SCHEDULING AND CONTROL CO-DESIGN FOR DELAY COMPENSATION IN THE NETWORKED CONTROL SYSTEM

Liang Chen, Jianming Zhang, and Shuqing Wang

ABSTRACT

The networked control system (NCS) is currently receiving increasing attention from researchers. Researches on this subject, however, have not considered the co-design of network quality of service (QoS) and control quality of performance (QoP). This paper proposes a novel NCS design framework based on scheduling and control co-design to compensate for random network-induced delays. In the framework, a scheduling algorithm used to find the optimal sampling regions of control loops performs rough adjustment and guarantees the network QoS, while a novel adaptive fuzzy PID controller is designed to perform accurate adjustment to guarantee the control QoP. Practical application results obtained with a multi-loop NCS show that the framework can ensure satisfactory performance due to its robustness against network uncertainty.

KeyWords: Networked control system, delay compensation, sampling time scheduling, fuzzy PID, NCS co-design.

I. INTRODUCTION

During the last decade, the technologies of computing, communication, and control, called the C3 environment [1], have progressed rapidly and given birth to the networked control system (NCS). The NCS is an architecture in which sensors, controllers, and actuators are connected together to build real-time feedback loops in a network. The typical topology of a networked control system is shown in Fig. 1.

Networked control plays an increasing important part in modern technologies, and the NCS can be applied in many applications in the process industry, manufacturing automation, automobiles, advanced aircraft, intelligent traffic, intelligent buildings etc.

One important feature of the NCS is that, instead of the instruments being wired with point-to-point connections, sensors, controllers, and actuators are all connected together via the network. Hence, the NCS has more advantages over the conventional control system, such as reduced wiring, easier maintenance, remote control, improved efficiency, and increased system flexibility.

One issue inherent to NCS, however, is the network-induced delay that occurs while data is exchanged among devices connected to the same medium [2]. The delay can degrade the performance of the NCS and even destabilize the system in the worst case. The characteristics of these time delays can be either constant or random, either bounded or boundless because of the effects of various factors, such as the network topology, communication protocol, load condition, transmission rate etc.

![Fig. 1. Typical topology of an NCS.](image-url)
In order to handle these notorious network-induced delays, researchers in this area have focused on two approaches: one is to design the controller without regard to the delay, and the other is to design a communication protocol that minimizes the delay. Advanced controller design is necessary to guarantee control quality of performance (QoP), whereas an appropriate communication protocol is required to ensure the network quality of service (QoS). From the viewpoint of guaranteeing QoP, this can be regarded as a problem of control; from the viewpoint of guaranteeing QoS, it is a scheduling problem. Much progress has been made in the study of network scheduling. Hong [3] developed a sampling time scheduling algorithm to allocate network bandwidth and select proper sampling times for a multi-loop NCS. It was originally used on a periodic delay network and later modified and applied to random delay networks, such as CAN [4], and cyclic-service fieldbuses, networks [5] and token-passing fieldbus such as Profibus [6]. Kim et al proposed a new method to obtain a maximum allowable delay bound (MADB) for scheduling an NCS [7]. Lian et al. [8] provided several useful network design methods for obtaining the optimal sampling times when designing an NCS. A novel control network protocol, called try-once-discard (TOD), was introduced by Walsh et al. [9] for a multiple input multiple out (MIMO) NCS. Meanwhile, the controller design problem for the NCS has also attracted a lot of interest. Various control methodologies have been developed that include augmented deterministic discrete-time model methodology [10], perturbation methodology [11], robust control [12], optimal stochastic control [13,14], predictor-based control [15], fuzzy logic control [16,17] etc.

