

Performance-Aware Scheduler Synthesis for Control Systems*

Rupak Majumdar
MPI-SWS and UCLA
rupak@mpi-sws.org

Indranil Saha
UCLA
indranil@cs.ucla.edu

Majid Zamani
UCLA
zamani@ee.ucla.edu

ABSTRACT

We consider the problem of designing a cyber-physical system where several control loops share the same architectural resources. Typically, the design of such systems proceeds in two steps. In the *platform independent* step, for each control loop in the system, the control designer calculates a control law and a sampling time that together ensure that the control loop has certain desired performance. Then, in the *platform dependent* step, these control tasks are scheduled on the platform, and a schedulability analysis determines if (and how) the control laws can be implemented and scheduled without missing the sampling deadlines.

In this paper, we explore an alternative *controller-scheduler co-design* approach that aims to achieve optimal performance for all the individual control loops maintaining fairness. We first analyze the control systems to find out the rates at which control signals should be dropped to maintain schedulability and the optimal performance. We then use the rates to compute a schedule statically. We show a control theoretic approach to compute the effect of the rate of drops on the performance of the control systems and provide an SMT-based scheduling algorithm that takes as input control tasks, their periods, worst-case execution times, and their rate of drops, and outputs a static schedule that optimizes the performance of the control systems. We demonstrate our results through a case study on a family of inverted pendulums sharing the same processor for their control computations.

Categories and Subject Descriptors

D.4.8 [Software Engineering]: Organization and Design—*Real-time systems and embedded systems*

*This research was sponsored in part by the DARPA award HR0011-09-1-0037, the NSF awards 0720881, 0953994, and 1035916, and a grant from Toyota Motors.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

EMSOFT'11, October 9–14, 2011, Taipei, Taiwan.

Copyright 2011 ACM 978-1-4503-0714-7/11/10 ...\$10.00.

General Terms

Performance

Keywords

Control Systems, Scheduling, Performance, Synthesis

1. INTRODUCTION

Cyber-physical systems, where computers interact with and control physical phenomena, form the basis of many critical applications. In a typical cyber-physical application, such as automotive control, there are many control loops, responsible for various aspects of the system, that run simultaneously and share the same architectural resources (such as CPUs or buses). The traditional design of such systems proceeds in two steps. In the first, *platform independent* step, control engineers separately construct mathematical models and control laws for each independent subsystem that ensure each controlled system has stability and desired performance. The end product of this step is a set of software tasks computing the control laws and sampling periods for each task, ignoring any architectural constraints. In the second, *platform dependent* step, software engineers implement the control tasks on the platform, taking into account worst-case execution times of the tasks as well as scheduling constraints arising out of the use of shared resources. For example, if two control loops are implemented on the same CPU, then the scheduler must ensure that both tasks can run while meeting the deadlines imposed by their periods. If the scheduling constraints cannot be met, then the designer typically changes the architectural constraints until the constraints can be met (or sends the problem back to the control engineers for a different solution).

While the two-phase design provides separation of concerns between control law design and software design, it often has the drawback that the cost of the implementation becomes higher in order to meet the often overly-strict requirements imposed by the control designers. On the other hand, a global control-software co-optimization problem does not usually scale. In this paper, we explore the co-design of control and software systems, while maintaining the separation of concerns. We exploit the observation that control systems do not always achieve optimal performance when a control signal is transmitted to the plant in every cycle; rather, the relationship between the performance of the control system with the number of times control signals are sent to the plant in a fixed period is quite irregular. We provide a precise analysis of the relationship between the fraction

of times when control tasks are “dropped” by the scheduler and the performance of the system. In a second step, we use this information to derive a scheduler that occasionally drops a control signal but ensures that the rate of dropped control signals corresponds to the rate for optimal performance. Thus, in our methodology, we achieve end-to-end performance but keep a separation of concerns: a purely modular, control theoretic approach provides the relationship between the performance of the control systems and the rate of drops, and a purely software-related scheduler synthesis exploits the slack.

Technically, we make two contributions. First, we give a control theoretic calculation that provides a relationship between the long run average of the dropped packets with the performance of a control system. We model control systems in which a scheduler can discard control computations in certain cycles as networked control systems. The action of the scheduler is modeled as a switch that determines if the control signal reaches the plant (the task is scheduled) or gets dropped (the task is discarded). In the latter case, the actuator values are kept fixed to their values in the previous cycle. Using results from networked control systems [6], we give bounds on the fraction of drops that still guarantees the stability of the plant, as well as derive a relationship between the upper bound of the \mathcal{L}_∞ to RMS gain of the plant (a standard measure of control systems performance) and the fraction of packet drops, as well as an optimization problem in terms of linear matrix inequalities that can be used to constructively determine this relationship. We note that the relationship between packet-drop rate and performance is not monotonic, as might be expected. In particular, if a system is not schedulable, simply adding more resources to make it schedulable (as done in current practice) may not necessarily yield a better performance.

