve (20), $x(\tau)$ must be made explicitly depending on τ , and then solve for τ . A general approach can be to use a n -order approximation of $x(\tau)$ if an exact expression for $x(\tau)$ does not exist. First of all, $\Phi(\tau)$ and $\Gamma(\tau)$ in (9) can be written in terms of $\Psi(\tau)$ as [8]

$$\Phi(\tau) = e^{A\tau} = I + A\Psi(\tau) \quad (21)$$

$$\Gamma(\tau) = \int_0^\tau e^{As}B ds = \Psi(\tau)B \quad (22)$$

where $\Psi(\tau)$ is

$$\Psi(\tau) = \int_0^\tau e^{As} ds = \sum_{i=0}^{\infty} \frac{A^i \tau^{i+1}}{(i+1)!}. \quad (23)$$

Considering (10), and substituting (21) and (22) into constraint (9) we obtain

$$\begin{aligned} x(\tau) &= \Phi(\tau)x_0 + \Gamma(\tau)u_0 \\ &= [I + A\Psi(\tau)]x_0 + \Psi(\tau)Bu_0 \\ &= [I + A\Psi(\tau)]x_0 + \Psi(\tau)BLx_0 \\ &= x_0 + A\Psi(\tau)x_0 + \Psi(\tau)BLx_0. \end{aligned} \quad (24)$$

Commuting matrices $\Psi(\tau)$ and A , (24) can be written as

$$x(\tau) = x_0 + \Psi(\tau)[A + BL]x_0. \quad (25)$$

Finally, we approximate $\Psi(\tau)$ by a Taylor series of n -order around x_0 in (25), thus obtaining

$$\begin{aligned} x(\tau) &= x_0 + (A + BL)x_0\tau + \frac{A[A + BL]}{2!}x_0\tau^2 \\ &\quad + \frac{A^2[A + BL]}{3!}x_0\tau^3 + \dots \end{aligned} \quad (26)$$

Using (21), (22), (23) and a convenient approximation given by (26), eq. (20) can be solved for τ . Note that using a n -order approximation will result in a $2n$ -order equation. Among all τ values obtained after solving (20), constraint (7) must be evaluated to chose only those values for τ fulfilling constraint (8) whose application will not increase the cost with respect to the cost obtained by the periodic scenario. And finally, from this subset of values for τ , we have to chose the biggest one.

C. Recipe

The explicit solution presented in the previous section can be summarized by the following recipe:

- 1) Given an initial state x_0 , compute $x(\tau)$ as in (26) if an exact solution does not exists, and compute u_0 as in (10) considering that L is designed for a given sampling period h
- 2) Compute (20) and solve for τ , and keep the real positives ones longer than h .
- 3) Compute the cost for each τ value, as in (3) considering (14) and (15), and compute the cost for the periodic controller, as in (13). Then keep those τ values that satisfy constraint (7), and choose the biggest one
- 4) If at any step the set of values for τ is empty, take $\tau = h$

D. Numerical example

In order to illustrate the receipt, a numerical example is introduced using the simulation setup presented later in sub-section IV-A. The plant and cost function matrices are given by (27) and (28), and for a sampling period of $h = 0.8$ s, the LQR gain and $S(0.8)$ are given by (29) and (30). With this setting, the receipt works as follows:

- 1) Let us consider $x_0 = [0.20421 \quad -0.22556]^T$ as the initial condition. Considering an approximation

of 7^{th} order¹, applying (26) we obtain

$$x(\tau) = \begin{bmatrix} x_1(\tau) \\ x_2(\tau) \end{bmatrix}$$

with

$$\begin{aligned} x_1(\tau) &= 0.00004\tau^7 - 0.00018\tau^6 - 0.00187\tau^5 \\ &\quad + 0.00548\tau^4 + 0.03759\tau^3 - 0.06586\tau^2 \\ &\quad - 0.22556\tau + 0.20421 \\ x_2(\tau) &= 0.00002\tau^7 + 0.00031\tau^6 - 0.00109\tau^5 \\ &\quad - 0.00939\tau^4 + 0.02195\tau^3 + 0.11278\tau^2 \\ &\quad - 0.13172\tau - 0.22556. \end{aligned}$$