The above researches have been fruitful, but, there is one fatal flaw. Both approaches, studying used through computer science and controller design carried out through control engineering, focus only on one direction. Little attention is paid to combining the two. Many methodologies proposed for networked control lack consideration of the co-design of network QoS and control QoP. Furthermore, in the networked control environment, control QoP is in contradiction with network QoS [8]; i.e., improvement in control QoP is achieved at the expense of synchronous degradation of network QoS. As a result, any NCS design methodology that solely aims at improving QoS simply through network scheduling or improving QoP simply through controller design cannot lead to satisfactory system performance and is not practical. To solve this problem, we propose a novel NCS co-design framework, in which network scheduling and controller design are complementary to each other and perform rough adjustment and accurate adjustment, respectively. In the NCS co-design framework, most methods for scheduling and control can be accommodate. However, these methods must be adapted to real applications, and their complementarity must be ensured.

This paper is organized as follow. Section II provides detailed analysis of the features of network-induced delay. Section III presents the NCS co-design framework for delay compensation, based on sampling time scheduling and fuzzy PID control. In section IV, experimental results are presented to show the power of the NCS co-design framework. This paper concludes with section V.

II. ANALYSIS OF NETWORK-INDUCED DELAY

Since the network-induced delay is inevitable and can adversely affect the performance of the control system, the analysis of delay features becomes extremely important. The total time delay in an NCS can be expressed as: \( \tau = \tau^{pre} + \tau^{trans} + \tau^{post} \), where \( \tau \) is the controller processing time delay, while network delays include the sensor to controller delay \( \tau^{ca} \) and controller to actuator delay \( \tau^{act} \). Although \( \tau \) always exists, it is usually small and can be neglected in most cases [19]. Therefore, the above three delays can be abbreviated as the network delay \( \tau^{trans} \) and \( \tau^{post} \), which are composed of the parts shown in Fig. 2. Here, \( \tau^{pre} \) is the preprocessing time delay that the source nodes incur while encoding data into frames or packets and place them onto the network; \( \tau^{wait} \) is the waiting time delay that a message consumes while waiting in the queue at the source node until it is given access to the network; \( \tau^{post} \) is the transmission time delay that a frame or a packet incurs while traveling through a physical media; \( \tau^{wait} = N/DR + Dist/3 \times 10^8 \); \( N \) is the message size in terms of bits; DR is the data rate; \( Dist \) represents the distance in terms of meters; \( \tau^{post} \) is the postprocessing time delay that the destination node incurs while encoding the network frames or packets into a physical data format.

Among the above four delays, \( \tau^{pre}, \tau^{trans}, \) and \( \tau^{post} \) are all deterministic once the network protocol is fixed, whereas waiting time delay \( \tau^{wait} \) may be the most significant uncertainty leading to the time-varying characteristic of network delay. The waiting time delay \( \tau^{wait} \) is primarily affected by the network protocol. The motivation of researches on scheduling through proper network protocol design is to minimize the uncertain waiting time delay to guarantee network QoS.

From the above analysis, it can be concluded that the characteristic of the delay basically depends on the type of network used, or to be precise, on the type of medium access control (MAC) used. There are, mainly, two different

![Fig. 2. Diagram of the composition of network delays.](image-url)
types of networks according to the medium access control protocols.

Cyclic service networks. One important feature of cyclic service networks is that they are all operated on the basis of cyclic token passing discipline, where data generated at each node is served in a cyclic order. Several cyclic service networks for industrial control have been released, including IEEE 802.3 Token Bus, IEEE 802.4 Token Ring, and Profibus. Since sensor and control signals are transmitted in a cyclic way, the delays are periodic and bounded.

Random access network. Random access networks, such as CAN and Ethernet, involve more uncertain delays that are nonperiodic. The medium access control protocol of Ethernet is called carrier sense multiple access with collision detection (CSMA/CD). In this protocol, any station can access the network when it becomes idle. However, the delays generated by collisions may be longer or even unbounded. Unlike Ethernet-like networks, CAN is a priority-based fieldbus, with carrier sense multiple access with collision avoidance (CSMA/CA) MAC. The collisions in CAN are nondestructive, and the delays are bounded. Few papers have focused on dealing with troublesome delays in random networks, a problem that is considered in this paper.

