A Hybrid Scheduling Protocol to Improve Quality of Service in Networked Control Systems

Ahmed Elmahdi*, Ahmad F Taha*, Stefen Hui†, and Stanislaw H. Żak*

Abstract— **In many networked control systems (NCSs) only one node is allowed to use the shared medium at any given time. This network constraint can adversely affect the performance of the system and its stability. There are two types of network schedulers, static and dynamic. Static schedulers, such as the token ring protocol, have problems handling large-scale systems. On the other hand, dynamic schedulers, such as try-once-discard (TOD) cannot guarantee that every node receives the same quality of service. In this paper, a hybrid scheduler, which is a combination of dynamic and static protocols, is proposed. This scheduler improves the medium access strategy in large-scale control systems. We refer to this scheduler as the traffic-division arbitration (TDA) protocol. The network-induced delay error bound and the system stability of the NCS using the proposed scheduler are investigated. Simulations illustrate the performance of the proposed scheduler and its difference from TOD. We use two different decision functions to prioritize the scheduling criteria of the protocol.**

I. INTRODUCTION

A networked control system (NCS) is a control system that closes its control loops through a network, which can be the Internet or any network. An essential feature of an NCS is that signals such as reference input or plant output are exchanged between control system components such as actuators, sensors, or a controller through a network. This requires a scheduling protocol for the NCS to control traffic flow. In this paper, we propose a scheduling protocol for NCSs. In the next subsection, we discuss some NCS applications and the importance of scheduling protocols in the NCS operation.

A. Remarks on Applications of Networked Control Systems

Control systems are characterized by the three main components: sensors to measure input/output signals, controllers to provide commands to the actuators, and actuators to execute the controllers' commands. A control system can be modeled by a transfer function or a state-space model. In many applications, controllers and plants are geographically separated which requires adding a real time network to the control system. A defining attribute of the NCS is that feedback and controlled signals are exchanged among the system's components in the form of information packages through a network [1]. In Figure 1 we illustrate a typical NCS setup.

*School of Electrical and Computer Engineering, Purdue University, West Lafayette, Indiana 47907 (aelmahdi,tahaa,zak)@purdue.edu

†Department of Mathematical Sciences, San Diego State University, San Diego, California 92182 hui@math.sdsu.edu

Many modern control systems are inherently NCSs. The advantages of connecting the system components via network compared with traditional point-to-point control systems are modularity and flexibility of the system design, the simplicity of implementation, such as reduced system wiring and configuration tools and ease of diagnosing and maintaining the system [4].

NCSs applications are found in manufacturing plants, automobiles, air conditioning/cooling systems, elevators, building automation, medical equipment and devices, remote surgery, mobile sensor networks, robotics, and many other industrial applications. New cars have built-in NCSs. According to Walsh and Ye [2], [5], a modern car can have in fact several controller area networks (CANs): high speed CAN in front of the firewall for the engine, transmission, and traction control and low speed CAN for locks, windows, and other devices. Integrating computer and communication networks with control systems having different operations and functionality is a new trend in the current industrial applications [3].

The addition of a network to the system will increase its complexity. In particular, the Quality of Service (QoS) of the network which includes the network's fairness, increases the network complexity. Important issues that an NCS designer must address are: network's fairness, stability, delays, and error analysis. In this paper, we propose a scheduling protocol that addresses some of the challenges mentioned above, such as QoS and network-induced delays.

B. NCS Limitations

Although an NCS can improve system reliability, reduce weight, space, power and wiring requirements, there are constraints that somewhat limits its applications. Multiple-packet transmission, data packet drop-outs and finite bandwidth, are all problems that have to be addressed when an NCS is used. These problems can cause signal delay and distortion, affect on the stability and fairness of the network [6].

The distribution and characteristics of the network-induced delay and the signal distortion in an NCS are determined by its medium access control (MAC) protocol. The computation-time delay of a controller computer, the time taken to execute programs that implement control algorithms at the controller's nodes or processes data at sensors or actuator nodes, is another effective source of time delay [4], [11].

The insertion of a communication network in the feedback control loop makes the analysis and design of an NCS more complex. Conventional control theories with many ideal assumptions, such as synchronized control and non-delayed

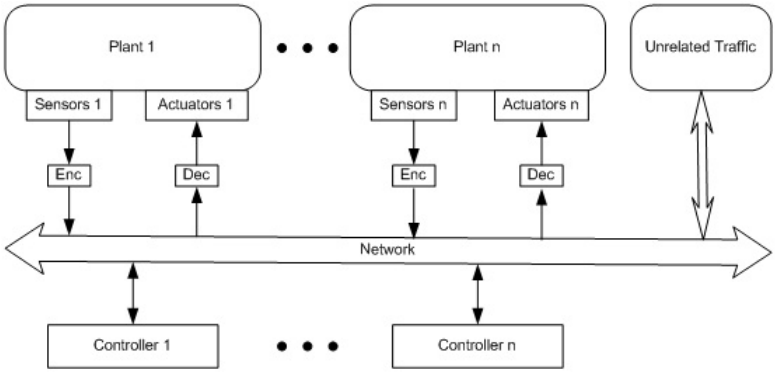


Fig. 1. A typical NCS setup.

