Preprints of the 20th World Congress
The International Federation of Automatic Control
Toulouse, France, July 9-14, 2017

Scheduling and Control Co-Design for Control Systems under Computational Constraints ⋆

Yun-Bo Zhao * **Hui Dong** * **Hongjie Ni** *

* *College of Information Engineering, Zhejiang University of Technology, Hangzhou 310023, China*

Abstract: A prediction-based approach is proposed for control systems with limited and time-varying computational resources. The limited and time-varying computational resources can make the control system run in an open-loop fashion which may severely degrade the system performance or even destabilize the system. This issue is dealt with by producing more than one forward control predictions when abundant computational resources are available, and then using these forward control predictions to close the system when the computational resources are insufficient to calculate real-time control signal. This achievement is made without additional requirement for the computational resources and can be regarded as a useful completion of the scheduling algorithms. With a controller designed using a modified model predictive control method, the effectiveness of the proposed approach is successfully illustrated by a numerical example.

Keywords: Embedded control systems, Computational constraints, Scheduling, Co-design

1. INTRODUCTION

Embedded control systems (ECSs) are those control systems whose controllers are implemented using performance-limited embedded computational devices. Due to the rapid technological developments of the embedded computational devices, ECSs are now more and more popular in various areas of applications. Indeed, ECSs can be found in almost all industrial applications and, they are also the indispensable parts of the modern intelligent life, such as smart home, smart transportation, and so on (Graham and Kumar, 2003; Baillieul and Antsaklis, 2007; Marti et al., 2010). ECSs also provides one key foundation for the widely spreading Internet of Things which further marks its importance(Mattern and Floerkemeier, 2010; Atzori et al., 2010; Kranz et al., 2010; Zhou and Chao, 2011).

Despite all these existing applications and the bright future ahead, the underlying theoretical foundation of ECSs is not yet well developed if we look from, in particular, the control theory perspective. In fact, in most ECSs applications, the engineers are mainly concerned with the power consumption of the embedded computational devices (which are typically running on batteries) and the computational resource allocation of the embedded computational devices (which are often shared among multiple applications)(Law et al., 2009; Samii et al., 2009; Sgora et al., 2015; Xia and Sun, 2006; Curu and Sorel, 2004), while the optimal utilization of the allocated computational resources is barely touched from the control theory perspective. Due to the scheduling algorithms used to allocate the computational resources among multiple applications, the available computational resources for the considered ECS are subject to the needs of other applications and thus are not only limited but time-varying. This means that the available computational resources can be limited on some occasions while abundant on some other occasions, compared to the specific needs of the considered ECS. If we keep using conventional control algorithms without optimizing the utilization of the available computational resources, the above fact makes it possible that, either the performance of the ECS is severely degraded, or much more expensive embedded computational devices are compulsorily implemented even though unnecessary.

In order to take better use of the allocated computational resources, a prediction-based control and scheduling co-design approach to ECSs is proposed in this work. With the help of a relationship built between the amount of the available computational resources and the complexity of the controller that can be solved using these computational resources, we show that the limited and time-varying computational resources can be efficiently used to derive a better system performance, by an efficient treatment of the situation where the control system is left open-loop due to insufficient computational resources. The underlying idea of this achievement is based on a sequence of forward control predictions, calculated using the abundant computational resources and then stored and used afterwards when the computational resources are insufficient to calculate real-time control signal. This approach takes advantage of the available computational resources only, meaning that the approach improves the system performance without additional requirement for

⋆ This work was supported in part by the National Natural Science Foundation of China under Grant 61673350, in part by the Thousand Talents Plan of China and Zhejiang, and in part by the Major Projects Foundation of Zhejiang under Grant 2017C03060.
Corresponding author: Hongjie Ni.

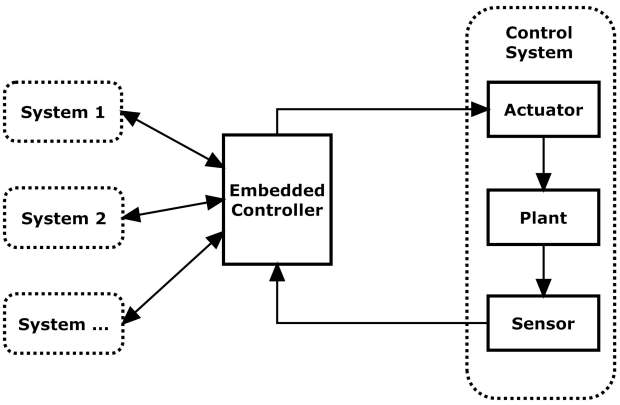

Fig. 1. Control systems with limited and time-varying computational resources.

the computational resources and thus is a useful completion of the existing ECSs configuration.

The remainder of the paper is organized as follows. We first formulate the problem of interest in Section 2, and then propose the prediction-based solution in Section 3. This solution is a general control framework for the considered problem which admits any appropriate controller design method to be used. As an example, a model predictive control (MPC) based controller is designed within this framework in Section 4. We then validate the proposed control framework and the MPC-based controller using a numerical example in Section 5. Section 6 concludes the paper.

2. PROBLEM FORMULATION

This paper considers a control system setting as illustrated in Fig. 1. The plant in the considered control system is described by a multi-input multi-output linear time-invariant system in discrete time, as follows,

$$x(k+1) = Ax(k) + Bu(k) \tag{1}$$

where $x(k) \in \mathbb{R}^n$, $u(k) \in \mathbb{R}^m$, $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$. The controller is implemented in an embedded device with limited computational resources. The use of performance-limited embedded devices can be out of the considerations of reducing the overall cost of implementation and maintenance, as is often seen in wireless sensor networks (Puccinelli and Haenggi, 2005). Even for such an embedded device with limited computational resources, it can still be shared with other applications which may not be solely for the control purposes, as shown in Fig. 1. This system setting implies that the computational resources allocated to the considered control system is: 1) limited due to the limited capability of the embedded device and 2) time varying in an unpredictable manner since the allocated computational resources are subject to the needs of other independent applications. Mathematically this fact can be formulated as follows,

$$R(k) \leq R^* \tag{2}$$

where $R(k)$ represents the computational resource available for the considered control system at time k and R^* is the total computational resource of the embedded computational device.

In order to calculate successfully the control signal $u(k)$ for the considered control system, some minimum computational resource is required, denoted by R_u^* and evidentally it is required that $R_u^* < R^*$. The time-varying feature of the available computational resources for the considered control system indicates the possibility, that under certain conditions the available computational resources for the considered control system can be too limited for the control signal to be successfully calculated, or, in other words, the control signal at time k can be calculated only if

$$R_u^* \leq R(k) \tag{3}$$

If the condition in (3) fails, the control signal $u(k)$ can not be calculated and thus unavailable to the actuator. In extreme cases, the control signals can possibly be absent to the actuator for so long a time that the system itself is destabilized, which can be absolutely forbidden in practical control systems.

Combining the conditions in (2) and (3), the computational resources required for the considered control system can be formulated as follows,

$$R_u^* \leq R(k) \leq R^*, \forall k \tag{4}$$

However, the time-varying feature of the available computational resources can result in a possible situation where for some k,

$$R(k) < R_u^* \leq R^* \tag{5}$$

meaning that at certain time instants the available computational resources are too limited to calculate the control signal, thus harming the system performance or even destabilize the system in extreme cases.

In order to deal with this issue, one possible solution is to apply certain type of scheduling algorithm to the computational resources. The scheduling policies can be predetermined offline, or adaptive to the current system condition while the system is up and running. In either case, the objective is to allocate the computational resources of the embedded device to all its occupiers efficiently, to ensure some determined performance indices(Walsh and Ye, 2001; Cervin, 2003; Zhao et al., 2008).