Second, we provide an automatic technique that takes control tasks, their periods, worst-case execution times and deadlines, as well as the rates at which control signals should be dropped, and outputs a static schedule (if possible) that guarantees that in a given period of time, the number of control signals are dropped in such a way that the optimal performance is achieved for each control loop. Our algorithm sets up the scheduler synthesis problem as a constraint satisfaction question, where the variables encode which task gets run in which time slot, and uses off-the-shelf SMT solvers to find models of the constraints.

We have implemented our methodology in a tool that determines schedulability of multiple control tasks sharing the same computation platform. We expect that a separate tool computes worst case execution times, but use off-the-shelf linear matrix inequality solvers (based on semi-definite programming) as well as SMT solvers to synthesize performance bounds and static schedulers. We have applied our implementation on a standard benchmark of multiple inverted pendulums sharing the same processor [28], and show how we can completely automatically derive schedulers ensuring optimal performance levels, even when the original system is unschedulable.

Related Work. The marriage of control-theoretic calculations and software verification has been studied in several recent papers [23, 24, 1, 2]. The co-design problem of feedback controllers and schedulers has been studied in the past [19, 18, 3, 17, 28], focusing on the choice of sampling times such that the system is schedulable and each

system attains optimal performance. An extensive review of this field can be found in [26]. Cervin introduced the idea of feedback scheduling [7] to perform online schedule design to cope with varying or unknown workloads. Recently Zhang et al. [28] present theoretical results on scheduling of a number of single-input single-output (SISOs) control systems to achieve balances among robustness, schedulability, and power consumption. However, the global optimization problems that arise in these works can be hard to solve efficiently. Branicky et al. [6] proposed a technique to co-design feedback controllers and schedulers for networked control systems [29] with packet dropouts. They studied the effect of packet dropout on the stability of control systems and applied the rate monotonic scheduling algorithm [14] to schedule controllers in such a way that the controllers maintain stability when the rates of packet dropouts are bounded above. However, they did not consider the tradeoff of performance and packet dropout in their work. In this work we show that it is possible to achieve better performance than just achieving stability by suitably choosing the rate of packet dropout for a controller. The rates of packet dropout are chosen in such a way that all the control tasks are schedulable and individual control systems achieve optimal performance. Our objective is to synthesize a scheduler statically that will provide a schedule following which all the control systems will achieve the best possible performance maintaining fairness.

We automatically synthesize a scheduler considering dropout of packets. We formulate the scheduling problem as a constraint solving problem, and use the SMT solver Yices [8] to synthesize a scheduler automatically. Very recently, the merits of using SMT solver for scheduler synthesis have been championed in [21] and [12]. Steiner [21] presents an evaluation of scheduler synthesis with Yices for time-triggered multi-hop networks. Legriél and Maler [12] present a framework to solve the task graph scheduling problem on a multiprocessor, while achieving the cheapest configuration. Our results fall in the general class of *program synthesis*, and we exploit domain knowledge (e.g., control theoretic performance requirements) to synthesize task schedulers.

2. CONTROL SYSTEMS PERFORMANCE WITH LIMITED COMPUTATION

We now describe a control theoretic formulation of the behavior of a linear time invariant control system in which the control law may not be computed at every step.

2.1 Preliminaries

We recall some preliminaries from control theory; see [4] for more details. We consider linear time-invariant (LTI) control systems in discrete time, given by the following difference equation:

$$\begin{aligned} x(k+1) &= Ax(k) + B_1w(k) + B_2u(k), \\ y(k) &= Cx(k), \end{aligned} \quad (1)$$

where $x(k) \in \mathbb{R}^n$ denotes the state of the system at the k^{th} time step, $w(k) \in \mathbb{R}^l$ denotes a disturbance signal at the k^{th} time step, $u(k) \in \mathbb{R}^m$ denotes a control input at the k^{th} time step, $y(k) \in \mathbb{R}^p$ denotes the outputs (the measurement of the states by the sensors) at the k^{th} time step, and A , B_1 , B_2 , and C are real-valued matrices of the appropriate dimensions. Figure 1 shows a discrete time LTI system with

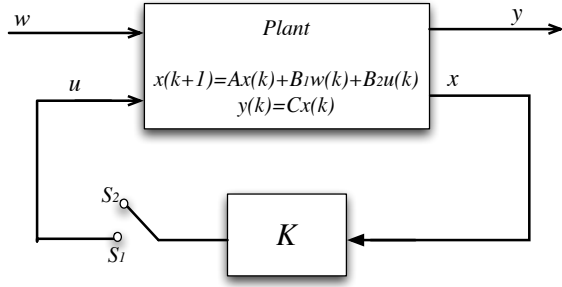


Figure 1: Linear control system with dropout

disturbance input w , control input u , and output y (ignore the switch for the moment). Such a discrete time system can be obtained from a continuous time system using a *sampling time* in the standard way [4]. The time steps $k = 0, 1, \dots$ are multiples of the given sampling time.