sensing and actuation, must be re-evaluated before they can be applied. Basically, the primary objective of NCS analysis and design is to develop a system that efficiently uses the finite bus capacity while maintaining reasonable closed-loop control system performance. Furthermore, because the time delay in NCSs ranges from constant to random time delay, the NCS may be time-varying systems, which makes the analysis and design of such systems more challenging [11]. Some of the limitations such as poor QoS can be improved using effect NCS scheduling protocols.

C. NCS Scheduling Protocols

An essential ingredient of any NCS is its scheduling protocol. Currently, there are two kinds of scheduling protocols, static and dynamic. A static scheduling protocol predetermines the access to the communication channel for each node. The dynamic scheduler, on the other hand, determines the network schedule during the system operation.

One of the best known scheduling protocols was proposed by Walsh and Ye [7]. This is the so-called the maximum error first-try once discard (MEF-TOD) protocol. It arbitrates between multiple nodes, allocating network resources to data sensors, and maintains stability of the closed loop system. A feature of TOD is that it does not give equal access to all nodes.

II. MOTIVATION

Currently, we can distinguish between two research trends in the NCS area. The first trend is concerned with the control over network while the second deals with the control of the network. The control over network deals with analysis and design of control systems where the network-induced time delays can be estimated, modeled or computed. The second approach targets the control of the network-induced time delay. In this paper, we propose a scheduling protocol that compensates for the network-induced delay, improving QoS. Our proposed protocol provides more even access to the network for all nodes, guaranteeing that none of the nodes are unserviced.

Most scheduling protocol work properly in low network traffic situations. Under heavy traffic conditions, the network may leave some nodes unserviced for long periods of time. In particular, when the control applications require the same level of QoS to all subsystems, the scheduling protocol such as TOD or the conventional CAN access protocol do not guarantee this. Moreover, it may not be able to enforce a fair subdivision of the network bandwidth among the stations [7], [8].

In low traffic conditions, the network messages from different nodes seldom collide. When the network load increases, the chance of collision becomes higher, leading to an increase in the network-induced time delay. Under the high traffic conditions, network with low relative priority messages would be delayed by messages with a higher relative priority. It could even be delayed more than once by the same higher priority message irrespective of how many times the lower priority message has attempted to access the network. Examples of the scheduling protocols that may face such scenarios are carrier sense multiple access/non-destructive bit-wise arbitration (CSMA/NDBA) and the TOD protocols [7]. One of the objectives of the proposed protocol is to reduce the probability that low priority massages would be ignored for long periods of time.

III. PROBLEM STATEMENT

In this paper, we propose a protocol that addresses the issue of network fairness while maintaining the closed-loop system stability. An important issue in an NCS is the network-induced time delay which is caused by the network's presence. It is well known that time delays can destabilize a closed loop system. The proposed protocol reduces the destabilizing effect of network-induced time delays.

A. The Maximum-Error-First/Try-Once-Discard (MEF/TOD) Protocol

In the MEF/TOD protocol, when two or more messages are competing to gain the access to the shared medium, the message with the largest error wins the right to be transmitted. Because of this arbitration technique, this protocol is called the maximum-error-first. The term TOD reflects the fact that if a data packet fails to win the competition for the network access, this packet would be discarded while new data would be used in the next iteration [7].

The network induced error determines the priority level of each node's message. The error function used is proportional to the norm of the error's sub-vector of the plant inputs

and outputs. At the transmission times, the message with the greatest error norm wins the competition to be transmitted. In the case when two or more messages have the same error norm value, the node with the highest predetermined priority is being transmitted [7].

The TOD protocol is a dynamic protocol in the sense that it is allocating network resources based on the needs of the system. It is possible that with the TOD protocol, one of the nodes may hog the network for data transmission for a long period of time when the error of this node is large. Hence, other nodes' ability to access the network is greatly delayed. Unlike dynamic schedulers, static schedulers guarantee equal access to the network. In the following section, we propose a hybrid static-dynamic scheduling protocol that improves this aspect of the QoS by guaranteeing access to the network for low priority messages.

IV. TRAFFIC DIVISION ARBITRATION (TDA) PROTOCOL

In this section, we propose a protocol, which we refer to as the traffic-division arbitration (TDA) protocol. The TDA protocol is a combination of static and dynamic scheduling protocols. This protocol guarantees access to the network even for low priority messages and we show that the closed-loop system is stable.