The use of scheduling algorithms can be effective in the efficient allocation of the computational resources. However, the reality is: 1) the computational resources subject to the constraint in (4) may simply impossible be scheduled under any scheduling algorithms, meaning that the situation in (5) inevitably occurs; and 2) the scheduling algorithm itself is essentially irrelevant to the specific usages of the allocated resources, meaning that the valuable computational resources, although allocated to the considered control system, may not be efficiently used. The latter implies that at certain time instants the available computational resources for the considered control system can be too much for the necessary requirement R_u^*, and thus most of them is wasted if conventional control methods are used, i.e., for some k, the following inequality is held,

$$R_u^* \ll R(k) \leq R^* \tag{6}$$

The above observations are illustrated in Fig. 2, where although the trajectory of the available computational resources in (a) is desirable, the reality may be more like that in (b): the available computational resources $R(k)$

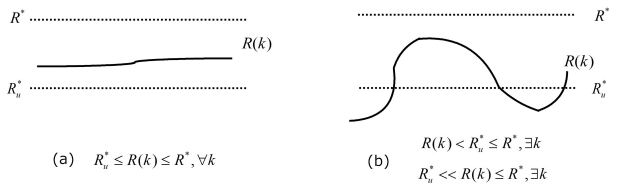

Fig. 2. Limited and time-varying computational resources. Although the trajectory in (a) is desirable, (b) might be more often seen in practice.

can either be too limited to perform the control signal calculation ($R(k) < R_u^*$) or too much such that most of them are wasted ($R(k) \gg R_u^*$).

In view of the above discussions on the limited and time-varying computational resources for the considered control system, the following problem thus naturally arises.

Problem: Given a control system as illustrated in Fig. 1 where the available computational resources are allocated by some scheduling algorithm and obey the constraints in (5) and (6) (illustrated in Fig. 2(b)), design the control strategy to take better use of the available computational resources to meet certain control objectives.

From the perspective mentioned earlier, the work presented here is thus a different view of the efficient use of the limited and time-varying computational resources and can be a useful completion of the scheduling algorithms. Indeed, we will assume in what follows that the computational resources of the embedded device have already been allocated to all the occupiers efficiently, and the focus of the work is only the efficient use of the available computational resources afterwards from, in particular, the control theory perspective.

3. PREDICTION-BASED CONTROL WITH LIMITED AND TIME-VARYING COMPUTATIONAL RESOURCES

In this section, we first discover a relationship between the length of the control predictions and the available computational resources. On the basis of this observation, a prediction-based control approach is proposed for control systems with limited and time-varying computational resources, resulting in an optimized utilization of the available computational resources and consequently a better system performance.

3.1 Further observation on $R(k)$

Before proceeding with the prediction-based approach to control systems with limited and time-varying computational resources, the following fact is first introduced (Bhattachary and Balas, 2004).

Fact 1. The computational resources required to calculate the control input (e.g., by solving an optimization problem) are strictly increasing as the number of variables to be determined increases.

In particular, this work is interested in the length of the control predictions that can be calculated using the available computational resources. Notice that in most cases the allocated computational resources for the considered control system can be measured by the allocated processor time. Without causing confusion, we use $R(k)$ itself to represent its processor time, and therefore $R(k) \in \mathbb{R}_0^+$ where $\mathbb{R}_0^+$ is the set of non-negative real numbers. If we denote the minimum computational resources required for calculating the control predictions with length i by R_i^* (thus $R_0^* = 0$ and $R_1^* = R_u^*$), the above fact implies that the following mapping from the available computational resources to the length of the control predictions, $f : \mathbb{R}_0^+ \to \mathbb{N}_0^+$, can be established,

$$f(R(k)) = N_k, \text{ if } R_{N_k}^* \le R(k) < R_{N_k+1}^* \tag{7}$$

where $\mathbb{N}_0^+$ is the set of non-negative integers. This definition makes sense because for any computational resource (or processor time), there is a unique non-negative integer corresponding to it by (7), due to the fact that the sequence $R_{N_k}^*$ is strictly increasing with N_k according to *Fact 1*, i.e., $R_{N_k}^* < R_{N_k+1}^*, \forall N_k \ge 0$. From the definition it is seen that the function $f(\cdot)$ is typically piecewise constant and continuous from the right. For simplicity of notations, in what follows we will use N_k in stead of $f(R(k))$ to denote the length of the control predictions that can be calculated using the computational resources $R(k)$.

3.2 Prediction-based approach to control systems with limited and time-varying computational resources

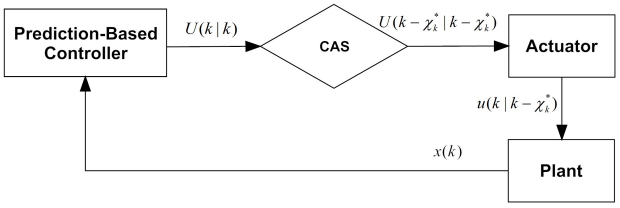

Fig. 3. Prediction-based control for control systems with limited and time-varying computational resources.

The mapping in (7) provides us with a different view on the available computational resources $R(k)$: It can be interpreted as the length of the control predictions that can be calculated using the available computational resources, i.e., N_k. In addition, by (5) and (6), it is readily concluded that: 1) $N_k = 0$ if (5) holds and more importantly, 2) $N_k \gg 1$ if (6) holds.

The first condition $N_k = 0$ is the source that fails conventional control approaches to control systems with limited and time-varying computational resources (which requires that $N_k \ge 1, \forall k$), while the latter places the foundation of the prediction-based control approach proposed in this work. The essence is that the condition, $N_k \gg 1$, makes it possible that more than 1 control predictions can be calculated at time k using the available computational resources.

The block diagram of the proposed approach is illustrated in Fig. 3. Different from conventional control methods where only one step control signal is calculated at any specific time instant, in this prediction-based control framework, the available computational resources are fully explored to calculate as many as possible control predictions which are then stored at the actuator side for future use.

The sequence of the control predictions, or forward control sequence (FCS), can be constructed at time k, as follows.

$$U(k|k) = [u(k|k)\ u(k+1|k)\ \dots\ u(k+N_k-1|k)] \quad (8)$$

where $u(k+i|k), i = 0, 1, \dots, N_k$ are the control predictions calculated based on the information at time k and it is noticed that the length of the FCS is typically time-varying, depending on the currently available computational resources, N_k.

The CAS in Fig. 3, or control action selector, is designed with a cache capable of storing one FCS and a simple logic to select from the FCS the appropriate control signal. The CAS is responsible for the following two functions:

- Update the cache whenever a new FCS arrives, in order that the FCS stored in the CAS is always up to date. The FCS stored at time k is not necessarily calculated at this very time instant due to (5). Denote, at time k at the actuator side, the number of the time steps during which the stored FCS has been in the cache by χ_k^*, then the FCS at time k was calculated at time $k-\chi_k^*$ and can be written as $U(k-\chi_k^*|k-\chi_k^*)$.
- Select the appropriate control signal from the FCS stored in its cache, in case that no control signal is calculated real time due to the insufficient computational resources. Using the above notations, the control signal at time k is selected from $U(k-\chi_k^*|k-\chi_k^*)$ and can be obtained as
$$u(k) = u(k|k-\chi_k^*) \quad (9)$$
Notice that this control signal $u(k|k-\chi_k^*)$, although delayed (based on the sensing information at time $k-\chi_k^*$), was particularly designed for this very time instant k, and therefore can effectively compensate for the delay provided that the FCS is properly designed.