III. NCS CO-DESIGN OF SCHEDULING AND CONTROL

3.1. NCS co-design framework

Generally, a smaller sampling time in a discrete control system will lead to performance that is close to that of the continuous control system. In an NCS, however, a smaller sampling time may increase the network load and data loss possibility, which in turn results in longer network-induced delays. The relationship between a sampling time and the performance of an NCS is shown in Fig. 3. In fact, a smaller sampling time may provide better control QoP but may also degrade the network QoS. In a bandwidth-limited network, degradation of network QoS can further worsen the control QoP due to long time delays. Thus, selecting an optimal sampling time that can guarantee both network QoS and control QoP is important when designing an NCS. To find the optimal sampling time (point A in Fig. 3), several approaches have been proposed by Hong [3-6], Lian [8,21], Kim [7] and Park [22] and others.

Unfortunately, these approaches are fragile because in practical applications, point A cannot be easily found and will probably shift to a suboptimal point (point B or point C) for the following two reasons:

1) To calculate the optimal sampling time, the above approaches must predetermine the maximum allowable delay bounds (MADBs) of all control loops, which are necessary for stability analysis and network scheduling of an NCS. In non-networked control systems, the MADBs can be obtained through classical frequency domain analysis, such as Bode or Nyquist diagrams. However, these analyses do not work in a networked control environment because the MADBs depend not only on the characteristics of the feedback loops of the plant and controller but also on the network conditions. There are several methods exist in the literature for obtaining MADBs [Branicky [2], Kim [7], Lian [8], Park [22], Moon [23], Weizhang [24] etc.]. In these methods MADBs are obtained through Ricatti or LMI approaches. The results are too conservative to be of practical use and must be improved, which means that the optimal sampling times acquired based on the above MADB analyses are conservative, too. In such cases, the optimal point usually shifts from point A to point C.

2) To calculate the optimal sampling time, the above scheduling approaches are based on a given network condition, which implies that the network characteristics, such as the network protocol, nodes, traffic load etc., are all time-invariant. However, a random access network like Ethernet involves more network uncertainties, including traffic load variation, adding or removing nodes, transmission collision, retransmission etc., which means that the network condition is time-variant and that the optimal point A should be moved to point B or point C at times. For instance, when the traffic load is heavy, collisions will multiply, and in this case, the optimal sampling time should be shifted from point A to point B; when the traffic load is light, the optimal point A should be shifted to point C.

From the above analysis, the optimal sampling time can seldom be obtained in practical applications; only an optimal sampling region containing points A, B, and C can be acquired. Then how can we further improve the control QoP after the optimal sampling region is calculated, especially at point B and point C? A natural idea is to improve the performance through the adaptive control approach. When network variation appears, the adaptive controller with a list of heuristic rules works like a human expert to update the controller parameters to improve the control QoP. Once the optimal sampling point shifts to point B (or C), the control QoP can be greatly improved by the adaptive controller. Even at point A, the control QoP can be
slightly improved. These results are validated by the NCS experiments described in section IV.

Basically, the idea of NCS co-design is to provide a general framework to accommodate methodologies for scheduling and control, and to make them complementary to each other. In this paper, a sampling time scheduling algorithm and a novel adaptive fuzzy PI controller are used in the framework. The sampling time scheduling algorithm described in part 3.2 can guarantee the network QoS and reduce the network delay, while the fuzzy PI controller described in part 3.3 can improve the control QoP. The goal of this paper is to maximize the system performance in the networked control environment via the NCS co-design approach. The NCS co-design framework employs a relatively systematic design procedure shown in Fig. 4.

The framework starts with initialization of the control system, which includes initialization of the PID controller gains and the plant models. These initial values can be easily obtained either from practical systems or from classical technologies, such as the Ziegler-Nichols tuning method and step response modeling technology. The initialization step can be used to perform both MADB analysis of control loops and fuzzy PID controller design. In the framework, the optimal sampling times are calculated off-line based on MADB analysis and network definition, while the gains of the PID controller are adjusted by the fuzzy controller on-line to compensate for the uncertain delays in the NCS. The performance can be evaluated by means of performance indexes, such as ISE and ITAE. If the performance is unacceptable, the system should be redesigned with new initialization (better-tuned PID gains and more accurate models).