A. TDA Protocol Description

The TDA protocol has two arbitration levels. The first level contains dynamic portion of TDA where the traffic of the network is divided into transmission cycles. In this level, a given threshold determines which of the competing messages are passed to the second level. The second level is static in the sense that messages from the first level are transmitted according to a globally pre-determined priority.

B. Illustrations of the TOD and TDA Protocols

In this Subsection, we discuss differences in the behavior between the TOD and TDA protocols using a simple example adapted from [8]. Figure 3(a) depicts the TOD protocol operation. The upper part of sub-figure 3(a) illustrates the sequence of nodes whose messages are to be sent. The lower part shows the transmission of messages and the sequence of messages attempting to gain access listed according to priority during the sending of a particular message. For example during the transmission of the first message from node 3, nodes 4 and 7 attempt to gain access and node 4 is determined to have higher priority. Note that the transmission of message number 7 was delayed by higher priority messages and that messages from node 3 were transmitted three times.

Figure 3(b) illustrates the operation of the TDA protocol. Before the transmission of the first message from node 3, the messages from node 3 and 7 exceeded the threshold for passage to the second level. Note that both messages 3 and 7 are sent. During the transmission of 3 and 7, messages from nodes 1, 3, and 4 exceeded a threshold and with priority as listed, and then transmitted next in order. Compared to TOD, TDA prevents the low priority messages from being discarded repeatedly. It is important to note that in TDA, all messages that are passed to the second level are sent. This combines the good features of both dynamic and static schedulers.

C. Notation and NCS Model Description

In this paper, we use the same notation as in Walsh and Ye [7, p. 439]. Figure 2 depicts a model of a node of an NCS consisting of the plant, network, and the controller.

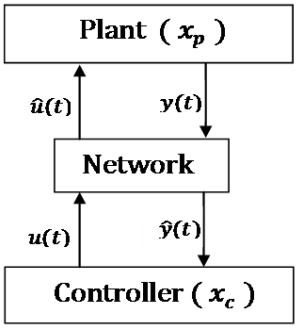


Fig. 2. A model of a node of an NCS.

The controller model is described as $\sum_c(\mathbf{A}_c, \mathbf{B}_c, \mathbf{C}_c, \mathbf{D}_c)$, where the controller state is $x_c \in \mathbb{R}^{n_c}$, the output $y \in \mathbb{R}^{n_r}$, and the input $u(t) \in \mathbb{R}^{n_q}$. The network states are denoted by $\hat{n}(t)^\top = [\hat{y}(t)^\top \hat{u}(t)^\top]$ where $\hat{y}(t)$ and $\hat{x}(t)$ are the delayed values of the output and input, respectively. The delay is caused by the presence of the network. Let the network-induced error be defined by $e(t)^\top := \hat{n}(t)^\top - [y(t)^\top u(t)^\top]$. The overall state of the controller and plant are $x(t)^\top = [x_p(t)^\top x_c(t)^\top]$. The combined NCS state is defined by $z(t)^\top = [x(t)^\top e(t)^\top]$. The system dynamics are represented as

$$\dot{z}(t) = \begin{bmatrix} \dot{x}(t) \\ \dot{e}(t) \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} x(t) \\ e(t) \end{bmatrix} = Az(t).$$

Without a network, $e(t) = 0$, and the dynamics reduce to $\dot{x}(t) = A_{11}x(t)$.

D. Error Bound and Stability Analysis in the TDA Scheduling Protocol

In this subsection we analyze the network-induced error $e(t)$ bounds when we apply the TDA scheduling protocol. We make the following assumptions:

Assumption 1: Without the network the closed loop system is asymptotically stable.

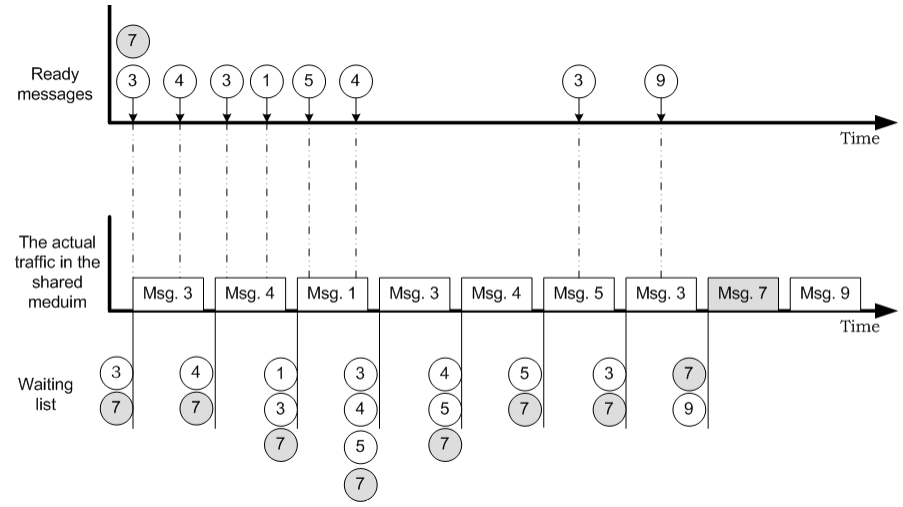
Assumption 2: The controller is continuous-time and the sampling delay and transmission time are negligibly small. In addition, the communication medium is error-free, and the observation noise is negligible.

