SUPPLEMENT OF "CLOSING THE GAP BETWEEN STABILITY AND SCHEDULABILITY: A NEW TASK MODEL FOR CYBER-PHYSICAL SYSTEMS"

Hoon Sung Chwa[1], Kang G. Shin[1], and Jinkyu Lee[2]

[1]Electrical Engineering and Computer Science, The University of Michigan, Ann Arbor, Michigan, U.S.A.

[2]Department of Computer Science and Engineering, Sungkyunkwan University (SKKU), Republic of Korea.

APPENDIX

A. The dynamics of a physical plant and stability analysis

In feedback control, the control task samples the state of the plant, computes a control input, and applies it to the plant; this entire process is repeated periodically. Then, the dynamics of a physical plant under such feedback control is described as

$$x(k+1) = Ax(k) + Bu(k) \tag{6}$$
$$y(k) = Cx(k) \tag{7}$$

where x is a vector of state variables, u is the control input, y is the control output. For sampling period T, the coefficient matrices A, B, and C describe the evolution of the state in $[t_k, t_{k+1})$ where $t_{k+1} = t_k + T$, and it is defined as

$$A = e^{A_c T}, \quad B = \int_0^T e^{A_c s} ds B_c, \quad C = C_c \tag{8}$$

where A_c, B_c, and C_c are the corresponding coefficient matrices of the continuous-time model [2]. For simplicity, we assume that the update is made at every sampling instant. The state representation for a situation where the update is applied as soon as the control signal is computed can be found in [2]. Note that our proposed task model and scheduling techniques can be applied to both cases.

Incorporating control update misses into the plant dynamics, let us consider two cases. If the control signal is computed within $[t_k, t_{k+1})$, it is updated at time t_{k+1} (see Fig. 7). On the other hand, if the computation of the control signal is not completed by t_{k+1}, the controller uses the last updated control signal, and this input is kept constant until the next control update according to zero-order hold. Let $m(k) - 1$ denote the number of consecutive control update misses at t_k where $m(k) \in \{1, 2, 3, ..\}$. Then, the evolution of $m(k)$ can be formulated as

$$m(k+1) = \begin{cases} 1, & \text{if a control signal is updated} \\ & \text{during } [t_k, t_{k+1}] \\ m(k)+1, & \text{otherwise.} \end{cases} \tag{9}$$

With control update misses, the dynamics presented in Eq. (6) is changed to

$$x(k+1) = Ax(k) + Bu(k - m(k)). \tag{10}$$

Note that the plant dynamics depends on both sampling period and deadline misses. In order to maintain plant stability, the control task generates a state feedback control signal, that is, $u(k) = -Kx(k)$, where K is the state feedback gain matrix.

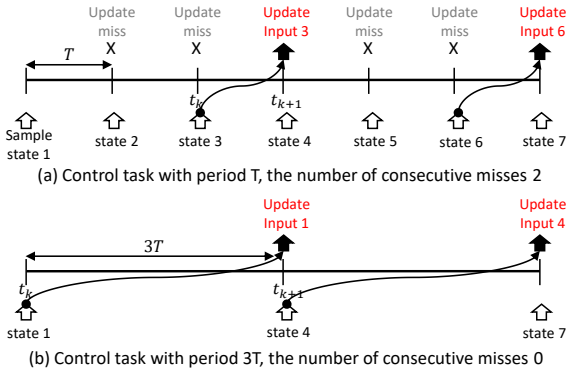

Fig. 7. Examples of periodic control task behavior with control update misses

For stability analysis, we need to consider the worst-case situation where the control inputs are not updated for N consecutive sampling intervals, where N is the maximum value of $m(k)$. Let the control input have been updated at t_k. Then, we define the augmented state vector for the N sampling intervals as $X(k) = [x(k), x(k-1), ..., x(k-N)]^\mathsf{T}$. Then, Eq. (10) can be written as

$$X(k+1) = A_{m(k)} X(k) \tag{11}$$

where

$$A_{m(k)} = \begin{bmatrix} A & -\delta_1 BK & -\delta_2 BK & \dots & -\delta_N BK \\ I & 0 & 0 & \dots & 0 \\ 0 & I & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & \dots & 0 & I & 0 \end{bmatrix} \tag{12}$$

Here, δ_i denotes the indicator function such that

$$\delta_i = \begin{cases} 0, & m(k) \neq i \\ 1, & m(k) = i. \end{cases} \tag{13}$$

The state equations under the worst-case situation become

$$\begin{aligned} X(k+1) &= A_1 X(k) \\ X(k+2) &= A_2 X(k+1) \\ &= A_2 A_1 X(k) \\ &\vdots \\ X(k+N) &= \prod_{i=1}^{N} A_i X(k) \\ &\triangleq GX(k). \end{aligned} \tag{14}$$

Then, we can use the following well-known stability analysis [2].

Lemma 5 ([2]): A discrete-time linear time-invariant system is *asymptotically stable* if and only if all eigenvalues of the corresponding system matrix G are strictly inside the unit circle in the complex plane.

The control performance of a physical plant can be expressed by a standard quadratic performance index [31]:

$$J = \sum_{k=0}^{N} \frac{1}{T} \int_{kT}^{(k+1)T} y(t)^2 dt. \tag{15}$$

Substituting the control output $y(t)$ in Eq. (15) by expressions in Eqs. (7), (10), and (11), we obtain:

$$\begin{aligned} J &= \sum_{k=0}^{N} \frac{1}{T} \int_{kT}^{(k+1)T} y(t)^2 dt \\ &= \sum_{k=0}^{N} \frac{1}{T} \int_{kT}^{(k+1)T} (CA_{m(t)}X(t))^2 dt. \end{aligned} \tag{16}$$

B. State-space representation of a case study

This subsection describes the state-space model for each physical plant considered in our case study.