### 3.2. Sampling time scheduling

Among the sampling time scheduling algorithms existing in the literature, Hong’s [3-6] method is the most typical and comprehensive one because it can be modified and applied to both cyclic service networks and random access networks. Hong developed the sampling time scheduling algorithm to appropriately specify the sampling times of an NCS composed of multi control loops. Consider a multi-loop NCS like that in Fig. 1, which includes $N$ loops. Let $\Phi_i$, $i = 1$ to $N$, be the MADB of control loop $i$.

The window concept is used here, as shown in Fig. 5, to formulate the sampling time scheduling algorithm. Here, $L$ is the transmission time of a packet or a frame, and $\sigma$ is the overhead of each transmission. Compared with the transmission time, $\sigma$ is small and often can be neglected. $r$ is the number of the packets or frames that can be transmitted in the medium in the worst case of data latency existing in the control loop.

Define $\Phi$ as a vector arranged according to the increasing order of $\Phi_i$, i.e., $\Phi_i \leq \Phi_{i+1}$, $i = 1$ to $N$, and let $T$ be a vector of the sampling times of $N$ control loops:

$$
\Phi = [\Phi_1, \Phi_2, \ldots, \Phi_N],
$$

$$
T = [T_1, T_2, \ldots, T_N],
$$

where $T_i$ is arranged in the same order according to $\Phi_i$. Then, for the predetermined MADB $\Phi_1$, the minimum sampling time $T_1$ can be calculated as follows:

$$
T_1 = \frac{\Phi_1 + L}{3}.
$$

The sampling times of other control loops, in a generic case, are multiples of $T_1$ and can be calculated as follows:

$$
T_i = k_i T_1, \quad \forall i = 2, \ldots, N,
$$

$$
v = \frac{\Phi_i - T_i + L}{2T_1}, \quad \forall i = 2, \ldots, N,
$$

$$
k_i = \Lambda(v), \quad \forall i = 2, \ldots, N,
$$

where $k_i$ is the power of two, i.e., $2^{k_i}$, $k_i \in \{0, 1, 2, \ldots\}$. And $k_i = \Lambda(v)$ implies that $2^{k_i}$ is “closest” to but does not exceed $v$.
Define $m$ as the number of nodes connected to the network. If the network load is light, which means that $r \geq m$, then the sampling times of other loops can be simplified:

$$T_i = \frac{\Phi_i - T_i + L}{2}, \quad \forall i = 2, \ldots, N.$$  \hspace{1cm} (7)

However, there are two disadvantages in Hong’s algorithm. First, the MADBs are obtained through classical frequency domain analysis, which indicates that they are incorrect and of little use in practical NCS design. Second, as in other network scheduling algorithms, the optimal sampling times are calculated for a given network, which means they are inaccurate when the network changes. These problems are solved by the NCS co-design framework.

### 3.3. PID controller reconstruction based on fuzzy logic

The PID control is by far the most widely used form of feedback in industrial systems. It is the first approach that is tried when feedback is adopted. More than 90% of the control loops in physical plants are PID [18]. Nowadays, many practical NCSs employ such PID control systems by connecting the controllers, sensors, and actuators to the network. The PID control strategy often remains unchanged for economic reasons. However, the performance of such NCSs is inevitably degraded due to the inherent network-induced delay. Hence, one flexible way to compensate for the delays existing in the networked PID control loops may be to reconstruct PID controllers rather than to design a brand-new advanced controller to improve the performance of the NCS.

When confronted with uncertain network delays existing in PID control loops, a skilled operator in a physical plant may adjust the gains of the PID controller according to the system performance. In this way, the degradation caused by the delays can be reduced. The above idea can be adopted to transform a PID controller into a fuzzy PID controller.