The control input (10) is

$$u_0 = Lx_0 = 0.07249.$$

- 2) By knowing that the matrix in (20) is

$$\begin{bmatrix} \bar{Q} & \bar{N} \\ \bar{N}^T & \bar{R} \end{bmatrix} = \begin{bmatrix} -0.0126 & 10.6092 & 1.5063 \\ -9.6750 & 2.0126 & 6.3375 \\ 1.5063 & 6.3375 & 20 \end{bmatrix}$$

equation (20) is

$$\begin{aligned} 0 &= 0.0000000024\tau^{14} + 0.0000000417\tau^{13} \\ &\quad - 0.0000000615\tau^{12} - 0.0000030094\tau^{11} \\ &\quad - 0.0000002042\tau^{10} + 0.0001028241\tau^9 \\ &\quad + 0.0000356694\tau^8 - 0.0018905926\tau^7 \\ &\quad - 0.0002815773\tau^6 + 0.0187336318\tau^5 \\ &\quad - 0.0039515483\tau^4 - 0.0713163403\tau^3 \\ &\quad + 0.0846151878\tau^2 - 0.0271364018\tau \\ &\quad + 0.0012904926 \end{aligned}$$

whose solutions are

$$\tau_i = \begin{bmatrix} -13.2076 \\ -5.7512 \\ -4.6907 + 1.1422i \\ -4.6907 - 1.1422i \\ -4.2045 + 1.2438i \\ -4.2045 - 1.2438i \\ 4.2134 + 1.6858i \\ 4.2134 - 1.6858i \\ 4.1143 + 1.5487i \\ 4.1143 - 1.5487i \\ 1.4312 \\ 1.0252 \\ 0.4639 \\ 0.0572 \end{bmatrix}.$$

From the set of τ_i values, the real positives values longer than $h = 0.8$ s are

$$\tau'_i = \begin{bmatrix} 1.4312 \\ 1.0252 \end{bmatrix}.$$

¹A 7^{th} order approximation is chosen because it gives an accuracy up to 4 decimals digits.

3) The cost for each τ value, as in (3), is

$$\tau'_i = \begin{bmatrix} 1.4312 \\ 1.0252 \end{bmatrix} \quad J(\tau_i) = \begin{bmatrix} 0.4249 \\ 0.4254 \end{bmatrix}$$

where for example

$$\begin{aligned} J(1.4312) &= J_0^{1.4312} + J_{1.4312}^\infty \\ &= 0.1792 + 0.2457 \\ &= 0.4249. \end{aligned}$$

And the cost for the periodic case (13) is $J_0^\infty = 0.4253$. Hence, from the set of remaining τ_i values, the only value that satisfies constraint (7) is the first one, thus we select $\tau = 1.4312$.

4) This step does not apply

Note that the selected value, $\tau = 1.4312$ s provides a lower cost than the periodic case while being almost twice than the period of the periodic case, $h = 0.8$ s.

E. Overhead analysis

At each closed loop execution, the recipe given in Sub-section III-C must be executed to determine τ , that is, to determine if the next execution will take place later than the time marked by the assumed periodic execution characterized by h . However, computing the full recipe may be expensive.

In particular, the application of the execution rule requires solving equation (20) (step 2 of the given recipe) at each closed loop execution. Note that in (20), both $x(\tau)$ given by (26) and u_0 given (10) depend on x_0 . Therefore, the only parameter of (20) that changes at each execution is x_0 , that is, the sampled state. However, as indicated in [9], for execution rules having the structure specified in (20), the solution to (20) remain the same for states lying in the same direction, that is, having the same orientation. Using the previous example, it is easy to check that the resulting τ after applying the recipe for a given x_0 is the same if we apply the recipe to the scaled state $c x_0$, $c \in \mathbb{R}$. This property permits an offline approach when applying the recipe at each control loop execution.