We consider *state feedback controllers* of the form $u(k) = -Kx(k)$, where the matrix K is called the *gain* of the controller. The aim of the controller is to ensure that the closed-loop system has certain properties, such as asymptotic stability, and a certain performance. Standard control-theoretic computations allow us to obtain state feedback controllers with the desired properties (see [4]).

2.2 Scheduled Linear Control Systems

The intuition behind our model is as follows. In each discrete time step $k = 1, 2, \dots$, the current state $x(k)$ is observed, and the scheduler tries to schedule the control computation task if possible. If the control task is scheduled, the signal $u(k) = -Kx(k)$ is computed and applied to the actuators. However, in the presence of shared resources, the control task may not be computed at every round. In time steps k at which the control signal is not computed (i.e., in which the scheduler does not schedule the control task), we expect the controller to retain its previous value: $u(k) = u(k-1)$.

We model the scheduled control system as a networked control system with a loss of “data packets” between the controller and the plant, that is, we model the link between the controller and the plant as a channel with a switch (see Figure 1). In each time step, if the switch is closed (indicating that the scheduler ran the control task), the control signal is the updated control law, but if the switch is open (indicating that the scheduler could not run the control task), the control signal is unchanged from the previous control signal. Cycles in which the switch is open are said to incur *dropouts* of the control signal. Our main result is the effect of dropouts on the performance of the control system.

In Figure 1, when the switch is closed (position S_1), the output of the controller is transmitted to the plant, and when the switch is open (position S_2), the output of the switch is held at the previous value. Similar to [6], the dynamics of the switch can be modeled formally as follows:

$$\text{When switch is in position } S_1 : u(k) = -Kx(k), \quad (2)$$

$$\text{When switch is in position } S_2 : u(k) = u(k-1). \quad (3)$$

Define the signal $s(k) = 1$ if the switch is in position S_1 at

the k^{th} time step, and $s(k) = 2$ if the switch is in position S_2 at the k^{th} time step. These correspond to the control input being computed or not.

By choosing $X = [x^T, u^T]^T$ as the new state vector, the closed loop system with dropout, depicted in figure 1, is given by:

$$\begin{aligned} X(k+1) &= \tilde{A}_{s(k)}X(k) + \tilde{B}_{1s(k)}w(k) \\ y(k) &= \tilde{C}_{s(k)}X(k), \end{aligned} \quad (4)$$

where, using the dynamics of the switch in (2) and (3), we obtain

$$\tilde{A}_1 = \begin{bmatrix} A & B_2 \\ -KA & -KB_2 \end{bmatrix}, \tilde{B}_{11} = \begin{bmatrix} B_1 \\ -KB_1 \end{bmatrix}, \tilde{C}_1 = [C \ 0_{p \times m}];$$

and

$$\tilde{A}_2 = \begin{bmatrix} A & B_2 \\ 0_{m \times n} & I_{m \times m} \end{bmatrix}, \tilde{B}_{12} = \begin{bmatrix} B_1 \\ 0_{m \times l} \end{bmatrix}, \tilde{C}_2 = [C \ 0_{p \times m}],$$

where $0_{m \times n}$ denotes the zero matrix in $\mathbb{R}^{m \times n}$ and $I_{m \times m}$ denotes the identity matrix in $\mathbb{R}^{m \times m}$.

Note that the successful transmission rate in this paper is the rate at which the switch in Figure 1 is in position S_1 . Following [10], the *successful transmission rate* r is given by

$$r = \lim_{L \rightarrow \infty} \frac{1}{L} \sum_{k=0}^L (2 - s(k)).$$

The dropout rate means the rate at which the switch in Figure 1 is in S_2 . If the successful transmission rate for a control system is r , its dropout rate is $1 - r$. Clearly, in the former case, the scheduler runs the control task and updates the actuator, and in the latter case, it does not. The following theorem provides a sufficient condition on the successful transmission rate that guarantees the stability of the closed loop system.

THEOREM 1 ([6]). *Consider the LTI control system in (4). Assume that r is the successful transmission rate and the closed loop system with no dropout and no disturbance is stable (i.e., $\max_i |\lambda_i(A - B_2K)| < 1$, where $\lambda_i(A - B_2K)$ is the i -th eigenvalue of the matrix $A - B_2K$).*

- If matrix A is marginally stable ($\max_i |\lambda_i(A)| \leq 1$), the LTI control system with dropout in (4), with no disturbance, is exponentially stable for all $0 < r \leq 1$;
- If matrix A is unstable ($\max_i |\lambda_i(A)| > 1$), then the LTI control system with dropout in (4), with no disturbance, is exponentially stable for all

$$r_{\min} < r \leq 1,$$

$$\text{where } r_{\min} = \frac{1}{1 - \gamma_1/\gamma_2}, \gamma_1 = \log[\max_i |\lambda_i(A - B_2K)|], \text{ and } \gamma_2 = \log[\max_i |\lambda_i(A)|].$$

2.3 Bound on \mathcal{L}_∞ to RMS gain

We now consider the \mathcal{L}_∞ to RMS gain as a performance criterion for LTI control systems and consider the effect of dropouts.