Assumption 3: Each node grants the medium access one and only one time every k transmissions starting at time t_0.

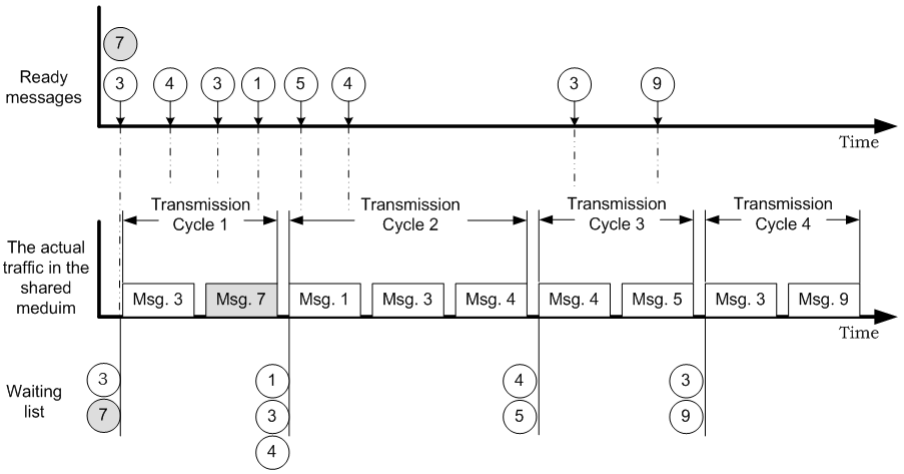
We use the following notation:

- maximum allowable transfer interval (MATI) is τ_m,
- number of nodes is k,
- maximum growth in error is β,
- starting time is t_0,
- error threshold e_t.

We next show the global exponential stability of the proposed hybrid scheduler.



(a) A traffic pattern with a TOD protocol.



(b) A traffic pattern with a TDA protocol.

Fig. 3. Traffic pattern comparison between TOD and TDA protocols.

By Assumption 1, the non-networked system $\dot{x}(t) = A_{11}x(t)$ is asymptotically stable, which means that for any $Q = Q^\top > 0$, the solution $P = P^\top$ to the Lyapunov matrix equation,

$$A_{11}^\top P + PA_{11} = -Q$$

is positive-definite.

Theorem 1: If the MATI is upper bounded by the minimum of

$$\frac{1}{4||A||\left(\sqrt{\frac{\lambda_{\max}(P)}{\lambda_{\min}(P)}}+1\right)k(k+1)},$$

and

$$\frac{1}{8\lambda_{\max}(P)\sqrt{\frac{\lambda_{\max}(P)}{\lambda_{\min}(P)}}||A||^2\left(\sqrt{\frac{\lambda_{\max}(P)}{\lambda_{\min}(P)}}+1\right)k(k+1)},$$

then the NCS, with the hybrid TDA scheduler, is globally exponentially stable.

Proof: We can use the method of Walsh, Ye, and Bushnell [7, p. 441], by verifying the error bound,

$$||e(t)|| < \beta\frac{k}{2}(k+1).$$

To verify the bound, note that each note is assigned a certain priority according to its network-induced error. This means that we have a predetermined order of transmissions, which is equivalent to the static round-robin scheduling criterion. This implies that we have a complete (consisting of k messages) TC with k transmissions in the time interval $[t-k\tau_m, t]$. Assuming that the nodes $n_1, n_2, ..., n_k$ are granted the medium access in the time instances $t_1, t_2, ..., t_k$ respectively, as illustrated in Figure 4. The network-induced error for each node in these instances is:

$$||e_{n_c}(t_c^-)|| < c\beta \text{ where } , c = 1, 2, ..., k.$$

Therefore, the weighted error at any time in this interval $t \geq t_0 + k\tau_m$ is:

$$||e(t)|| < \sum_{c=1}^{k} ||e_{nc}(t_c^-)|| < \sum_{c=1}^{k} c\beta = \beta\frac{k}{2}(k+1)$$

■

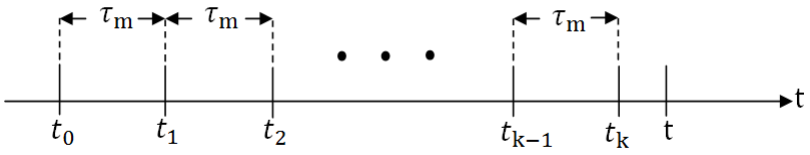


Fig. 4. Transmission times and the MATI τ_m.