In particular, searching for τ requires only working with the states belonging to the unit hypersphere rather than the full state space because states lying in the same ray will provide the same value for τ . Even more, due to the linear systems symmetry, we only have to explore half hypersphere [9].

Figure 2 illustrates the application of the recipe for the states belonging to the unit sphere, thus moving the orientation within the interval $[0, 2\pi]$, for the given set-up used in the numerical example presented in sub-section III-D. It plots τ as a function of the state orientation. As it can be seen, the different τ values range from 0.8s to 1.65s approximately. In particular, for the initial condition given in the previous numerical example, $x_0 = [0.20421 \quad -0.22556]^T$, it can be seen from the figure that being its orientation 2.30rad

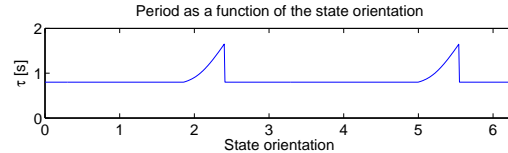


Fig. 2: Values for τ as a function of the state orientation

(approximately), the τ value is about 1.4s, as we already computed previously.

Since the computation of the τ values as shown in Figure 2 can be performed offline, the overhead introduced by executing the recipe at each loop operation (and in particular at each sampling in the sensor node) can be highly reduced. For each networked closed loop system, in the sensor node, the information drawn in Figure 2 can be stored in memory for run-time table look up, where for each state orientation, the corresponding τ is specified. The size of the table will depend on the granularity of states orientations specified in the interval $[0, 2\pi]$. In fact, the linear systems symmetry property permits to shrink the interval to $[0, \pi]$. To further reduce the table size, only those orientations that give a τ value longer than h in the shrunk interval need to be stored.

Using the look-up table, the recipe given in sub-section III-C simplifies to:

- 1) Given an initial state x_0 , compute its orientation.
- 2) Given the orientation, search for the τ in the table.

Hence, its implementation is easy and the computational overhead is negligible. The memory storage overhead depends on the granularity chosen for the table, which at the end will affect the accuracy of the selected τ .

E. Implementation issues

Assuming a simple model of execution for each networked control loop, Figure 3 illustrates the basic codes to be executed in sensor/controller/actuator nodes if the periodic approach is considered. The main property is that sampling in the sensor node is periodically time-triggered with a periodicity given by h , while the rest of operations at the other nodes are triggered by incoming sensor-to-controller and controller-to-actuator messages. For example, the periodic execution of the code of the sensor node can be achieved by an interrupt.

The implementation of the presented approach modifies the sensor node pseudo-code as illustrated in Figure 4. In this case, the sensor node is triggered according to τ , which may vary at each sensor execution. And the execution of `compute_tau`, that gives the solution to the optimization problem (5)-(9), is the recipe developed in sub-section III-C if sufficient computational power is available at the sensor node, or the recipe developed in sub-section III-E based on run-time table look-up if minimization of computational overhead is required.

It is important to stress that the networked control system model presented in section II and the implementation details given in this section did not account for

Algorithm 1: Sensor node (triggered by interrupt every h)

```

1 begin
2    $x_k := \text{sample}()$ 
3    $\text{send\_sensor\_message}(x_k)$ 
4 end

```

Algorithm 2: Controller node (triggered on message reception)

```

1 begin
2    $x_k := \text{receive\_sensor\_message}()$ 
3    $u_k := \text{compute\_control\_signal}(L, x_k)$ 
4    $\text{send\_controller\_message}(u_k)$ 
5 end

```

Algorithm 3: Actuator node (triggered on message reception)

```

1 begin
2    $u_k := \text{receive\_controller\_message}()$ 
3    $\text{apply\_control\_signal}(u_k)$ 
4 end

```

Fig. 3: Nodes pseudo-codes for periodic execution of a networked control loop

Algorithm 4: Sensor node (triggered by interrupt every τ)

```

1 begin
2    $x_k := \text{sample}()$ 
3    $\tau := \text{compute\_}\tau(x_k, u_k)$ 
4    $\text{send\_sensor\_message}(x_k)$ 
5    $\text{reset\_interrupt}(\tau)$ 
6 end