In order that the control signal required by the actuator is always available from the CAS, the available computational resources are subject to certain constraints, as stated explicitly in the following proposition. This is also the basis on which Algorithm 1 can work.

Proposition 2. The control action selected by (9) is always available if and only if the following inequality is held,

$$N_{k-\chi_k^*} \geq \chi_k^* + 1, \forall k > \chi_k^* \quad (10)$$

Proof. It is noticed that the FCS used at time k at the actuator side, $U(k-\chi_k^*|k-\chi_k^*)$, was calculated at time $k-\chi_k^*$ at the controller side. In order that the control action selected by (9) is within $U(k-\chi_k^*|k-\chi_k^*)$, the length of this FCS, i.e., $N_{k-\chi_k^*}$, is required to be not less than $k-(k-\chi_k^*)+1$, which is exactly the condition stated in the proposition.

Remark 3. Consider the following two special cases of the condition stated in Proposition 2.

- $\chi_k^* \equiv 0$. This condition implies that the available computational resources for the considered control system, $R(k)$, is always sufficient to calculate at least a single step of the control signal, i.e., $N_k \geq 1, \forall k$. In this case conventional control methods can also work and thus this proposed approach includes conventional control methods as its special case.
- $\exists k, \chi_k^* > 0$. In this case the available computational resources for the considered control system, $R(k)$, is always insufficient to calculate a single step of the control signal from time $k-\chi_k^*+1$ to k, i.e., $N_i \equiv 0, k-\chi_k^*+1 \leq i \leq k$. Conventional control methods will not work in this case while the proposed prediction-based approach deals successfully with this issue by using previously predicted control signal, $u(k|k-\chi_k^*)$.

The algorithm of the prediction-based control approach can now be organized as follows, the block diagram of which is illustrated in Fig. 3.

Algorithm 1. (The prediction-based control approach).

S1. The sensor samples the plant output and sends it to the controller;

S2. Based on the allocated computational resources $R(k)$, the prediction-based controller determines the length of the FCS, N_k by (7), produces the required FCS by (8) and sends it to the actuator side.

S3. The CAS updates its cache accordingly and select the appropriate control signal by (9) and applies it to the plant.

Remark 4. It is noticed that the delay caused by the processor time used to calculate the FCS is not taken into account in the above algorithm. This makes sense because this time interval is usually sufficiently small and thus can be ignored without affecting the system behavior too much. However, it is worth pointing out that if absolutely necessary, this delay can be integrated into the waiting delay at the actuator side, χ_k^*, and therefore can be efficiently treated without modifying significantly the proposed control structure.

4. MPC-BASED CONTROLLER REALIZATION

The prediction-based control approach proposed in the last section for control systems with limited and time-varying computational resources is a general control framework for this system setting, in the sense that it admits any appropriate controller design method to produce the FCS provided it can give rise to a desirable system performance. As an example, an MPC-based method is proposed to design the FCS in this section.

Classic MPC is an optimal control strategy with finite horizon: it optimizes the trajectory of the considered system to a certain length by solving an optimization problem involving finite inputs and outputs (or states) predictions of the system , and then the first control input is applied to the control system while others are discarded. The optimization problem is solved at every time step, making it possible to deal with noises, uncertainties and constraints at an affordable cost.

The objective function of MPC is typically designed as follows,

$$J_k(N) = X^T(k|k)Q(N)X(k|k) + U^T(k|k)R(N)U(k|k) \quad (11)$$

where $J_k(N)$ is the objective function at time k, $U(k|k) = [u(k|k)\ \dots\ u(k+N-1|k)]^T$ as in (8) is the control predictions to be determined, $X(k|k) = [x(k+1|k)\ \dots\ x(k+N|k)]^T$ is the predictive state trajectory, $Q(N)$ and $R(N)$ are weighting matrixes with appropriate dimensions and N is the prediction horizon.

It is noticed that the computational complexity of the optimization problem with the objective function in (11) is mainly determined by the prediction horizon N. Denote the minimum computational resources required to solve the optimization problem with a prediction horizon of N by R_{J_N}, the definition of function f in (7) in correspondence to the objective function in (11) can then be specified as follows,

$$f(R(k)) = N_k, \text{ if } R_{J_{N_k}} \leq R(k) < R_{J_{N_k+1}} \tag{12}$$

In what follows N_k obtained by (12) will be used to represent the available computational resources $R(k)$.

The specified function $f(\cdot)$ in (12) implies that an optimization problem with the objective function in (11) and $N = N_k$ can be solved at time k using the available computational resources. In order to solve this optimization problem, the predictive states at time k can be firstly obtained by iteration for $j = 1, 2, \cdots, N_k$, as follows

$$x(k+j|k) = A^j x(k) + \sum_{l=0}^{j-1} A^{j-l-1} B u(k+l|k)$$

and hence the predictive states in the vector form can be constructed as

$$X(k|k) = E(N_k)x(k) + F(N_k)U(k|k) \tag{13}$$

where $E(N_k) = [A^T \cdots (A^{N_k})^T]^T$ and $F(N_k)$ is a $N_k \times N_k$ block lower triangular matrix with its non-null elements defined by $F(N_k)_{ij} = A^{i-j}B$, $j \leq i$.

The optimal FCS can then be calculated by substituting (13) to (11) and minimizing $J_k(N_k)$, which turns out to be state feedback control, as follows,

$$U(k|k) = K(N_k)x(k) \tag{14}$$

where $K(N_k) = -(F^T(N_k)Q(N_k)F(N_k) + R(N_k))^{-1} \times F^T(N_k)Q(N_k)E(N_k)$.

The algorithm of the prediction-based control approach with the use of MPC can then be specified as follows.

Algorithm 2. (Prediction-based control with MPC).

S1. The sensor samples the plant output and sends it to the controller;

S2. Based on the allocated computational resources $R(k)$, the prediction-based controller determines the length of the FCS, N_k by (12), solves the MPC optimization problem, obtains the FCS by (14) and sends it to the actuator side.

S3. The CAS updates its cache accordingly and select the appropriate control signal by (9) and applies it to the plant.

Remark 5. The modified MPC algorithm in this section is distinct from conventional MPC algorithms in two aspects. Firstly, unlike conventional MPC algorithms, the prediction horizon N_k in Algorithm 2 is time-varying and determined by the available computational resources $R(k)$. Secondly, conventional MPC algorithm uses only the first control input obtained by minimizing the objective function in (11) while Algorithm 2 takes advantage of the full information of the control predictions.

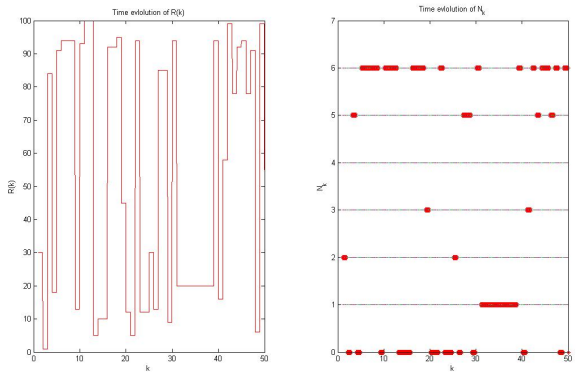

Fig. 4. The time evolution of $R(k)$ and N_k.