For the discrete-time LTI control system in (4), the \mathcal{L}_∞ to RMS induced gain from w to y is defined as follows:

$$\sup_{\|w\|_\infty \neq 0, X(0)=0} \frac{\left(\limsup_{l \rightarrow \infty} \frac{1}{l} \sum_{j=0}^l y^T(j)y(j) \right)^{\frac{1}{2}}}{\|w\|_\infty}, \quad (5)$$

where $\|w\|_\infty := \sup\{\|w(k)\|_2, k \geq 0\}$, and $\|w(k)\|_2 = \sqrt{w^T(k)w(k)}$.

The \mathcal{L}_∞ to RMS induced gain is a performance criterion showing the effect of the disturbance on the output of the plants [10]. Having smaller \mathcal{L}_∞ to RMS induced gain implies better performance in the sense that the effect of disturbance on the output of the plant is smaller.

The following theorem provides a relationship between the successful transmission rate and the upper bound of the \mathcal{L}_∞ to RMS gain for the control system in (4).

THEOREM 2. *Consider the discrete time LTI control system in (4) with the successful transmission rate r . The \mathcal{L}_∞ to RMS gain is less than positive constant γ if there exists a piecewise continuous function $V : \mathbb{R}^{n+m} \rightarrow \mathbb{R}_{\geq 0}$, such that $V(0) = 0$, and $\gamma_1, \gamma_2 \in \mathbb{R}$ such that*

$$r\gamma_1^2 + (1-r)\gamma_2^2 < \gamma^2, \quad (6)$$

and

$$V(\tilde{A}_i X + \tilde{B}_{1i} w) - V(X) \leq \gamma_i^2 w^T w - y^T y, \text{ for } i = 1, 2. \quad (7)$$

PROOF. We proceed as in [10], adapting the argument to discrete time control systems. Using inequality (7), we have:

$$\begin{aligned} V(X(k+1)) - V(X(k)) &\leq \gamma_i^2 w(k)^T w(k) - y(k)^T y(k) \\ &\leq \gamma_i^2 \|w\|_\infty^2 - y(k)^T y(k), \end{aligned} \quad (8)$$

where either $i = 1$ or $i = 2$ because at time instant k , the switch in Figure 1 is either closed or open. By summing up inequalities (8) for $k = 0, 1, \dots, l$, we obtain:

$$\begin{aligned} V(X(l+1)) - V(X(0)) &\leq \gamma_1^2 \left(\begin{array}{l} \text{total times the} \\ \text{switch is closed} \\ \text{between 0 and } l \end{array} \right) \|w\|_\infty^2 \\ &+ \gamma_2^2 \left(\begin{array}{l} \text{total times the} \\ \text{switch is open} \\ \text{between 0 and } l \end{array} \right) \|w\|_\infty^2 - \sum_{k=0}^{k=l} y(k)^T y(k). \end{aligned}$$

In the limit, when $l \rightarrow +\infty$, the total times that the switch is closed is equal to rl and the total times that the switch is open is equal to $(1-r)l$. Therefore, since $X(0) = 0$ in (5), we have:

$$V(X(l+1)) \leq \gamma_1^2 r l \|w\|_\infty^2 + \gamma_2^2 (1-r) l \|w\|_\infty^2 - \sum_{k=0}^{k=l} y(k)^T y(k).$$

Using (6) and $V(X(l+1)) \geq 0$, we obtain:

$$\frac{\frac{1}{l} \sum_{j=0}^l y^T(j)y(j)}{\|w\|_\infty^2} \leq r\gamma_1^2 + (1-r)\gamma_2^2 < \gamma^2. \quad (9)$$

Therefore, we get:

$$\limsup_{l \rightarrow \infty} \frac{\frac{1}{l} \sum_{j=0}^l y^T(j)y(j)}{\|w\|_\infty^2} < \gamma^2, \quad (10)$$

which completes the proof. \square

In the next lemma, we show that by choosing $V(X) = X^T P X$, the inequality (7) becomes a Linear Matrix Inequality (LMI) for the control system in (4).