Almutairi et al. [19] proposed the fuzzy logic modulation methodology for delay compensation in an NCS. However, only the output of the PI controller is updated externally by the fuzzy controller; the gains of the PI controller cannot be updated on-line. In addition, the fuzzy controller has only two fuzzy rules, and the membership functions should be tuned through sophisticated on-line and off-line optimizations. In this paper, a novel adaptive fuzzy PID controller is proposed to overcome the above shortcomings. The fuzzy controller contains 49 well-defined rules and can update the PID gains on-line according to network conditions.

Figure 6 shows the structure of the adaptive fuzzy PID controller. As shown in the figure, a physical plant is controlled over the network. Here, the fuzzy controller takes the place of the human expert. The knowledge about how to control a process under the uncertain network environment is distilled and applied as fuzzy rules. When the performance is degraded, the gains of the PID controller can be adjusted adaptively by the fuzzy controller. Thus, the conventional PID controller is transformed into an adaptive fuzzy PID controller to perform delay compensation.

Since the PID control strategy often remains unchanged, the main part of the reconstructed controller for the design is the fuzzy controller. As shown in Fig. 7, it consists of four parts [20]: (1) a fuzzifier that converts the controller input into linguistic values; (2) a rule base that contains a set of the expert’s linguistic descriptions of how to achieve good control; (3) an inference engine that emulates the expert’s decision making in applying knowledge about how best to control the plant; (4) a defuzzifier that converts the conclusions of the inference mechanism into actual inputs for the process. In Fig. 7, $\hat{e}(k)$ is the reference input, in each sampling period; define $e(k) = \hat{e}(k)$; $\hat{e}(k)$ is the error between reference input and plant output; $\Delta e(k)$ is the change of the error; $\Delta e(k)$ and $\Delta \hat{e}(k)$ are the inputs to the fuzzy controller, while the outputs are $\Delta u_p$, $\Delta u_i$, $\Delta u_d$. The gain increments $\Delta K_p$, $\Delta K_i$, and $\Delta K_d$ are expressed as follows.

$$e(k) = r(k) - y(k - \tau^w_k), \hspace{1cm} (8)$$

$$\hat{e}(k) = \frac{e(k)}{r(k)} = \frac{r(k) - y(k - \tau^w_k)}{r(k)}, \hspace{1cm} (9)$$

$$\Delta \hat{e}(k) = \frac{e(k) - e(k-1)}{r(k)} = \frac{y(k - \tau^w_k) - y(k - \tau^w_k)}{r(k)}, \hspace{1cm} (10)$$

$$\Delta K_p = \Delta u_p K_p, \quad \Delta K_i = \Delta u_i K_i, \quad \Delta K_d = \Delta u_d K_d, \hspace{1cm} (11)$$

where $K_p$, $K_i$, and $K_d$ are the initial gains of the PID controller, and $\tau^w_k$ is the delay in sampling period $k$.

Figure 8 shows the membership functions of the input and output linguistic variables. Seven fuzzy linguistic variables, i.e., Negative Big (NB), Negative Medium (NM),
Negative Small (NS), Zero (Z), Positive Small (PS), Positive Medium (PM), and Positive Big (PB), are defined. The range of the input variables $\hat{e}$ and $\Delta \hat{e}$ is $(-1, 1)$. In order to simplify the design procedure, the membership functions of the output variables $\Delta \hat{u}_p$, $\Delta \hat{u}_i$, and $\Delta \hat{u}_d$ are defined identically and expressed as the percentage of the original gains $K_p$, $K_i$, and $K_d$ as shown in Eq. (11). The range of the output variables is limited to $(-0.5, 0.5)$ based on both the performance analysis and the stability consideration. Hence, the fuzzy controller can be applied to reconstruct a large number of PID controllers without regard to the specialty of the original control systems.

Table 1 does not contain the fuzzy rules for $\Delta \hat{u}_d$, which can be obtained using a similar method. There are total 49 rules in Table 1. Mamdani’s min-max inference method and the center average defuzzifier are used in the fuzzy controller design.