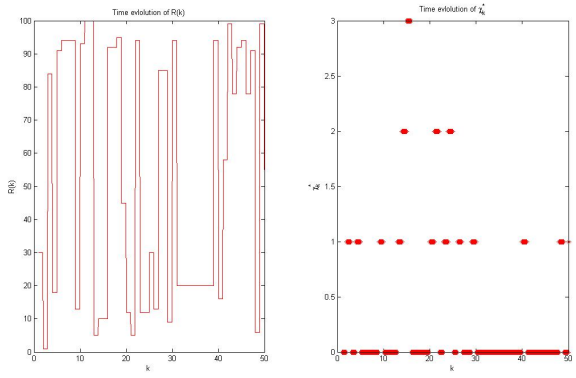

Fig. 5. The time evolution of $R(k)$ and χ_k^*.

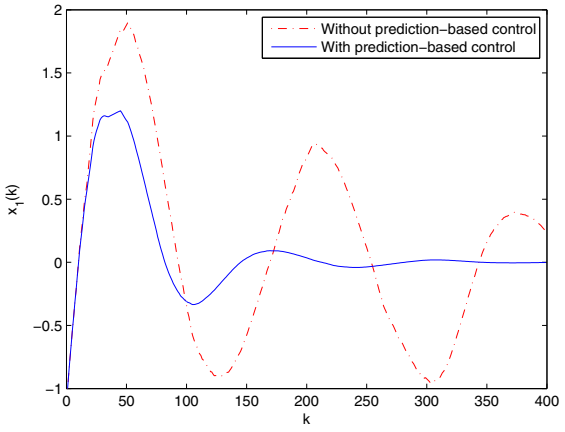

Fig. 6. Comparison of the state responses.

5. NUMERICAL EXAMPLE

Consider the system in (1) with the following system matrices, which is seen to be open-loop unstable,

$$A = \begin{pmatrix} 0.99 & 0.12 \\ 0 & 1.04 \end{pmatrix}, B = \begin{pmatrix} 0.04 \\ 0.1 \end{pmatrix}$$

Suppose the controller is implemented within an embedded computational device and the available computational resources for the considered control system are allocated by certain scheduling algorithm. The levels of the available computational resources $R(k)$ are assumed to be integers,

within the range from 1 to 100, i.e., $R^* = 100$. We also assume that the minimum computational resources required for calculating $1-7$ steps of the control signal are

$$[R_1^* \ R_2^* \ R_3^* \ R_4^* \ R_5^* \ R_6^* \ R_7^*] = [20 \ 30 \ 45 \ 60 \ 75 \ 90 \ 105] \tag{15}$$

which implies that $R_u^* = 20$ and function $f(\cdot)$ in (12) can then be defined accordingly.

$R(k)$ is assumed to be a Markovian process, with higher probabilities of being below R_u^* (5) or close to R^* (6). In practice, the allocated computational resources at the next time instant are likely to stay at the same level with a high probability, or, in other words, additional forces are required for the allocated computational resources to switch between different levels. This fact is considered in the generation of $R(k)$ by assigning a higher probability of self-transition in the transition probability matrix.

A typical time evolution of $R(k)$ can be found in Fig. 4 or Fig. 5. With (15) the time evolution of the length of the FCSs, N_k, can be determined in correspondence to $R(k)$, as illustrated in Fig. 4. In addition, it is also seen from the time evolution of χ_k^* in Fig. 5 that the available computational resources are often under the minimum requirement of calculating a signal step of the control signal.

In order to verify the effectiveness of the proposed prediction-based control approach, we compare the time evolution of the system states with the same controller design method as proposed in Section 4, but one implemented within the prediction-based control framework as discussed in Section 3 and the other without it. That is, the former uses the predicted control signal from the FCS while the latter uses zero control, in the case that the available computational resources are below the minimum requirement for calculating a single step of the control signal, i.e., $R(k) < R_u^*$.

The time evolution of the system states in Fig. 6 clearly supports our previous statement: without consuming additional computational resources, the prediction-based control approach can give rise to a better system performance (a rapidly stable trajectory) than conventional control method (a highly oscillating trajectory).

6. CONCLUSION

By redesigning the control structure, we show that the system performance of control systems subject to limited and time-varying computational resources can be improved significantly without requiring additional computational resources. This achievement is made based on the better use of the available computational resources and can be readily implemented since it does not affect the existing scheduling algorithms. In this sense the proposed approach can be regarded as a useful completion of the scheduling algorithms used in embedded control systems and is of importance in the practical implementation of embedded control systems.

REFERENCES

Atzori, L., Iera, A., and Morabito, G. (2010). The Internet of things: A survey. *Computer Networks*, 54(15), 2787 – 2805.

Baillieul, J. and Antsaklis, P.J. (2007). Control and communcation challenges in networked real-time systems. *IEEE Proc.*, 95(1), 9–28.

Bhattachary, R. and Balas, G. (2004). Anytime control algorithm: Model reduction approach. *J. Guid. Control Dyn.*, 27(5), 767–776.

Cervin, A. (2003). *Integrated Control and Real Time Scheduling*. Ph.D. thesis, Department of Automatic Control, Lund Institute of Technology, Lund, Sweden.

Curu, L. and Sorel, Y. (2004). Non-preemptive scheduling algorithms and schedulability conditions for real-time systems with precedence and latency constraints. Technical report, INRIA.

Graham, S. and Kumar, P.R. (2003). The convergence of control, communication and computation. In M. Conti, S. Giordano, E. Gregori, and S. Olariu (eds.), *Personal Wireless Communication*, 458–475. Springer-Verlag, Heidelberg.

Kranz, M., Holleis, P., and Schmidt, A. (2010). Embedded interaction: Interacting with the Internet of Things. *IEEE Internet Comput.*, 14(2), 46 –53.

Law, Y., Palaniswami, M., Hoesel, L., Doumen, J., Hartel, P., and Havinga, P. (2009). Energy-efficient link-layer jamming attacks against wireless sensor network MAC protocols. *ACM Trans. Sens. Netw.*, 5(1), 1–38.

Marti, P., Velasco, M., Fuertes, J., Camacho, A., and Buttazzo, G. (2010). Design of an embedded control system laboratory experiment. *IEEE Trans. Ind. Electron.*, 57(10), 3297–3307.

Mattern, F. and Floerkemeier, C. (2010). From the internet of computers to the internet of things. In K. Sachs, I. Petrov, and P. Guerrero (eds.), *From Active Data Management to Event-Based Systems and More*, volume 6462 of *Lecture Notes in Computer Science*, 242–259. Springer Berlin Heidelberg.

Puccinelli, D. and Haenggi, M. (2005). Wireless sensor networks: Applications and challenges of ubiquitous sensing. *IEEE Circuits Syst. Mag.*, 5(3), 19–31.

Samii, S., Cervin, A., Eles, P., and Peng, Z. (2009). Integrated scheduling and synthesis of control applications on distributed embedded systems. In *Design, Automation Test in Europe Conference Exhibition (DATE '09)*, 57 –62.

Sgora, A., Vergados, D.J., and Vergados, D.D. (2015). A survey of tdma scheduling schemes in wireless multihop networks. volume 47, 1–39.

Walsh, G.C. and Ye, H. (2001). Scheduling of networked control systems. *IEEE Control Syst. Mag.*, 21(1), 57–65.

Xia, F. and Sun, Y. (2006). Control-scheduling codesign: A perspective on integrating control and computing.

Zhao, Y.B., Liu, G.P., and Rees, D. (2008). Integrated predictive control and scheduling co-design for networked control systems. *IET Control Theory Appl.*, 2(1), 7–15.

Zhou, L. and Chao, H.C. (2011). Multimedia traffic security architecture for the internet of things. *IEEE Netw.*, 25(3), 35 –40.

Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 50-1 (2017) 5881–5886

Scheduling and Control Co-Design for Control Systems under Computational Constraints ⋆

Yun-Bo Zhao * **Hui Dong** * **Hongjie Ni** *

* *College of Information Engineering, Zhejiang University of Technology, Hangzhou 310023, China*

Abstract: A prediction-based approach is proposed for control systems with limited and time-varying computational resources. The limited and time-varying computational resources can make the control system run in an open-loop fashion which may severely degrade the system performance or even destabilize the system. This issue is dealt with by producing more than one forward control predictions when abundant computational resources are available, and then using these forward control predictions to close the system when the computational resources are insufficient to calculate real-time control signal. This achievement is made without additional requirement for the computational resources and can be regarded as a useful completion of the scheduling algorithms. With a controller designed using a modified model predictive control method, the effectiveness of the proposed approach is successfully illustrated by a numerical example.

Keywords: Embedded control systems, Computational constraints, Scheduling, Co-design

1. INTRODUCTION

Embedded control systems (ECSs) are those control systems whose controllers are implemented using performance-limited embedded computational devices. Due to the rapid technological developments of the embedded computational devices, ECSs are now more and more popular in various areas of applications. Indeed, ECSs can be found in almost all industrial applications and, they are also the indispensable parts of the modern intelligent life, such as smart home, smart transportation, and so on (Graham and Kumar, 2003; Baillieul and Antsaklis, 2007; Marti et al., 2010). ECSs also provides one key foundation for the widely spreading Internet of Things which further marks its importance(Mattern and Floerkemeier, 2010; Atzori et al., 2010; Kranz et al., 2010; Zhou and Chao, 2011).

Despite all these existing applications and the bright future ahead, the underlying theoretical foundation of ECSs is not yet well developed if we look from, in particular, the control theory perspective. In fact, in most ECSs applications, the engineers are mainly concerned with the power consumption of the embedded computational devices (which are typically running on batteries) and the computational resource allocation of the embedded computational devices (which are often shared among multiple applications)(Law et al., 2009; Samii et al., 2009; Sgora et al., 2015; Xia and Sun, 2006; Curu and Sorel, 2004), while the optimal utilization of the allocated computational resources is barely touched from the control theory perspective. Due to the scheduling algorithms used to allocate the computational resources among multiple applications, the available computational resources for the considered ECS are subject to the needs of other applications and thus are not only limited but time-varying. This means that the available computational resources can be limited on some occasions while abundant on some other occasions, compared to the specific needs of the considered ECS. If we keep using conventional control algorithms without optimizing the utilization of the available computational resources, the above fact makes it possible that, either the performance of the ECS is severely degraded, or much more expensive embedded computational devices are compulsorily implemented even though unnecessary.

In order to take better use of the allocated computational resources, a prediction-based control and scheduling co-design approach to ECSs is proposed in this work. With the help of a relationship built between the amount of the available computational resources and the complexity of the controller that can be solved using these computational resources, we show that the limited and time-varying computational resources can be efficiently used to derive a better system performance, by an efficient treatment of the situation where the control system is left open-loop due to insufficient computational resources. The underlying idea of this achievement is based on a sequence of forward control predictions, calculated using the abundant computational resources and then stored and used afterwards when the computational resources are insufficient to calculate real-time control signal. This approach takes advantage of the available computational resources only, meaning that the approach improves the system performance without additional requirement for

⋆ This work was supported in part by the National Natural Science Foundation of China under Grant 61673350, in part by the Thousand Talents Plan of China and Zhejiang, and in part by the Major Projects Foundation of Zhejiang under Grant 2017C03060.
Corresponding author: Hongjie Ni.

10.1016/j.ifacol.2017.08.1305

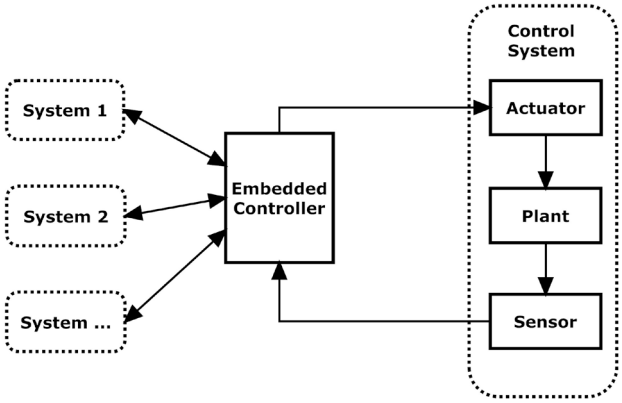

Fig. 1. Control systems with limited and time-varying computational resources.

the computational resources and thus is a useful completion of the existing ECSs configuration.

The remainder of the paper is organized as follows. We first formulate the problem of interest in Section 2, and then propose the prediction-based solution in Section 3. This solution is a general control framework for the considered problem which admits any appropriate controller design method to be used. As an example, a model predictive control (MPC) based controller is designed within this framework in Section 4. We then validate the proposed control framework and the MPC-based controller using a numerical example in Section 5. Section 6 concludes the paper.

2. PROBLEM FORMULATION

This paper considers a control system setting as illustrated in Fig. 1. The plant in the considered control system is described by a multi-input multi-output linear time-invariant system in discrete time, as follows,

$$x(k+1) = Ax(k) + Bu(k) \tag{1}$$

where $x(k) \in \mathbb{R}^n$, $u(k) \in \mathbb{R}^m$, $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$. The controller is implemented in an embedded device with limited computational resources. The use of performance-limited embedded devices can be out of the considerations of reducing the overall cost of implementation and maintenance, as is often seen in wireless sensor networks (Puccinelli and Haenggi, 2005). Even for such an embedded device with limited computational resources, it can still be shared with other applications which may not be solely for the control purposes, as shown in Fig. 1. This system setting implies that the computational resources allocated to the considered control system is: 1) limited due to the limited capability of the embedded device and 2) time varying in an unpredictable manner since the allocated computational resources are subject to the needs of other independent applications. Mathematically this fact can be formulated as follows,

$$R(k) \leq R^* \tag{2}$$

where $R(k)$ represents the computational resource available for the considered control system at time k and R^* is the total computational resource of the embedded computational device.

In order to calculate successfully the control signal $u(k)$ for the considered control system, some minimum computational resource is required, denoted by R_u^* and evidentally it is required that $R_u^* < R^*$. The time-varying feature of the available computational resources for the considered control system indicates the possibility, that under certain conditions the available computational resources for the considered control system can be too limited for the control signal to be successfully calculated, or, in other words, the control signal at time k can be calculated only if

$$R_u^* \leq R(k) \tag{3}$$

If the condition in (3) fails, the control signal $u(k)$ can not be calculated and thus unavailable to the actuator. In extreme cases, the control signals can possibly be absent to the actuator for so long a time that the system itself is destabilized, which can be absolutely forbidden in practical control systems.

Combining the conditions in (2) and (3), the computational resources required for the considered control system can be formulated as follows,

$$R_u^* \leq R(k) \leq R^*, \forall k \tag{4}$$

However, the time-varying feature of the available computational resources can result in a possible situation where for some k,

$$R(k) < R_u^* \leq R^* \tag{5}$$

meaning that at certain time instants the available computational resources are too limited to calculate the control signal, thus harming the system performance or even destabilize the system in extreme cases.

In order to deal with this issue, one possible solution is to apply certain type of scheduling algorithm to the computational resources. The scheduling policies can be predetermined offline, or adaptive to the current system condition while the system is up and running. In either case, the objective is to allocate the computational resources of the embedded device to all its occupiers efficiently, to ensure some determined performance indices(Walsh and Ye, 2001; Cervin, 2003; Zhao et al., 2008).