V. EXAMPLE

In this section, we illustrate the TDA performance on a 12-node NCS and compare the TDA scheduler performance with that of the TOD. The 12 nodes consist of identical plants and controllers. We compare the performance of the two schedulers on the NCS shown in Figure 5.

A. System Description

The NCS used in the simulations is obtained by closing the feedback loops through a network. In the simulations, we considered the NCS comprising of 3 and 12 subsystems, respectively, in order to analyze the effect of increased traffic in an NCS. Each subsystem is an armature-controlled DC motor state-space model from [12, p. 150]:

$$\begin{cases} \dot{x}(t) = A_p x(t) + B_p u(t) \\ y(t) = C_p x(t) \\ x(t) \in \mathbb{R}^3, u(t) \in \mathbb{R}, \text{ and } y(t) \in \mathbb{R} \end{cases}$$

The motor transfer function is:

$$H_p(s) = \frac{4.626}{s^3 + 12s^2 + 22s + 20}.$$

In our simulation we used the following state-space realization:

$$A_p = \begin{bmatrix} -12 & -22 & -20 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}, B_p = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, C_p = \begin{bmatrix} 0 \\ 0 \\ 4.6 \end{bmatrix}^\top.$$

To improve the transient performance, we applied a state-feedback controller to each sub-system with controller gain:

$$K = \begin{bmatrix} -2.6 & 0.0864 & -0.005 \end{bmatrix}.$$

We first simulated the NCS using the TOD scheduler. Then we replaced the TOD scheduler with the TDA scheduler and repeated the simulation. In our simulation of the TDA scheduler, we used the algorithm shown in Figure 6. The transceiver represents the software/hardware components that are responsible for transmitting and receiving data to and from the shared medium.

B. Decision Functions

In our simulations, we used two decision functions for the schedulers to prioritize medium access. Both decision functions use the tracking error $\varepsilon_k(n)$ for the kth node at time n. The first decision function has the form:

$$d_k(n) = \frac{|\varepsilon_k(n) - \varepsilon_k(n-1)|}{|\varepsilon_k(n-1)| + \delta},$$

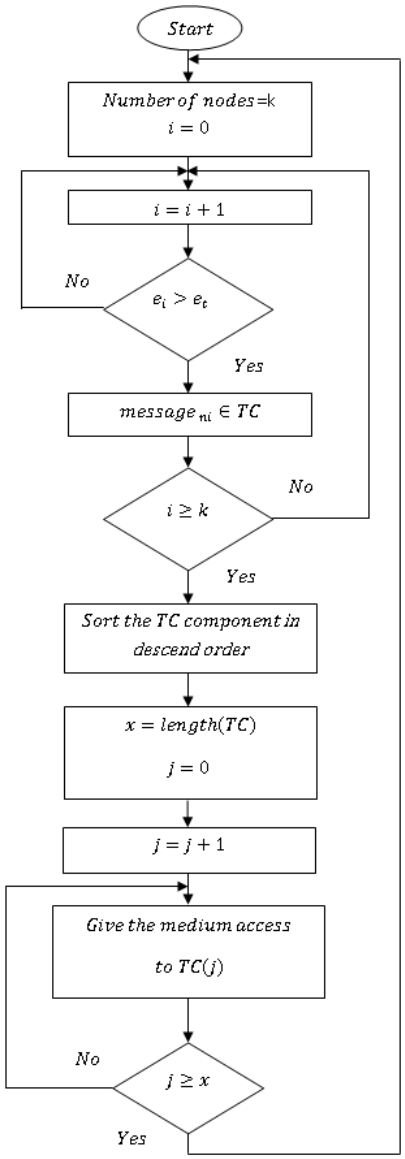


Fig. 6. TDA scheduler flow chart.

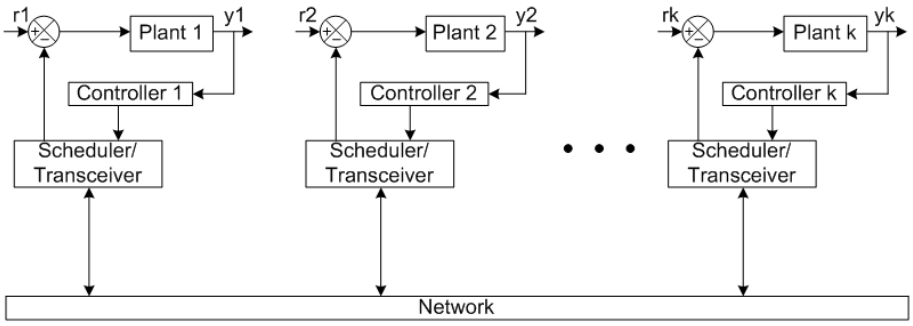


Fig. 5. NCS diagram used in the simulation study.

where $\delta > 0$ is a parameter used to prevent the denominator from becoming zero. The motivation behind this decision function is that d_k computes the relative error growth rate. The denominator is included to normalize tracking errors that come from different types of sub-systems. This decision function works as our simulation will show, but it may slow the rate of convergence as the error gets small.