### IV. EXPERIMENT VALIDATION OF NCS CO-DESIGN

In order to evaluate the effectiveness of the NCS co-design framework proposed in this paper, a practical NCS composed of two networked control DC motors was designed as an experimental test bed. A DC motor has the advantage of fast step response and is sensitive to the delays existing in control loops. These are desirable motor performance characteristics for many industrial applications. Hence, DC motors are the most suitable ones for the evaluation of system performance in an NCS.

Experimental platform. The experimental setup for networked control is shown in Fig. 9. Five computer nodes were connected to the Ethernet-based networks. Two computers with the Linux systems served as the sensors and actuators of DC motor 1 and motor 2, while two other computers with Windows systems implemented the control algorithms to control the DC motors over the network. DC motor 1 and motor 2 were controlled by controller 1 and controller 2, respectively. In the experiment, the time-driven sensor sampled the process output (the shaft angular velocity of the DC motor) periodically and sent each sample to the controller nodes over the network. When a sample was received, the event-driven controller computed the control signal and sent it to the event-driven actuator node, where it was subsequently actuated.

In fact, the network used in the experiment was a LAN (Local area network) in our campus networks. The LAN is 10mbit-Ethernet and can access the Internet via a router server. There are several other LANs connected to the router. And the computers in these LANs may visit the ftp server in LAN 1 at times. In the experiment, the values of $\tau_{\text{c}}$, $\tau_{\text{pre}}$, and $\tau_{\text{post}}$ were quite small compared with the waiting time delay $\tau_{\text{wait}}$ that directly depended on the CSMA/CD protocol. When network traffic was heavy, more collisions...
were generated, resulting in longer network delays. Hence, the NCS involved more network uncertainties, and more obstacles were encountered when the DC motors were controlled over the networks.

As shown in the flow chart of the co-design framework, the models of the DC motors must be acquired for initialization of the NCS co-design. In the experiment, the DC motors were modeled using different methods. Figure 10 shows a schematic diagram of the DC motors. We assumed the motors were linear and time-invariant. For motor 1, a second order model can be given by

\[
\begin{aligned}
\dot{x} &= \begin{bmatrix}
\frac{di_a}{dt} \\
\frac{d\omega}{dt}
\end{bmatrix} =
\begin{bmatrix}
-R -K_b \\
L & -L
\end{bmatrix}
\begin{bmatrix}
i_a \\
\omega
\end{bmatrix} +
\begin{bmatrix}
1 \\
0
\end{bmatrix} u,
\end{aligned}
\]

\[
y = \begin{bmatrix}
i_a \\
\omega
\end{bmatrix},
\]

where \(u\) is the armature voltage, \(x = [i_a, \omega]^T\), and \(i_a\) and \(\omega\) are the armature current and shaft angular velocity, respectively. The parameters of the actual motors were estimated via system identification technology, and their values are shown in Table 2. Thus, the transfer function could be easily calculated as follows:

\[
G_{M_1}(s) = \frac{C(sI - A)^{-1}B}{s^2 + 1138s + 30530}, \quad (14)
\]

DC motor 2 was modeled using a signal compression method based on the assumption that it could be modeled as a first-order system. The resulting transfer function was

\[
G_{M_2}(s) = \frac{y(s)}{u(s)} = \frac{4529}{s + 8.5}. \quad (15)
\]

The DC motors were originally under direct digital control of the PI controllers described below:

\[
u_1(k) = u_1(k-1) + k_{p1}(e_1(k) - e_1(k-1)) + k_{i1}T_s e_1(k), \quad (16)
\]

\[
u_2(k) = u_2(k-1) + k_{p2}(e_2(k) - e_2(k-1)) + k_{i2}T_s e_2(k), \quad (17)
\]

where the gains of the PI controllers were \(k_{p1} = 12.2, k_{i1} = 7.5, k_{p2} = 0.006,\) and \(k_{i2} = 0.055\). The sampling times \(T_{s1}\) and \(T_{s2}\) were 20ms and 10ms for DC motor 1 and motor 2, respectively. Perfect control was achieved by DDC as shown in Fig. 14.