```

Fig. 4: Sensor pseudo-code with the novel execution rule

delays. The omission is done on purpose for the sake of simplicity. Regarding the theoretical part, accounting for delays would require adding a time delay to each plant model (1) and designing the control gain L in (2) using a suitable technique for systems with time delays (see for example [10] for a recent control approach to NCS, and the references therein for a state of the art on control techniques for NCS). Still in the theoretical part, and in terms of the cost evaluation, considering systems with time delays would also require treating the discretization of the cost function (12) in terms of delays, as it is done for example in [11]. In terms of implementation, time delays could be treated as in the approach presented in [12]. Then the pseudo-code for the sensor node shown in Figure 4 together with the pseudo-code for the controller and actuator node shown in Figure 4 should be modified to account for the cited technique.

G. Scheduling issues

In terms of messages timing constraints, each execution rule acting at each sensor node changes the period of the sampling task (recall Figure 4). Changing this period means that sensor-to-controller messages have a varying periodicity as well as controller-to-actuator messages.

Hence, the application of the execution rule would require analysis the effect of such period variations in the message set schedulability for the given communication protocol. Or alternatively, noting that the period variations are lower bounded by the sampling period h used in the controller design, the concept of sustainability in real-time scheduling introduced in [13] could also be applied for the message set schedulability analysis.

IV. Simulations

A. Simulation settings

The networked control system consists on three networked control loops, where sampling, control algorithm computation, and actuation, are performed for each loop in isolated nodes that exchange data through a digital serial network. To simplify the performance analysis, the simulation parameters are selected in such a way that the time delays introduced by the network are negligible with respect to the dynamics of the plant. In this way, the effect of transmission time delays, queuing time delays, and other delays originated by the medium access control protocol are removed and the simulation results only reflect the direct benefits of the presented approach in terms of traffic reduction.

The LTI plant for each networked control is an oscillator system given by [8]

$$\dot{x} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u. \quad (27)$$

The cost function (12) to be minimized for each control loop is characterized by

$$Q_c = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}, \quad N_c = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \quad R_c = 20. \quad (28)$$

Each networked loop is characterized by a given sampling period that is used to compute the LQR controller. For example, for a sampling period $h = 0.8s$, the optimal LQR gain is

$$L = [0.0512 \quad -0.2751] \quad (29)$$

and $S(0.8)$ used in the cost computation given by (13) and (15) is

$$S = \begin{bmatrix} 4.7916 & 0.5023 \\ 0.5023 & 5.3258 \end{bmatrix} \quad (30)$$

Finally, each simulation run of 40s assumes a different initial condition for each plant.

Future work will introduce tighter simulation settings to study the effect of time delays in the cost functions that are then used for selecting the appropriated τ value

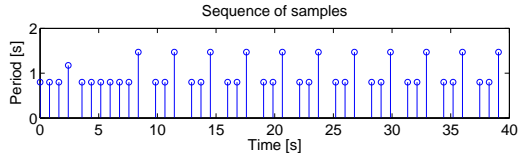


Fig. 5: Sampling intervals

(recall step 3 of the recipe given in sub-section III-C or the computation of the look-up table explained in sub-section III-E).

B. Main result

Figure 5 illustrates the main result achieved after applying the presented technique. The x -axis is simulation time, and the y -axis is the sampling interval (τ) also in seconds for the given networked control loop whose period at the design time was specified to be $h = 0.8$. Each sampling activation time at the sensor node is represented by a vertical line, whose height indicates the next sampling release time.

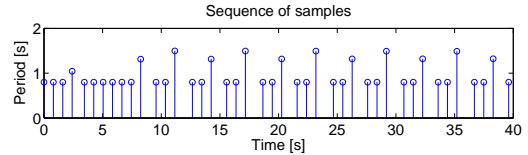
It can be seen in the figure that after 10 samples separated by a period of 0.8s (except the fourth one, which is a bit above 1s), the remaining sampling intervals adopt a pseudo periodic pattern with a repeating sequence of one sampling interval of 1.4s followed by two sampling intervals of 0.8s. In the long term run, this pattern is able to reduce the network traffic by 20%. And this reduction is achieved without decreasing the control performance compared to the one achieved by a pure periodic execution with $h = 0.8$ s. That is, comparing the cost achieved by the aperiodic execution run with respect to the cost achieved by the periodic execution run, the performance is very similar. In fact, the performance plot is not shown because the difference between achieved costs can not be distinguished although the aperiodic execution run gives a slightly better performance (lower cost) than the periodic run.