LEMMA 1. *Consider the control system in (4). Using $V(X) = X^T P X$, the inequality (7) becomes an LMI as follows:*

$$\begin{bmatrix} \tilde{A}_i^T P \tilde{A}_i - P + \tilde{C}_i^T \tilde{C}_i & \tilde{A}_i^T P \tilde{B}_{1i} \\ \tilde{B}_{1i}^T P \tilde{A}_i & \tilde{B}_{1i}^T P \tilde{B}_{1i} - \gamma_i^2 \end{bmatrix} \leq 0. \quad (11)$$

PROOF. In the proof, we drop the arguments of X and w for simplicity.

$$\begin{aligned} V(\tilde{A}_i X + \tilde{B}_{1i} w) - V(X) &= (\tilde{A}_i X + \tilde{B}_{1i} w)^T P (\tilde{A}_i X + \tilde{B}_{1i} w) \\ &- X^T P X = X^T \tilde{A}_i^T P \tilde{A}_i X + w^T \tilde{B}_{1i}^T P \tilde{A}_i X + X^T \tilde{A}_i^T P \tilde{B}_{1i} w \\ &+ w^T \tilde{B}_{1i}^T P \tilde{B}_{1i} w - X^T P X \leq \gamma_i^2 w^T w - X^T \tilde{C}_i^T \tilde{C}_i X. \end{aligned}$$

By arranging the terms we obtain:

$$\begin{aligned} X^T \left[\tilde{A}_i^T P \tilde{A}_i - P + \tilde{C}_i^T \tilde{C}_i \right] X + w^T \tilde{B}_{1i}^T P \tilde{A}_i X \\ + X^T \tilde{A}_i^T P \tilde{B}_{1i} w + w^T \left[\tilde{B}_{1i}^T P \tilde{B}_{1i} - \gamma_i^2 \right] w \leq 0. \end{aligned} \quad (12)$$

The inequality (12) can be rewritten as:

$$\begin{bmatrix} X \\ w \end{bmatrix}^T \begin{bmatrix} \tilde{A}_i^T P \tilde{A}_i - P + \tilde{C}_i^T \tilde{C}_i & \tilde{A}_i^T P \tilde{B}_{1i} \\ \tilde{B}_{1i}^T P \tilde{A}_i & \tilde{B}_{1i}^T P \tilde{B}_{1i} - \gamma_i^2 \end{bmatrix} \begin{bmatrix} X \\ w \end{bmatrix} \leq 0.$$

Since $[x^T w^T]$ is an arbitrary vector, the previous inequality is equivalent to the following LMI:

$$\begin{bmatrix} \tilde{A}_i^T P \tilde{A}_i - P + \tilde{C}_i^T \tilde{C}_i & \tilde{A}_i^T P \tilde{B}_{1i} \\ \tilde{B}_{1i}^T P \tilde{A}_i & \tilde{B}_{1i}^T P \tilde{B}_{1i} - \gamma_i^2 \end{bmatrix} \leq 0, \quad (13)$$

which completes the proof. \square

Note that by choosing $V(X) = X^T P X$ and for a given successful transmission rate r , we can minimize γ , the upper bound of the \mathcal{L}_∞ to RMS induced gain, by solving the following optimization problem:

$$\begin{aligned} \text{minimize} \quad & \gamma \\ \text{subject to} \quad & r\gamma_1^2 + (1-r)\gamma_2^2 < \gamma^2, \\ & \gamma_1, \gamma_2 \in \mathbb{R}, \\ & P > 0, \\ & \begin{bmatrix} \tilde{A}_i^T P \tilde{A}_i - P + \tilde{C}_i^T \tilde{C}_i & \tilde{A}_i^T P \tilde{B}_{1i} \\ \tilde{B}_{1i}^T P \tilde{A}_i & \tilde{B}_{1i}^T P \tilde{B}_{1i} - \gamma_i^2 \end{bmatrix} \leq 0, \end{aligned} \quad (14)$$

for $i = 1, 2$.

2.4 Scheduled Nonlinear Control Systems

We now provide an extension of the preceding results to nonlinear control systems in discrete time. We consider nonlinear control systems in discrete time given by the following difference equations:

$$\begin{aligned} x(k+1) &= f(x(k), u(k), w(k)), \\ y(k) &= h(x(k)), \end{aligned} \quad (15)$$

where f and h are smooth maps and $x(k)$, $u(k)$, $w(k)$, and $y(k)$ belong to the same spaces as in (1). Here, we consider state feedback controllers of the form $u(k) = k(x(k))$, where k is a smooth map. As in the linear case, the dynamics of the switch can be modeled formally as the following:

$$\text{When switch is in position } S_1 : u(k) = k(x(k)), \quad (16)$$

$$\text{When switch is in position } S_2 : u(k) = u(k-1). \quad (17)$$

By following the same strategies as we did for linear systems and choosing $X = [x^T, u^T]^T$ as the new state vector, the closed loop system with dropout is given by:

$$\begin{aligned} X(k+1) &= \tilde{f}_{s(k)}(X(k), w(k)), \\ y(k) &= \tilde{h}_{s(k)}(X(k)), \end{aligned} \quad (18)$$

where, using the dynamics of the switch in (16) and (17), we obtain:

$$\tilde{f}_1(X(k), w(k)) = \begin{bmatrix} f(x(k), u(k), w(k)) \\ k(f(x(k), u(k), w(k))) \end{bmatrix}, \quad (19)$$

$$\tilde{h}_1(X(k)) = h(x(k));$$

and

$$\tilde{f}_2(X(k), w(k)) = \begin{bmatrix} f(x(k), u(k), w(k)) \\ u(k) \end{bmatrix}, \quad (20)$$

$$\tilde{h}_2(X(k)) = h(x(k)).$$

The following theorem provides a sufficient condition on the successful transmission rate that guarantees the stability of the closed loop system.