The use of scheduling algorithms can be effective in the efficient allocation of the computational resources. However, the reality is: 1) the computational resources subject to the constraint in (4) may simply impossible be scheduled under any scheduling algorithms, meaning that the situation in (5) inevitably occurs; and 2) the scheduling algorithm itself is essentially irrelevant to the specific usages of the allocated resources, meaning that the valuable computational resources, although allocated to the considered control system, may not be efficiently used. The latter implies that at certain time instants the available computational resources for the considered control system can be too much for the necessary requirement R_u^*, and thus most of them is wasted if conventional control methods are used, i.e., for some k, the following inequality is held,

$$R_u^* \ll R(k) \leq R^* \tag{6}$$

The above observations are illustrated in Fig. 2, where although the trajectory of the available computational resources in (a) is desirable, the reality may be more like that in (b): the available computational resources $R(k)$

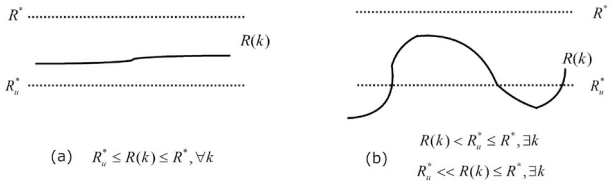

Fig. 2. Limited and time-varying computational resources. Although the trajectory in (a) is desirable, (b) might be more often seen in practice.

can either be too limited to perform the control signal calculation ($R(k) < R_u^*$) or too much such that most of them are wasted ($R(k) \gg R_u^*$).

In view of the above discussions on the limited and time-varying computational resources for the considered control system, the following problem thus naturally arises.

Problem: Given a control system as illustrated in Fig. 1 where the available computational resources are allocated by some scheduling algorithm and obey the constraints in (5) and (6) (illustrated in Fig. 2(b)), design the control strategy to take better use of the available computational resources to meet certain control objectives.

From the perspective mentioned earlier, the work presented here is thus a different view of the efficient use of the limited and time-varying computational resources and can be a useful completion of the scheduling algorithms. Indeed, we will assume in what follows that the computational resources of the embedded device have already been allocated to all the occupiers efficiently, and the focus of the work is only the efficient use of the available computational resources afterwards from, in particular, the control theory perspective.

3. PREDICTION-BASED CONTROL WITH LIMITED AND TIME-VARYING COMPUTATIONAL RESOURCES

In this section, we first discover a relationship between the length of the control predictions and the available computational resources. On the basis of this observation, a prediction-based control approach is proposed for control systems with limited and time-varying computational resources, resulting in an optimized utilization of the available computational resources and consequently a better system performance.

3.1 Further observation on $R(k)$

Before proceeding with the prediction-based approach to control systems with limited and time-varying computational resources, the following fact is first introduced (Bhattachary and Balas, 2004).

Fact 1. The computational resources required to calculate the control input (e.g., by solving an optimization problem) are strictly increasing as the number of variables to be determined increases.

In particular, this work is interested in the length of the control predictions that can be calculated using the available computational resources. Notice that in most cases the allocated computational resources for the considered control system can be measured by the allocated processor time. Without causing confusion, we use $R(k)$ itself to represent its processor time, and therefore $R(k) \in \mathbb{R}_0^+$ where $\mathbb{R}_0^+$ is the set of non-negative real numbers. If we denote the minimum computational resources required for calculating the control predictions with length i by R_i^* (thus $R_0^* = 0$ and $R_1^* = R_u^*$), the above fact implies that the following mapping from the available computational resources to the length of the control predictions, $f : \mathbb{R}_0^+ \to \mathbb{N}_0^+$, can be established,

$$f(R(k)) = N_k, \text{ if } R_{N_k}^* \leq R(k) < R_{N_k+1}^* \tag{7}$$

where $\mathbb{N}_0^+$ is the set of non-negative integers. This definition makes sense because for any computational resource (or processor time), there is a unique non-negative integer corresponding to it by (7), due to the fact that the sequence $R_{N_k}^*$ is strictly increasing with N_k according to *Fact 1*, i.e., $R_{N_k}^* < R_{N_k+1}^*, \forall N_k \geq 0$. From the definition it is seen that the function $f(\cdot)$ is typically piecewise constant and continuous from the right. For simplicity of notations, in what follows we will use N_k in stead of $f(R(k))$ to denote the length of the control predictions that can be calculated using the computational resources $R(k)$.

3.2 Prediction-based approach to control systems with limited and time-varying computational resources

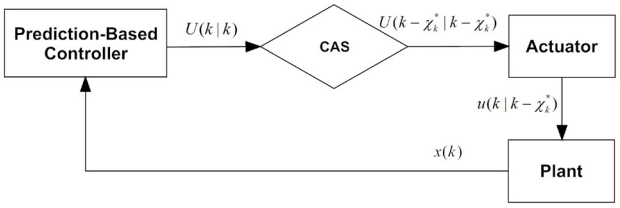

Fig. 3. Prediction-based control for control systems with limited and time-varying computational resources.

The mapping in (7) provides us with a different view on the available computational resources $R(k)$: It can be interpreted as the length of the control predictions that can be calculated using the available computational resources, i.e., N_k. In addition, by (5) and (6), it is readily concluded that: 1) $N_k = 0$ if (5) holds and more importantly, 2) $N_k \gg 1$ if (6) holds.

The first condition $N_k = 0$ is the source that fails conventional control approaches to control systems with limited and time-varying computational resources (which requires that $N_k \geq 1, \forall k$), while the latter places the foundation of the prediction-based control approach proposed in this work. The essence is that the condition, $N_k \gg 1$, makes it possible that more than 1 control predictions can be calculated at time k using the available computational resources.

The block diagram of the proposed approach is illustrated in Fig. 3. Different from conventional control methods where only one step control signal is calculated at any specific time instant, in this prediction-based control framework, the available computational resources are fully explored to calculate as many as possible control predictions which are then stored at the actuator side for future use.

The sequence of the control predictions, or forward control sequence (FCS), can be constructed at time k, as follows.

$$U(k|k) = [u(k|k)\ u(k+1|k)\ \dots\ u(k+N_k-1|k)] \quad (8)$$

where $u(k+i|k), i = 0, 1, \dots, N_k$ are the control predictions calculated based on the information at time k and it is noticed that the length of the FCS is typically time-varying, depending on the currently available computational resources, N_k.

The CAS in Fig. 3, or control action selector, is designed with a cache capable of storing one FCS and a simple logic to select from the FCS the appropriate control signal. The CAS is responsible for the following two functions:

- Update the cache whenever a new FCS arrives, in order that the FCS stored in the CAS is always up to date. The FCS stored at time k is not necessarily calculated at this very time instant due to (5). Denote, at time k at the actuator side, the number of the time steps during which the stored FCS has been in the cache by χ_k^*, then the FCS at time k was calculated at time $k-\chi_k^*$ and can be written as $U(k-\chi_k^*|k-\chi_k^*)$.
- Select the appropriate control signal from the FCS stored in its cache, in case that no control signal is calculated real time due to the insufficient computational resources. Using the above notations, the control signal at time k is selected from $U(k-\chi_k^*|k-\chi_k^*)$ and can be obtained as
$$u(k) = u(k|k-\chi_k^*) \quad (9)$$
Notice that this control signal $u(k|k-\chi_k^*)$, although delayed (based on the sensing information at time $k-\chi_k^*$), was particularly designed for this very time instant k, and therefore can effectively compensate for the delay provided that the FCS is properly designed.