When the DC motors were connected to the network to form an NCS as shown in Fig. 9, however, the system performance was seriously degraded. The motors finally became uncontrollable because the initial sampling times were so small for networked control that they resulted in extremely heavy network traffic and long time delays.

Hence, in order to reduce the network-induced delay, the NCS had to be carefully designed based on the co-design framework. Sampling time scheduling was a prerequisite. The key problem in the scheduling algorithm is the MADB analysis. In the experiment, the MADBs of two control loops were obtained using the method of trial and error rather than by using any academic method because the MADBs calculated with these methods were too conservative to be of practical use. We limited the bandwidth of the communication port of the sensor at the Linux station to generate a series of delays and then defined the MADB as the time delay under which the motor became marginally stable. With this approach, the optimal sampling region could be calculated (note: as stated above the optimal sampling time could not be acquired in practice). Here, the MADBs were \(\Phi_1 = (190-220)\)ms and \(\Phi_2 = (160-200)\)ms. Thus, control loop 2 was the most sensitive loop. LAN 1 contained 5 nodes \((m = 5)\), the minimum frame size used was about 20 bytes, and the distance between the source and the destination was about 50 meters. Thus, the transmission time of a frame could be calculated as \(L = 20B/(10M/s) + 50m/(3 \times 10^3 m/s) \approx 2\)ms. Hence, the optimal sampling regions for motor 2 and motor 1 were \(T_{s2} = (\Phi_2 + L)/3 = (54.67)\)ms and \(T_{s1} = (\Phi_1 - T_{s2} + L)/2 = (64.78)\)ms. The step responses of the networked control DC motors with sampling time scheduling (denoted as \(T_s\) scheduling in the figure) are shown in

| \(R\) | 4.67Ω |
| \(L\) | 170 \times 10^{-3} H |
| \(J\) | 42.6 \times 10^{-4} Nm·s/rad |
| \(B\) | 47.3 \times 10^{-7} Nm·s/rad |
| \(K\) | 14.7 \times 10^{-3} Nm/A |
| \(K_b\) | 14.7 \times 10^{-2} V·s/rad |

Table 2. DC motor parameters.
Fig. 11. It is obvious that the DC motors could be controlled over the network and that fast response in speed regulation was achieved. Also, it was easy to find the actual optimal sampling times: for motor 1, about 70ms, and for motor 2 about 60ms. When a smaller sampling time is selected, it results in longer network delays that greatly degrade the performance and can even destabilize the system in the worst case. This is verified by the results shown in Fig. 13. Take motor 1 for example; when $T_{s1} = 60$ms, the delays existing in the control loop were much longer than those at 70ms. At the optimal sampling time (70ms), the delays were almost constant (25ms), which indicates that no collisions were generated when the control signals and sensor signals were transmitted on the Ethernet, and the network QoS was greatly improved by the scheduling algorithm. In Fig. 13, around 80 seconds, delays were extremely large and packet dropout occurred. This indicates that the ftp server was visited by other PCs, leading to heavier network traffic.

However, at the suboptimal sampling times (for motor 1, 60ms and 80ms; for motor 2, 50ms and 70ms), the step responses involved more oscillations and higher overshoot, which means that the control QoP needed to be improved after QoS scheduling. In the co-design framework, an adaptive controller is designed based on PI controller reconstruction. The fuzzy PI controller provides higher control QoP since the PI controller can continue to perform the control function while the fuzzy controller adjusts the gains of the PI controller on-line. Figure 11 shows the step responses of the DC motors with NCS co-design. The performance was greatly improved at the suboptimal sampling times and slightly improved at the optimal point. The step
responses of the DC motors with direct digital control, sampling time scheduling ($T_{s1} = 70\text{ms}$, $T_{s2} = 60\text{ms}$), and NCS co-design are compared in Fig.14. The power of the co-design framework is clear because the DC motors achieved almost perfect control.