THEOREM 3 ([10]). *Consider the nonlinear control system with dropout in (18) with the successful transmission rate r . The system with no disturbance is exponentially stable with decay rate greater than γ if there exists a piecewise continuous function $V : \mathbb{R}^{n+m} \rightarrow \mathbb{R}_{\geq 0}$, and $\gamma_1, \gamma_2, \beta_1, \beta_2 \in \mathbb{R}_{>0}$, such that*

$$\beta_1 \|X\|_2^2 \leq V(X) \leq \beta_2 \|X\|_2^2, \quad (21)$$

$$\gamma_1^T \gamma_2^{(1-r)} > \gamma > 1, \quad (22)$$

and

$$V(\tilde{f}_i(X, 0)) - V(X) \leq (\gamma_i^{-2} - 1) V(X), \text{ for } i = 1, 2. \quad (23)$$

As in the linear case, we can prove a similar relationship between the successful transmission rate and the \mathcal{L}_∞ to RMS gain for the nonlinear control system in (18). The proof of the following theorem is the same as the proof of Theorem 2.

THEOREM 4. *Consider the nonlinear control system with dropout in (18) with the successful transmission rate r . The \mathcal{L}_∞ to RMS gain is less than positive constant γ if there exists a piecewise continuous function $V : \mathbb{R}^{n+m} \rightarrow \mathbb{R}_{\geq 0}$, such that $V(0) = 0$, and $\gamma_1, \gamma_2 \in \mathbb{R}$ such that*

$$r\gamma_1^2 + (1-r)\gamma_2^2 < \gamma^2, \quad (24)$$

and

$$V(\tilde{f}_i(X, w)) - V(X) \leq \gamma_i^2 w^T w - y^T y, \text{ for } i = 1, 2. \quad (25)$$

Note that if functions f and h in (15) and the state feedback controller k are polynomial with respect to their arguments, by using SOS programming [16] we can search for functions V , decay rates and upper bounds of \mathcal{L}_∞ to RMS gain satisfying the inequalities in Theorems 3 and 4 for given values of r .

3. OPTIMAL PERFORMANCE SCHEDULER SYNTHESIS

Suppose we have N control systems that are all sharing the same computational resources. Each controller is implemented as a software task C_i that computes the control law ($u(k)$ in (1)). Each task C_i has two parameters (h_i, c_i), where h_i denotes the sampling time and c_i the worst-case execution time of the task. A task will be active immediately after the beginning of a sampling period and, following [13],

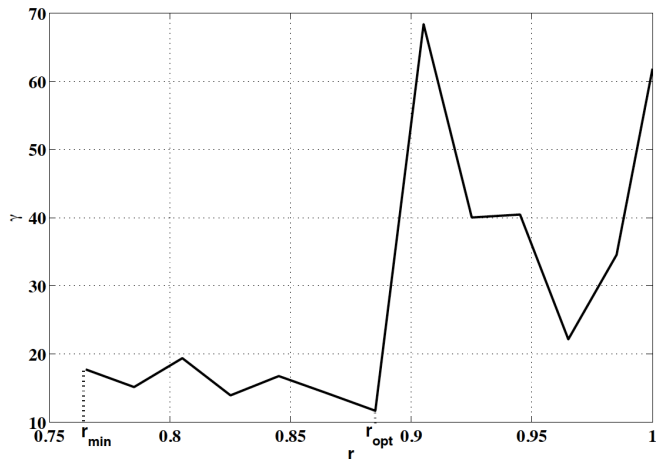


Figure 2: The upper bound of the \mathcal{L}_∞ to RMS gain vs successful transmission rate for an inverted pendulum

we assume that the deadline for the control task is the end of the sampling period. Suppose we have chosen successful transmission rates r_i for each control task. The system $\{(C_i, r_i) \mid i \in \{1, \dots, N\}\}$ is *schedulable with dropout* if it is possible to schedule the tasks so that each scheduled task finishes before its deadline, but the scheduler can drop r_i fraction of the task C_i . The following proposition follows using an argument similar to Theorem 7 in [13].

PROPOSITION 1. *The system $\{(C_i, r_i) \mid i \in \{1, \dots, N\}\}$ is schedulable with dropout iff $\sum_{i=1}^N r_i c_i / h_i \leq 1$.*

Notice that if a system is schedulable with dropout, then it remains schedulable with dropout if the rates are decreased. There are two issues in designing a scheduler. First, how should we choose the rates r_i so that the system is schedulable with dropout? Second, having chosen the rates such that the system is schedulable with dropout, how can we assign a static schedule that schedules all control tasks and ensures the rates chosen?