In order that the control signal required by the actuator is always available from the CAS, the available computational resources are subject to certain constraints, as stated explicitly in the following proposition. This is also the basis on which Algorithm 1 can work.

Proposition 2. The control action selected by (9) is always available if and only if the following inequality is held,

$$N_{k-\chi_k^*} \geq \chi_k^* + 1, \forall k > \chi_k^* \quad (10)$$

Proof. It is noticed that the FCS used at time k at the actuator side, $U(k-\chi_k^*|k-\chi_k^*)$, was calculated at time $k-\chi_k^*$ at the controller side. In order that the control action selected by (9) is within $U(k-\chi_k^*|k-\chi_k^*)$, the length of this FCS, i.e., $N_{k-\chi_k^*}$, is required to be not less than $k-(k-\chi_k^*)+1$, which is exactly the condition stated in the proposition.

Remark 3. Consider the following two special cases of the condition stated in Proposition 2.

- $\chi_k^* \equiv 0$. This condition implies that the available computational resources for the considered control system, $R(k)$, is always sufficient to calculate at least a single step of the control signal, i.e., $N_k \geq 1, \forall k$. In this case conventional control methods can also work and thus this proposed approach includes conventional control methods as its special case.
- $\exists k, \chi_k^* > 0$. In this case the available computational resources for the considered control system, $R(k)$, is always insufficient to calculate a single step of the control signal from time $k-\chi_k^*+1$ to k, i.e., $N_i \equiv 0, k-\chi_k^*+1 \leq i \leq k$. Conventional control methods will not work in this case while the proposed prediction-based approach deals successfully with this issue by using previously predicted control signal, $u(k|k-\chi_k^*)$.

The algorithm of the prediction-based control approach can now be organized as follows, the block diagram of which is illustrated in Fig. 3.

Algorithm 1. (The prediction-based control approach).

S1. The sensor samples the plant output and sends it to the controller;

S2. Based on the allocated computational resources $R(k)$, the prediction-based controller determines the length of the FCS, N_k by (7), produces the required FCS by (8) and sends it to the actuator side.

S3. The CAS updates its cache accordingly and select the appropriate control signal by (9) and applies it to the plant.

Remark 4. It is noticed that the delay caused by the processor time used to calculate the FCS is not taken into account in the above algorithm. This makes sense because this time interval is usually sufficiently small and thus can be ignored without affecting the system behavior too much. However, it is worth pointing out that if absolutely necessary, this delay can be integrated into the waiting delay at the actuator side, χ_k^*, and therefore can be efficiently treated without modifying significantly the proposed control structure.

4. MPC-BASED CONTROLLER REALIZATION

The prediction-based control approach proposed in the last section for control systems with limited and time-varying computational resources is a general control framework for this system setting, in the sense that it admits any appropriate controller design method to produce the FCS provided it can give rise to a desirable system performance. As an example, an MPC-based method is proposed to design the FCS in this section.

Classic MPC is an optimal control strategy with finite horizon: it optimizes the trajectory of the considered system to a certain length by solving an optimization problem involving finite inputs and outputs (or states) predictions of the system , and then the first control input is applied to the control system while others are discarded. The optimization problem is solved at every time step, making it possible to deal with noises, uncertainties and constraints at an affordable cost.

The objective function of MPC is typically designed as follows,

$$J_k(N) = X^T(k|k)Q(N)X(k|k) + U^T(k|k)R(N)U(k|k) \quad (11)$$

where $J_k(N)$ is the objective function at time k, $U(k|k) = [u(k|k)\ \dots\ u(k+N-1|k)]^T$ as in (8) is the control predictions to be determined, $X(k|k) = [x(k+1|k)\ \dots\ x(k+N|k)]^T$ is the predictive state trajectory, $Q(N)$ and $R(N)$ are weighting matrixes with appropriate dimensions and N is the prediction horizon.

It is noticed that the computational complexity of the optimization problem with the objective function in (11) is mainly determined by the prediction horizon N. Denote the minimum computational resources required to solve the optimization problem with a prediction horizon of N by R_{J_N}, the definition of function f in (7) in correspondence to the objective function in (11) can then be specified as follows,

$$f(R(k)) = N_k, \text{ if } R_{J_{N_k}} \leq R(k) < R_{J_{N_k+1}} \tag{12}$$

In what follows N_k obtained by (12) will be used to represent the available computational resources $R(k)$.

The specified function $f(\cdot)$ in (12) implies that an optimization problem with the objective function in (11) and $N = N_k$ can be solved at time k using the available computational resources. In order to solve this optimization problem, the predictive states at time k can be firstly obtained by iteration for $j = 1, 2, \cdots, N_k$, as follows

$$x(k+j|k) = A^j x(k) + \sum_{l=0}^{j-1} A^{j-l-1} B u(k+l|k)$$

and hence the predictive states in the vector form can be constructed as

$$X(k|k) = E(N_k)x(k) + F(N_k)U(k|k) \tag{13}$$

where $E(N_k) = [A^T \cdots (A^{N_k})^T]^T$ and $F(N_k)$ is a $N_k \times N_k$ block lower triangular matrix with its non-null elements defined by $F(N_k)_{ij} = A^{i-j}B$, $j \leq i$.

The optimal FCS can then be calculated by substituting (13) to (11) and minimizing $J_k(N_k)$, which turns out to be state feedback control, as follows,

$$U(k|k) = K(N_k)x(k) \tag{14}$$

where $K(N_k) = -(F^T(N_k)Q(N_k)F(N_k) + R(N_k))^{-1} \times F^T(N_k)Q(N_k)E(N_k)$.

The algorithm of the prediction-based control approach with the use of MPC can then be specified as follows.

Algorithm 2. (Prediction-based control with MPC).

S1. The sensor samples the plant output and sends it to the controller;

S2. Based on the allocated computational resources $R(k)$, the prediction-based controller determines the length of the FCS, N_k by (12), solves the MPC optimization problem, obtains the FCS by (14) and sends it to the actuator side.

S3. The CAS updates its cache accordingly and select the appropriate control signal by (9) and applies it to the plant.

Remark 5. The modified MPC algorithm in this section is distinct from conventional MPC algorithms in two aspects. Firstly, unlike conventional MPC algorithms, the prediction horizon N_k in Algorithm 2 is time-varying and determined by the available computational resources $R(k)$. Secondly, conventional MPC algorithm uses only the first control input obtained by minimizing the objective function in (11) while Algorithm 2 takes advantage of the full information of the control predictions.

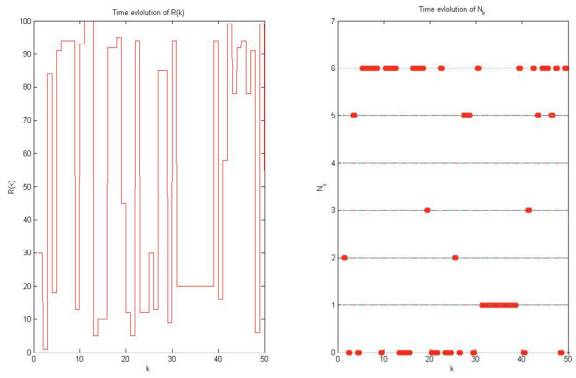

Fig. 4. The time evolution of $R(k)$ and N_k.