**Remark 1.** One conclusion drawn from thorough analysis of the experiment results is that the sampling time scheduling algorithm in the co-design framework performs rough adjustment, while the fuzzy PID controller performs accurate adjustment. This is illustrated in Fig. 12. A discrete control system with a smaller sampling time will achieve better performance at point D. The performance worsens from point D to point E when the system is controlled over a network. The optimal sampling region containing the optimal point A and suboptimal points B and C is acquired by the scheduling algorithm, and the performance is greatly improved due to the improvement of the network QoS. Here, this is called rough adjustment. The improvement of the control QoP can be achieved by the fuzzy PI controller due to compensation of the time-varying delays. When the sampling time is assigned to suboptimal point B or C, the control QoP can be greatly improved. At the optimal point A, slight improvement is also achieved. This effect is called accurate adjustment in this paper, and in the best case, the control QoP can almost equal that of DDC as shown in performance region F.

**Remark 2.** Since the delay characteristics of Ethernet are random, nonperiodic, and boundless, Ethernet-based NCSs are notoriously difficult to control. There are few control methodologies for handling troublesome Ethernet delays. The NCS co-design methodology, however, can be successfully implemented with Ethernet for plant automation. One important point is that the NCS co-design framework can ensure reasonable performance (see Fig.11) even when the network-induced delays are longer than one sampling time (see Fig. 13; the maximum delay is about 100ms). Another advantage of the NCS co-design framework is that it can be easily applied to other networks, such as CAN and Token Bus, in which the delays are bounded and regular.
V. CONCLUSION

The methodologies for delay compensation in the networked control system can be categorized as network scheduling and controller design. However, little attention has been paid to combining the two. This paper has proposed a novel NCS design framework based on scheduling and control co-design to compensate for delays existing in networked control loops. The scheduling algorithm can guarantee the network QoS, while the advanced controller design can improve the control QoP. Network QoS, however, is in contradiction with control QoP. Hence, the NCS co-design framework achieves a trade-off between network QoS and control QoP. In this paper, Hong’s sampling time scheduling algorithm has been used to obtain the optimal sampling regions in a multi-loop NCS. Furthermore, the PI controller has been transformed into an adaptive fuzzy PI controller. Experimental results for an Ethernet-based multi-loop NCS show the power of the NCS co-design framework. Sampling time scheduling and the fuzzy PI controller are complementary; the former performs rough adjustment to ensure the network QoS, while the latter performs accurate adjustment to guarantee the control QoP. The disadvantages of the scheduling methods are overcome, and satisfactory performance is achieved with the NCS co-design framework.

REFERENCES

Liang Chen was born in Jiangsu Province, P.R. China, in 1981. He received his B.S. degree from the Department of Control Sci. & Eng., Zhejiang University, in 2004. He is now pursuing the Ph.D. in control theory & engineering in the National Lab. of Industrial Control Technology, Zhejiang University. His research interests include networked-based control systems, advanced modelling technologies and advanced process control in complex distillation processes.

Jianming Zhang was born in Zhejiang Province, P.R. China, in 1968. He received the B.S. degree in Industrial Automation from Wuhan Transportation University, in 1990, the M.S. degree in chemical engineering process mechanism from Zhejiang University of technology in 1996 and the Ph.D. degree in control theory & control engineering from Zhejiang University in 1999, respectively. He is currently an associate professor at Zhejiang University. His research interests include artificial neural network, fuzzy control theory, intelligent computing and their applications in industry.

Shuqing Wang was born in Zhejiang Province, P.R. China, in 1939. He received his B.S. degree from the Department of Chemical Engineering, Zhejiang University, in 1964. He is currently a professor in Zhejiang University. Professor Wang has studied the control theory, predictive control and chemical process control for almost 50 years. His currently interests include model predictive control, intelligent control and their applications in process industry. He is the Chairman of Modelling and Control of Bioprocess Committee, the member of Control Theory and Applications Committee Chinese Automatica Society, and the Chairman of Chemical Process Control of Zhejiang Province, Chemical Engineering Society.