We now consider the problem of choosing the successful transmission rates. A lower bound on the rate for each system is given by Theorem 1: this is the rate r_{\min} required to ensure stability. Figure 2 shows how the upper bound on the \mathcal{L}_∞ to RMS gain varies with the successful transmission rate for an inverted pendulum. The figure is obtained by quantizing the successful transmission rates between their minimum value r_{\min} that ensures stability (see Theorem 1) and 1, and then solving the optimization problem in (14) for each choice of the successful transmission rate. As can be seen from Figure 2, the upper bound on the \mathcal{L}_∞ to RMS gain does not change monotonically with respect to the successful transmission rate. Ideally, for each system, we should choose the successful transmission rate for which the bound on the \mathcal{L}_∞ to RMS gain attains the minimum. However, when there are multiple control systems, the controllers compete for scheduling resources. Thus it may not be possible to accommodate the best successful transmission rates for all the control systems. For scheduler synthesis, we rather consider

a set of successful transmission rates that we call *eligible rates* for scheduler synthesis.

Fix a control system with dropout. For $0 < r \leq 1$, let $\gamma(r)$ denote the upper bound on the \mathcal{L}_∞ to RMS gain obtained by solving the optimization problem (14) for this choice of r . A successful transmission rate r is called *eligible* if it satisfies the following two conditions:

- $r \geq r_{\min}$, where r_{\min} is as in Theorem 1,
- for each $r' \in [r_{\min}, r)$, we have $\gamma(r') \geq \gamma(r)$.

The eligible rates provide “undominated” solutions: one cannot simultaneously reduce r and get better performance. We would like to find points on the Pareto curve of eligible rates for the N systems.

For each system C_i , we calculate the lower bound $r_{\min,i}$. We discretize the range $[r_{\min,i}, 1]$ with a chosen discretization factor and find the subset E_i of the discrete points that are eligible. We order the points in E_i by the total ordering \preceq : for $r_1, r_2 \in E_i$, if $r_1 \preceq r_2$ then $\gamma(r_2) \leq \gamma(r_1)$, that is, “higher” values in \preceq give better performance. Let $r_i^{best} \in E_i$ denote the maximal (in the \preceq -ordering) successful transmission rate in E_i .

We now have a multi-objective optimization problem: choose points in $E_1 \times \dots \times E_N$ that optimize the performance of each system. Though the performance of each control system is independently specified, the controllers compete for scheduling resources, and we may not be able to choose r_i^{best} for each control system. Instead, we look for undominated solutions.

One way to deal with this problem is to use a ranking method [27]. In this method, the objectives in the multi-objective optimization problem are ranked based on their importance. Of the N objectives, the rank 1 is assigned to the most important objective, rank N is assigned to the least important one, and the ranks of the other elements are assigned inversely proportional to their importance. If a number of elements have the same importance, then the average rank is used for all of them. Assume that the control system designer can provide such a ranking for the control systems. These ranks can be used to assign a weight to each objective in the multi-criterion optimization problem. There are several ways to assign the weights based on the ranks, we use the following two formulas [22]:

$$w_i = \frac{\frac{1}{q_i}}{\sum_{j=1}^N \frac{1}{q_j}} \quad w_i = \frac{(N - q_i + 1)}{\sum_{j=1}^N (N - q_j + 1)}$$

where q_i and w_i denote the rank and the weight of the i -th element respectively. The weights obtained by these formulas are called *rank reciprocal weights* and *rank sum weights* respectively.

Once the weights have been chosen, we define the *optimal performance scheduler synthesis problem* as choosing rates $r_i \in E_i$ such that the system is schedulable and the weighted sum $w_i \gamma(r_i)$ is minimized. Formally, we require

$$\begin{aligned} & \text{minimize} && \sum_{i=1}^N w_i \gamma(r_i) \\ & \text{such that} && r_i \in E_i && \text{for each } i \in \{1, \dots, N\} \\ & && \sum_{i=1}^N c_i r_i / h_i \leq 1 \end{aligned}$$

It is straightforward to show that the multiple-choice knapsack problem [20, 11] can be reduced to the above optimization problem. This establishes that the decision version

of the optimal-performance scheduler synthesis problem is NP-hard. (In the decision version, we ask if there exists $r_i \in E_i$ such that the system is schedulable with dropout and $\sum_{i=1}^N w_i \gamma(r_i) \leq K$ for an input parameter K .)

THEOREM 5. *The optimal performance scheduler synthesis decision problem is NP-hard.*

For the upper bound on the complexity of the problem, we need to be careful because in general the optimization problem (14) can only be approximated to a given accuracy ϵ . However, given an ϵ , the LMI can be solved in time polynomial in the size of the input matrices and $\log \frac{1}{\epsilon}$. Thus, the approximate version of the optimal performance scheduler synthesis decision problem, where we ask if $|\sum_{i=1}^N w_i \gamma(r_i) - K| \leq \epsilon$ for input parameters K and ϵ can be solved in NP.