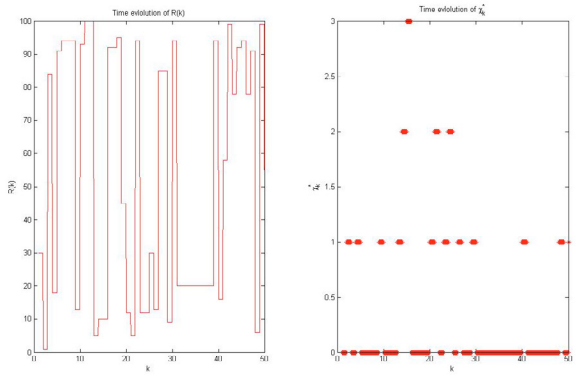

Fig. 5. The time evolution of $R(k)$ and χ_k^*.

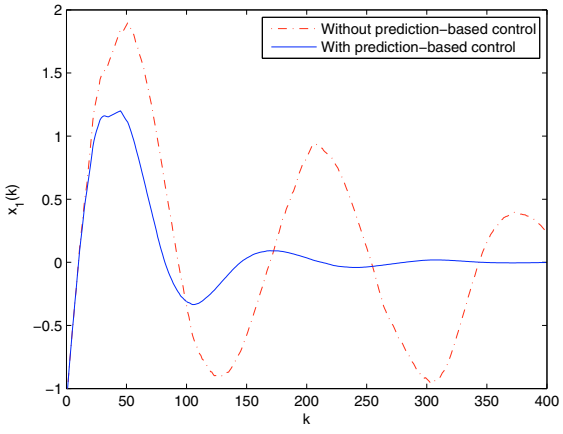

Fig. 6. Comparison of the state responses.

5. NUMERICAL EXAMPLE

Consider the system in (1) with the following system matrices, which is seen to be open-loop unstable,

$$A = \begin{pmatrix} 0.99 & 0.12 \\ 0 & 1.04 \end{pmatrix}, B = \begin{pmatrix} 0.04 \\ 0.1 \end{pmatrix}$$

Suppose the controller is implemented within an embedded computational device and the available computational resources for the considered control system are allocated by certain scheduling algorithm. The levels of the available computational resources $R(k)$ are assumed to be integers,

within the range from 1 to 100, i.e., $R^* = 100$. We also assume that the minimum computational resources required for calculating $1-7$ steps of the control signal are

$$[R_1^*\ R_2^*\ R_3^*\ R_4^*\ R_5^*\ R_6^*\ R_7^*] = [20\ 30\ 45\ 60\ 75\ 90\ 105] \tag{15}$$

which implies that $R_u^* = 20$ and function $f(\cdot)$ in (12) can then be defined accordingly.

$R(k)$ is assumed to be a Markovian process, with higher probabilities of being below R_u^* (5) or close to R^* (6). In practice, the allocated computational resources at the next time instant are likely to stay at the same level with a high probability, or, in other words, additional forces are required for the allocated computational resources to switch between different levels. This fact is considered in the generation of $R(k)$ by assigning a higher probability of self-transition in the transition probability matrix.

A typical time evolution of $R(k)$ can be found in Fig. 4 or Fig. 5. With (15) the time evolution of the length of the FCSs, N_k, can be determined in correspondence to $R(k)$, as illustrated in Fig. 4. In addition, it is also seen from the time evolution of χ_k^* in Fig. 5 that the available computational resources are often under the minimum requirement of calculating a signal step of the control signal.

In order to verify the effectiveness of the proposed prediction-based control approach, we compare the time evolution of the system states with the same controller design method as proposed in Section 4, but one implemented within the prediction-based control framework as discussed in Section 3 and the other without it. That is, the former uses the predicted control signal from the FCS while the latter uses zero control, in the case that the available computational resources are below the minimum requirement for calculating a single step of the control signal, i.e., $R(k) < R_u^*$.

The time evolution of the system states in Fig. 6 clearly supports our previous statement: without consuming additional computational resources, the prediction-based control approach can give rise to a better system performance (a rapidly stable trajectory) than conventional control method (a highly oscillating trajectory).

6. CONCLUSION

By redesigning the control structure, we show that the system performance of control systems subject to limited and time-varying computational resources can be improved significantly without requiring additional computational resources. This achievement is made based on the better use of the available computational resources and can be readily implemented since it does not affect the existing scheduling algorithms. In this sense the proposed approach can be regarded as a useful completion of the scheduling algorithms used in embedded control systems and is of importance in the practical implementation of embedded control systems.

REFERENCES

Atzori, L., Iera, A., and Morabito, G. (2010). The Internet of things: A survey. *Computer Networks*, 54(15), 2787 – 2805.

Baillieul, J. and Antsaklis, P.J. (2007). Control and communcation challenges in networked real-time systems. *IEEE Proc.*, 95(1), 9–28.

Bhattachary, R. and Balas, G. (2004). Anytime control algorithm: Model reduction approach. *J. Guid. Control Dyn.*, 27(5), 767–776.

Cervin, A. (2003). *Integrated Control and Real Time Scheduling*. Ph.D. thesis, Department of Automatic Control, Lund Institute of Technology, Lund, Sweden.

Curu, L. and Sorel, Y. (2004). Non-preemptive scheduling algorithms and schedulability conditions for real-time systems with precedence and latency constraints. Technical report, INRIA.

Graham, S. and Kumar, P.R. (2003). The convergence of control, communication and computation. In M. Conti, S. Giordano, E. Gregori, and S. Olariu (eds.), *Personal Wireless Communication*, 458–475. Springer-Verlag, Heidelberg.

Kranz, M., Holleis, P., and Schmidt, A. (2010). Embedded interaction: Interacting with the Internet of Things. *IEEE Internet Comput.*, 14(2), 46 –53.

Law, Y., Palaniswami, M., Hoesel, L., Doumen, J., Hartel, P., and Havinga, P. (2009). Energy-efficient link-layer jamming attacks against wireless sensor network MAC protocols. *ACM Trans. Sens. Netw.*, 5(1), 1–38.

Marti, P., Velasco, M., Fuertes, J., Camacho, A., and Buttazzo, G. (2010). Design of an embedded control system laboratory experiment. *IEEE Trans. Ind. Electron.*, 57(10), 3297–3307.

Mattern, F. and Floerkemeier, C. (2010). From the internet of computers to the internet of things. In K. Sachs, I. Petrov, and P. Guerrero (eds.), *From Active Data Management to Event-Based Systems and More*, volume 6462 of *Lecture Notes in Computer Science*, 242–259. Springer Berlin Heidelberg.

Puccinelli, D. and Haenggi, M. (2005). Wireless sensor networks: Applications and challenges of ubiquitous sensing. *IEEE Circuits Syst. Mag.*, 5(3), 19–31.

Samii, S., Cervin, A., Eles, P., and Peng, Z. (2009). Integrated scheduling and synthesis of control applications on distributed embedded systems. In *Design, Automation Test in Europe Conference Exhibition (DATE '09)*, 57 –62.

Sgora, A., Vergados, D.J., and Vergados, D.D. (2015). A survey of tdma scheduling schemes in wireless multihop networks. volume 47, 1–39.

Walsh, G.C. and Ye, H. (2001). Scheduling of networked control systems. *IEEE Control Syst. Mag.*, 21(1), 57–65.

Xia, F. and Sun, Y. (2006). Control-scheduling codesign: A perspective on integrating control and computing.

Zhao, Y.B., Liu, G.P., and Rees, D. (2008). Integrated predictive control and scheduling co-design for networked control systems. *IET Control Theory Appl.*, 2(1), 7–15.

Zhou, L. and Chao, H.C. (2011). Multimedia traffic security architecture for the internet of things. *IEEE Netw.*, 25(3), 35 –40.