Adaptive cruise control system. The continuous-time state-space model for the adaptive cruise control system [27] is presented as

$$\dot{x}(t) = A_c x(t) + B_c u(t) \tag{17}$$

where $x(t) = [\delta \ \Delta v \ a]^\mathsf{T}$, δ is the spacing error, Δv is the relative velocity between vehicles, a is the acceleration, and

$$A_c = \begin{bmatrix} 0 & 1 & -0.7 \\ 0 & 0 & -1 \\ 0 & 0 & 4 \end{bmatrix} \quad B_c = \begin{bmatrix} 0 \\ 0 \\ 4 \end{bmatrix}. \tag{18}$$

We set the state feedback gain K as $K = -[-15.2 \ 4.4 \ 1.6]$. Then, the continuous time model is discretized with sampling period T according to Eq. (6). For a given number of maximum consecutive update misses, we can calculate Eq. (14) and then apply for the stability analysis presented in Lemma 5.

Lane keeping control system. The continuous-time state-space model for the lane keeping control system [28] is presented as

$$\dot{x}(t) = A_c x(t) + B_c u(t) + F\dot{\Psi}_{des} \tag{19}$$

where $x(t) = [e_1 \ \dot{e_1} \ e_2 \ \dot{e_2}]^\mathsf{T}$, e_1 is the position error, e_2 is the yaw angle error, and

$$A_c = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -7.7585 & 116.3777 & 0.8709 \\ 0 & 0 & 0 & 1 \\ 0 & 0.4571 & -6.8558 & -7.0941 \end{bmatrix} \quad B_c = \begin{bmatrix} 0 \\ 69.9628 \\ 0 \\ 35.2093 \end{bmatrix}. \tag{20}$$

We assume the vehicle is traveling on a straight road, and then the desired yaw rate $\dot{\Psi}_{des}$ is zero. We set the state feedback gain K as $K = -[0.9258 \ 0.1695 \ 0.0015 \ 0.0454]$. Similarly to the adaptive cruise control system, we can derive a discrete-time model and apply for the stability analysis.

DC-servo control system. The continuous-time state-space model for the DC-servo control system [29] is presented as

$$\dot{x}(t) = A_c x(t) + B_c u(t) \tag{21}$$

where $x(t) = [v \ p]^\mathsf{T}$, v is the angular velocity, p is the angular position of the servo, and

$$A_c = \begin{bmatrix} -0.12 & 0 \\ 1 & 0 \end{bmatrix} \quad B_c = \begin{bmatrix} 22.5 \\ 0 \end{bmatrix}. \tag{22}$$

We set the state feedback gain K as $K = -[0.2391 \ 1.156]$. Similarly to the adaptive cruise control system, we can derive a discrete-time model and apply for the stability analysis.

C. Proof of Lemma 3

Lemma 3: Suppose J_j^q satisfies the following condition at time t. Then, J_j^q are schedulable under fixed-priority scheduling unless any non-critical job is added to Q_r after t.

$$\begin{aligned} &\max\Big(\sum_{\tau_k \in hp(j)} C_k(t) - (r_j(t) - t), 0\Big) \\ &+ \sum_{\tau_k \in hp(j)} WC_k(\max(r_j(t) + T_j - r_k(t), 0)) + C_j \leq T_j. \end{aligned} \tag{5}$$

Proof: We will prove that the execution of J_j^q will be completed within its scheduling window if Inequality (5) holds. By definition, the release time of J_j^q is $r_j(t)$, and its scheduling window is $[r_j(t), r_j(t) + T_j)$. The first term of the left-hand side (LHS) of Inequality (5) describes an upper-bound of leftover execution of higher-priority active jobs in $Q_r(t) \cup \{J_i^c\}$ at time $r_j(t)$. The second term of the LHS of Inequality (5) describes an upper-bound of the execution of critical jobs of τ_k in $[r_k(t), r_j(t) + T_j)$ as shown in Fig. 4(b). There exist three cases: (a) $r_k(t) \leq r_j(t)$, (b) $r_j(t) < r_k(t) < r_j(t) + T_j$, (c) otherwise. In case (a), we consider that the job of τ_k released before $r_j(t)$ is fully executed in $[r_k(t), r_j(t) + T_j)$, which is pessimistic but safe. In case (b), no critical job of τ_k release after t is executed in $[r_j(t), r_k(t))$. In case (c), no critical job of τ_k release after t is executed in $[r_j(t), r_j(t) + T_j)$. This implies that J_j^q will finish its execution within its scheduling window if Inequality (5) holds, which proves the lemma. ■

D. Hard real-time task set generation

We generate 91 hard real-time task sets while varying their total utilization of tasks from 0.1 to 1.0 with an incremental step of 0.1, resulting in 910 task sets. Given the total utilization (U_h) for a hard real-time task set, the number of tasks is uniformly chosen in $[4, 10]$, and each task is generated as follows. The utilization U_i of each task τ_i is randomly generated such that $\sum U_i = U_h$. The period and deadline of τ_i ($T_i = D_i$) are uniformly chosen in $[10, 1000]$, and the worst-case execution time is computed as $C_i = U_i T_i$. To generate an integer value of C_i, we use the floor function and generate a task set whose total utilization is in $[U_h - 0.1, \ U_h)$ for a given U_h.