The second decision function has the form:

$$\tilde{d_k}(n) = 2\varepsilon_k(n) - \varepsilon_k(n-1)$$

This decision function is a one-step predictor of the tracking error. Indeed,

$$\begin{aligned} \varepsilon_k(n+1) &\approx \varepsilon_k(n) + \varepsilon_k^{'}(n) \\ &\approx \varepsilon_k(n) + (\varepsilon_k(n) - \varepsilon_k(n-1)) \\ &= 2\varepsilon_k(n) - \varepsilon_k(n-1). \end{aligned}$$

The second decision function $\tilde{d_k}(n)$ essentially uses an approximate future tracking error $\varepsilon_k(n+1)$ in the scheduler operation to prioritize.

In the next subsection, we perform simulations using both decision functions.

C. Simulation Results

We used MATLAB for our simulations. We simulated a system consisting of 12 DC motors using the first decision function d_k. To improve the transient performance we incorporated PIDF controllers with the transfer function,

$$G_{PIDF}(s) = P + I\frac{1}{s} + D\frac{N}{1+N(\frac{1}{s})} = 8 + \frac{1}{s} + 8\frac{50}{1+50\frac{1}{s}},$$

where K_P, T_I, T_D, and N are the PIDF controller parameters, obtained using tuning rules from [9, Chapter 10]. Figure 7 shows the the performance of the system after adding the PIDF controller, which significantly improves the transient response. Figure 8 shows the improvement in the performance of the TDA-based NCS, using the second decision formula, $\tilde{d_k}$ over the first one, even without the use of the PIDF controller. In Figure 9, we compare the performance of the TDA and TOD schedulers using the second decision function, $\tilde{d_k}$. The plots show that the TDA performs excellently. We note that the TOD scheduler using $\tilde{d_k}$ does not work.

In Figure 10, the cumulative number of network access instances for a 12-node TDA system is plotted as a function of time. The plot illustrates the idea behind the TDA scheduler, which is to distribute access equitably. Note that there are only two nodes gaining more access to the network, while the other nodes gained access evenly.

VI. CONCLUSIONS

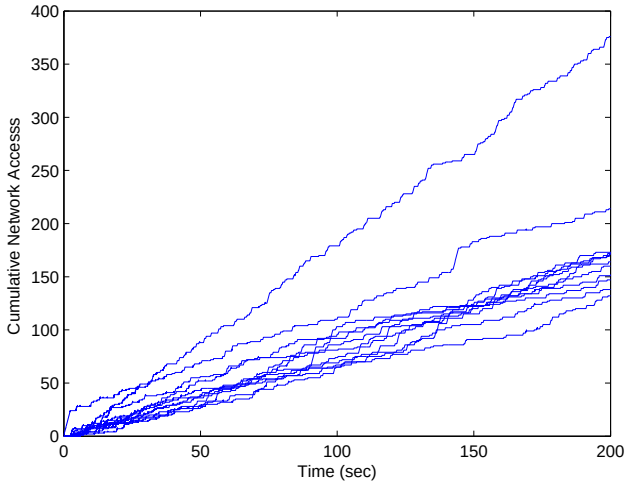


Fig. 10. Number of network access instances for a 12-node TDA system.

The contribution of this paper is a hybrid static-dynamic scheduler, called the traffic-division arbitration (TDA). The main objective for introducing this hybrid scheduler is to improve the quality of service (QoS) for each node in the network. The TDA scheduler governs the network access fairly, while at the same time, it assigns priority to the messages that have higher relative tracking errors. We showed that the error bound of the proposed scheduler is the same as the static and dynamic schedulers in the existing literature. In our simulations, we used identical sub-systems and we plan to investigate the behavior of the proposed scheduler on a diverse large scale NCSs.

Simulation examples are included to compare the performance of TDA scheduler versus TOD protocol. We used two different decision functions to prioritize the tracking error function. We used the same decision function in our implementations of TOD and TDA. In our implementations of TOD we used the same decision functions as in the