Henceforth, control performance numbers are not further discussed because the presented approach focuses on traffic reduction. Future work will focus on both performance metrics at the same time: traffic reduction and control performance improvement (cost reduction). This can be achieved by reformulating the optimization problem (5)-(9) in terms of maximizing τ and minimizing $J(\tau)$. Note that the cost also depends on the control signal u . Therefore, this dependence can be made explicit, $J(\tau, u)$, and then the theoretical analysis should look for the critical points of $J(\tau, u)$, that is, deriving the cost with respect to τ as it is done in (19) as well as with respect to u .

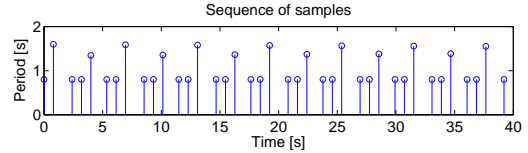
C. Parameters vs. performance

This section analyzes the benefits of the presented approach with respect to several parameters that influence the amount of traffic that can be reduced.

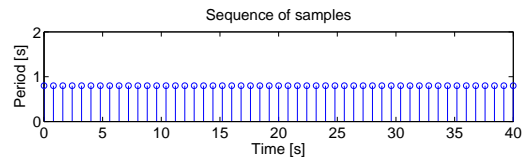
First of all, it is of interest to evaluate the perfor-



(a) $x_0 = (0.92 \ 0.38)^T$



(b) $x_0 = (1 \ 0)^T$



(c) $x_0 = (0 \ 0)^T$

Fig. 6: Sampling intervals as a function of the initial condition.

mance with respect to several initial conditions. The sampling intervals shown in Figure 5 were obtained with an execution run where the plant initial state was $x_0 = [0 \ 1]^T$. Sub-figures 6a, 6b and 6c show the sampling intervals when the initial condition for the plant is $x_0 = [0.92 \ 0.38]^T$, $x_0 = [1 \ 0]^T$ and $x_0 = [0 \ 0]^T$, respectively. Looking at Sub-figures 6a and 6b, although both sub-figures show a similar execution pattern in terms of sampling intervals than Figure 5, some differences can be identified. In Sub-figure 6a, the τ values exhibit also a repeating sequence after 10 sampling intervals, alternating two sampling intervals of 0.8s with a longer sampling interval. However, the longer sampling interval varies, alternating between 1.57s (almost twice the sampling period) and 1.37s. And looking at Sub-figure 6b, the execution pattern is similar to Sub-figure 6a but it starts from the very beginning of the simulation run. However, Sub-figure 6c shows a very different pattern. In fact it shows a pure periodic pattern. In this case, during the entire execution, all the states visited by the system trajectory have an orientation that gives always $\tau = 0.8$ (recall Figure 2). Hence, for this particular case, the presented execution rule is not able to enlarge the sampling period at any control loop execution.

The execution rule is characterized by the particular cost function (12) that is considered. In the simulation settings, we gave specific values to Q_c , N_c and R_c . Recovering $x_0 = [0 \ 1]^T$ as a initial condition for the plant, and remembering that we had $R_c = 20$ for Figure 5, Sub-figures 7a and 7b plot the sampling intervals for $R_c = 1$ and $R_c = 10$, respectively. Clearly, the type of sampling intervals also vary. For Sub-figure 7a only a few

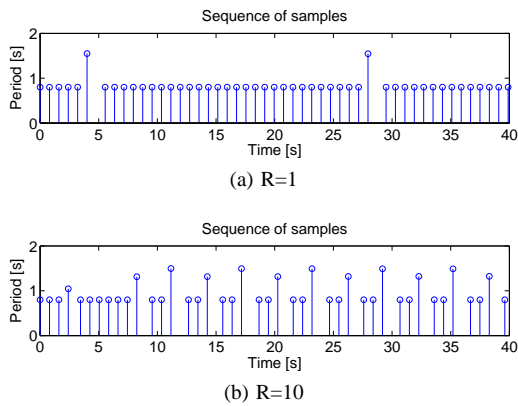


Fig. 7: Sampling intervals as a function of the cost function settings.

sampling intervals are enlarged while in Sub-figure 7b the same amount of sampling intervals than the original case (Figure 5) are longer than 0.8s.

The simulation results ask for a wider theoretical analysis able to relate parameters of the execution rule to the traffic reduction that can be achieved, which is left for future work.

V. Conclusion

This paper has presented an execution rule for networked control systems that produces aperiodic executions. It is effective at lowering the traffic generated in the network by each control loop while guaranteeing that control performance is not decreased compared to the case of periodic executions. The theoretical analysis indicates that the application of the execution rule may be expensive in terms of computational overhead. But a careful implementation analysis reveals an alternative strategy that highly reduces the overhead. Simulation results confirm that important reductions in the traffic can be achieved. They indicate that further work is required to characterize the types of traffic reduction vs. execution rule parameters.

Acknowledgments

This work was partially supported by EVENTS CI-CYT DPI2010-18601, and by ArtistDesign NoE IST-2008-214373.

References

- [1] D. Hristu-Varsakelis and W. S. Levine, *Handbook of Networked and Embedded Control Systems*, Birkhäuser Boston, June, 2008.
- [2] M. Velasco, P. Martí and E. Bini, "Control-driven Tasks: Modeling and Analysis," *29th IEEE Real-Time Systems Symposium*, Barcelona, Spain, December, 2008.
- [3] A. Cervin and T. Henningson, "Scheduling of event-triggered controllers on a shared network," *47th IEEE Conference on Decision and Control*, Cancun, Mexico, Dec. 2008.

- [4] A. Molin, S. Hirche, "Distributed event-triggered scheduling of wireless networked control systems," *Symposium on Recent Trends in Networked Systems and Cooperative Control*, Stuttgart, Germany, September, 2009.
- [5] A. Anta and P. Tabuada, "On the benefits of relaxing the periodicity assumption for networked control systems over CAN," *Real Time Systems Symposium*, December 2009.
- [6] X. Wang and M.D. Lemmon, "Event-triggering in distributed networked control systems," *IEEE Transactions on Automatic Control*, vol. 56, no. 3, pp 586-601, March 2011.
- [7] P. Martí, A. Camacho, M. Velasco, M. Ben Gaid "Run-Time Allocation of Optional Control Jobs to a Set of CAN-based Networked Control Systems," *IEEE Transactions on Industrial Informatics*, vol.6, no. 4, November 2010.
- [8] K.J. Åström and B. Wittenmark, *Computer-Controlled Systems*, 3rd Ed., Prentice-Hall, 1997.
- [9] M. Velasco, P. Martí and E. Bini, "Equilibrium sampling interval sequences for event-driven controllers," *European Control Conference 2009*, Budapest, Hungary, August 2009.
- [10] D. Nešić and D. Liberzon, "A unified framework for design and analysis of networked and quantized control systems," *IEEE Transactions on Automatic Control*, Vol. 54, No. 4, pp. 732-747, April 2009.
- [11] A. Cervin, M. Velasco, P. Martí and A. Camacho, "Optimal on-line sampling period assignment: theory and experiments," *IEEE Transactions on Control Systems Technology*, accepted for publications, 2011.
- [12] C. Lozoya, P. Martí, M. Velasco and J.M. Fuertes, "Analysis and design of networked control loops with synchronization at the actuation instants," *In 34th Annual Conference of the IEEE Industrial Electronics Society*, November, 2008.
- [13] A. Burns and S. Baruah, "Sustainability in real-time scheduling," *Journal of Computing Science and Engineering*, vol. 2 n.1, pp 74-97. 2008.