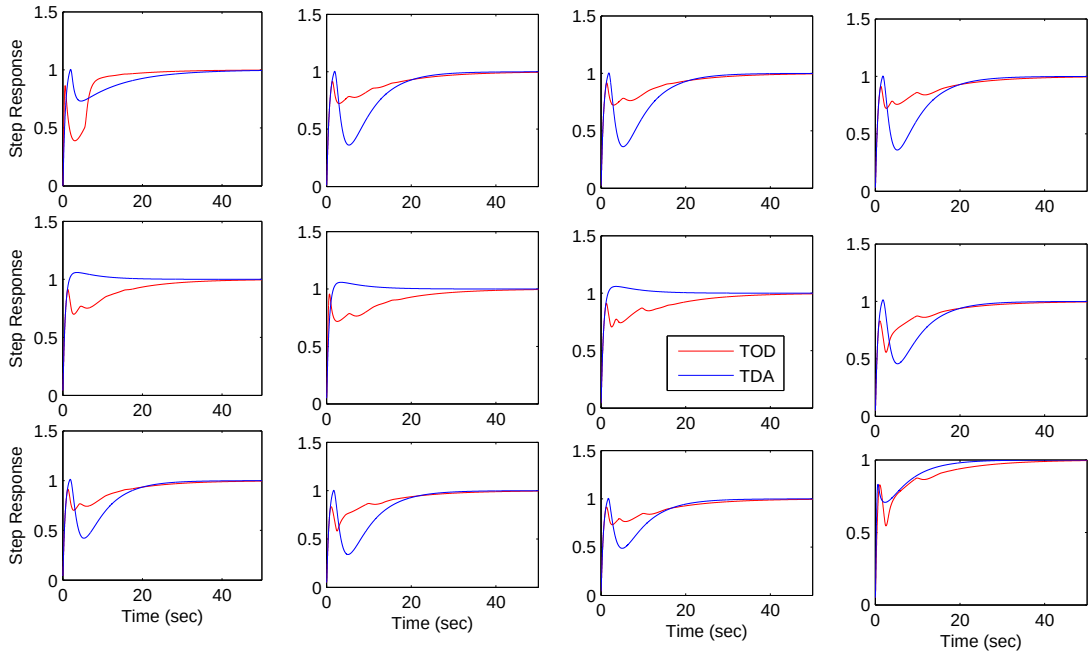


Fig. 7. Step response comparison of the TOD and TDA schedulers' on a 12-node system with PIDF using d_k.

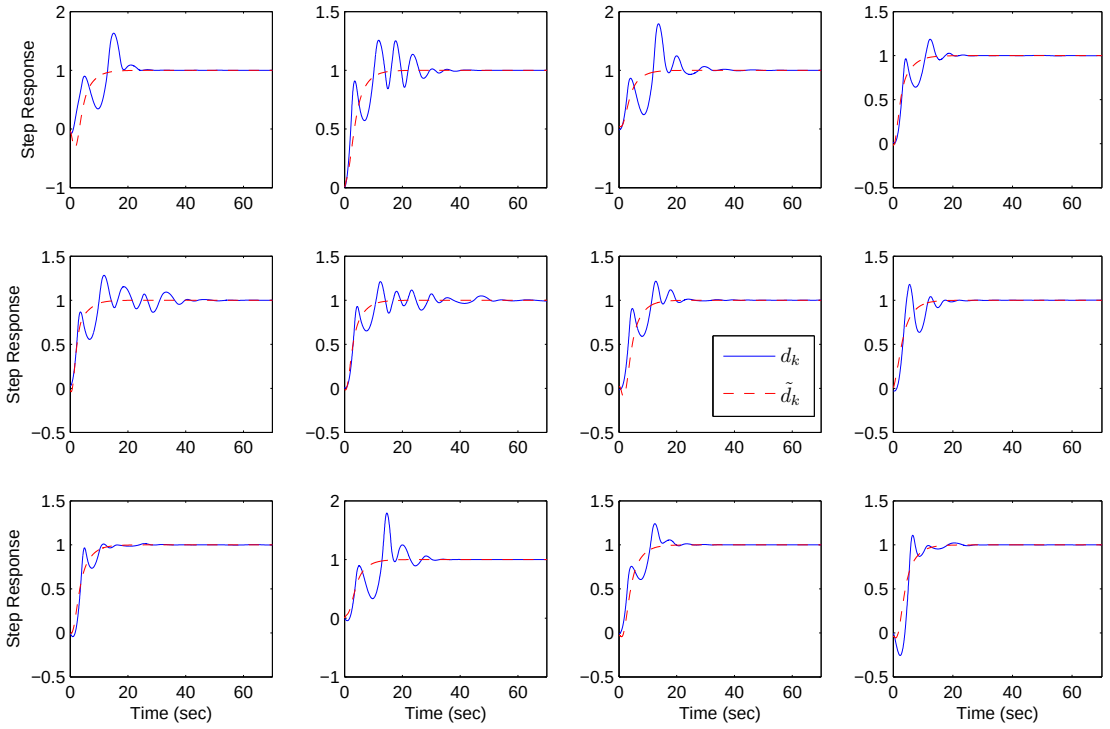


Fig. 8. Step response comparison of the TDA schedulers' on a 12-node system between d_k and $\tilde{d}_k$.

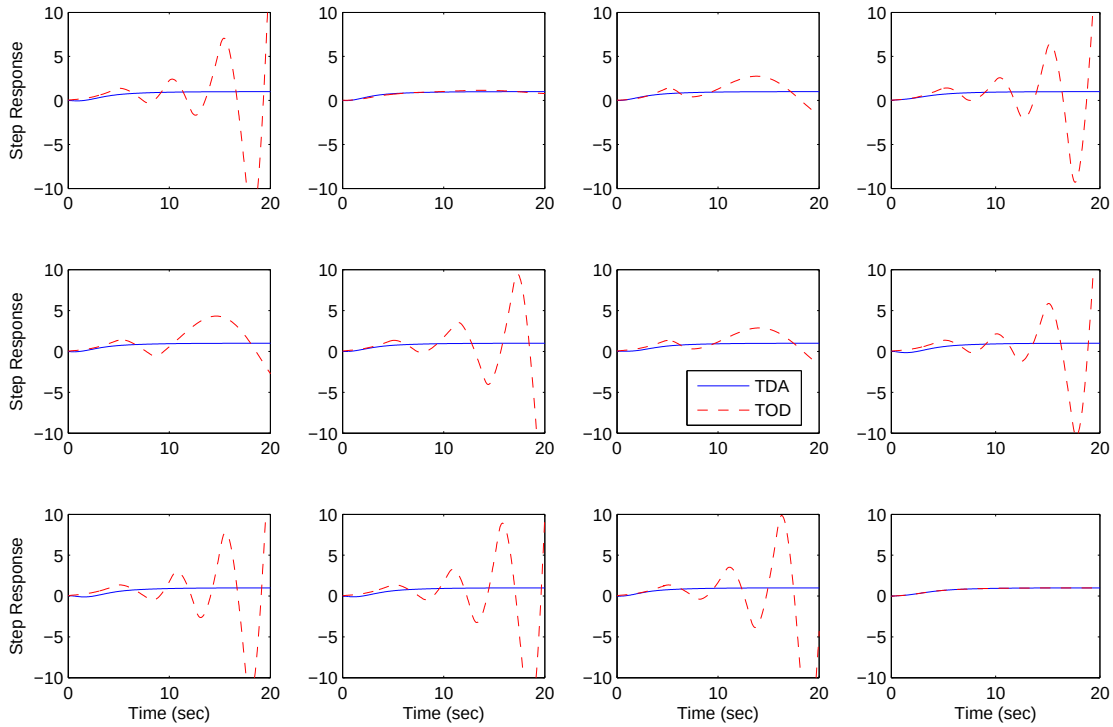


Fig. 9. Step response comparison of the TDA and TOD schedulers' on a 12-node system using $\tilde{d}_k$.

TDA implementations. In our simulations, we used a static error-threshold. It is our plan to investigate using dynamic error-threshold, which we believe should further improve the performance of the NCS.

REFERENCES

[1] D. Hristu-Varsakelis and W. S. Levine (Ed.): Handbook of Networked and Embedded Control Systems, 2005

[2] G. C. Walsh and H. Ye, "Scheduling of networked control systems," Control Systems, IEEE , Vol. 21, No. 1, pp.57–65, Feb 2001

[3] T. C. Yang, "Networked Control System: A Brief Survey," IEE Proceedings: Control Theory and Applications, Vol. 153, No. 4, pp.403–412, July, 2006

[4] Z. Huo, H. Fang, and C. Ma, "Networked control system: State of the art," Proceedings of the World Congress on Intelligent Control and Automation (WCICA), Vol. 2, pp. 1319–1322, 2004

[5] H. F. Othman, Y. R. Aji, F. T. Fakhreddin, A. R. Al-Ali, "Controller Area Networks: Evolution and Applications," Proceedings of the 2nd International Conference on Information and Communication Technologies, Vol. 2, pp. 3088–3093, 2006

[6] E.-H. Jung and H.-H. Lee, "A study on controller design for network-induced time delay system," Proceedings of the International Conference on Mechatronics and Information Technology, Vol. 6042, September 2006

[7] G. C. Walsh, H. Ye, L. G. Bushnell, "Stability analysis of networked control systems," IEEE Transactions on Control Systems Technology, Vol. 10, No. 3, pp. 438–446, May 2002

[8] T. A. Beitelmal, A. M. El-Mahdi, and H. S. Abdulkarim, "New technique to enforce a fair behavior in accessing the medium in the controller area network," Proceedings of the 2nd International Conference on Electrical Systems Design and Technologies, Hammamet, Tunisia, November 8–10, 2008

[9] K. Ogata, Modern Control Engineering, Prentice Hall, Upper Saddle River, New Jersey, third edition, 1997

[10] S. Bennett, "Development of the PID controller," IEEE Control Systems Magazine, Vol. 13, pp. 58–65, 1993

[11] E.-H. Jung and H.-H. Lee, "A study on controller design for network-induced time delay system," Proceedings of The international Conference on Mechatronics and Information technology, Vol. 6042 I, 2005

[12] S. H. Żak, Systems and Control, Oxford University Press, New York, 2003