4. SCHEDULER DESIGN

As the optimal-performance scheduler synthesis problem is computationally hard, we instead heuristically solve the following simplified version. First, for each system, we find out the minimum successful transmission rates $r_{\min,i}$ using Theorem 1. Second, we find out an upper bound $r_{\max,i}$ on the successful transmission rates for all the control systems, such that the system $\{(C_i, r_{\max,i}) \mid i \in \{1, \dots, N\}\}$ is schedulable with dropout. If for some i we have $r_{\max,i} < r_{\min,i}$, clearly the system is unschedulable. Otherwise, for each control system individually, we choose the best eligible successful transmission rate $r_{best,i}$ in the range $[r_{\min,i}, r_{\max,i}]$ (i.e., for which $\gamma(r)$ is minimized for $r \in [r_{\min,i}, r_{\max,i}]$). We then synthesize a static scheduler, ensuring the chosen rates of successful transmission rates for all the control systems. While the algorithm is not guaranteed to produce an optimal schedule, it heuristically attempts to jointly maximize the performance of all the systems.

We present our overall algorithm in two steps. In Section 4.1, we assume that we are given a successful transmission rate for each controller and present a set of constraints that must be satisfied by any static scheduler that schedules the system with dropout. In Section 4.2, we show how these set of constraints can be modified to find an upper bound on the successful transmission rates in such a way that the system is schedulable with dropout when these rates are chosen. Section 4.3 summarizes the algorithm.

4.1 Synthesis through Constraint Solving

To synthesize a static scheduler, we consider a duration T , called the *basic cycle*, for which we synthesize a schedule. The schedule for any duration $T' > T$ is obtained by repeating the synthesized schedule. Let us assume that each rate r_i is a fraction $\frac{k_i}{K_i}$, for integers k_i and K_i . We also assume, by scaling, that the sampling times h_i of the control systems take integer values. The duration of the basic cycle T is chosen such that the cycle of duration T can accommodate an integer number of tasks after considering dropouts. We set $T = lcm(K_1, K_2, \dots, K_N) \times lcm(h_1, h_2, \dots, h_N)$, where *lcm* stands for the least common multiple.

Let $m_i = \frac{T}{h_i}$ denote the number of task instances of C_i that are generated in the duration T . We introduce m_i variables $s_{i1}, s_{i2}, \dots, s_{im_i}$ for $i \in \{1, \dots, N\}$. Each variable s_{ij} takes values in $\{0, 1\}$. If the variable $s_{ij} = 1$, this denotes that the j th instance of task C_i (running in the time interval

$[(j-1)h_i, jh_i)$ was scheduled and if the variable $s_{ij} = 0$, this denotes that the j th instance of task C_i was dropped by the scheduler. Additionally, we introduce m_i variables $t_{i1}, t_{i2}, \dots, t_{im_i}$ for $i \in \{1, \dots, N\}$. If $s_{ij} = 1$, i.e., the instance of C_i in the slot $[(j-1)h_i, jh_i)$ is scheduled, then t_{ij} denotes the time in the interval $[(j-1)h_i, jh_i)$ when the execution of the task begins. Otherwise, if $s_{ij} = 0$, then t_{ij} is set to -1 .

Below we present the constraints on the scheduling problem.

1. For each controller i , for each of the m_i tasks in the basic cycle, the instance of the task C_i is either scheduled or dropped:

$$\bigwedge_{i \in \{1, \dots, N\}} \bigwedge_{j \in \{1, \dots, m_i\}} (s_{ij} = 1) \vee (s_{ij} = 0). \quad (26)$$

2. For each controller i , the number of instances of C_i that are scheduled in the basic cycle is equal to $m_i \times r_i$ (which is an integer, by choice of T):

$$\bigwedge_{i \in \{1, \dots, N\}} \sum_{j=1}^{m_i} s_{ij} = m_i \times r_i. \quad (27)$$

3. For all tasks C_i , if the instance of C_i is scheduled in a slot, the start time of the task should be scheduled after the beginning of the slot and the end time of the task should be before the end of the slot.

$$\bigwedge_{i \in \{1, \dots, N\}} \bigwedge_{j \in \{1, \dots, m_i\}} (s_{ij} = 1) \implies (t_{ij} \geq (j-1) \times h_i) \wedge (t_{ij} + c_{ij} \leq j \times h_i). \quad (28)$$

If the instance of the task is dropped, the start time of the task is set to -1 .

$$\bigwedge_{i \in \{1, \dots, N\}} \bigwedge_{j \in \{1, \dots, m_i\}} (s_{ij} = 0) \implies (t_{ij} = -1) \quad (29)$$

4. The time slot assigned for two tasks should not intersect:

$$\bigwedge_{\substack{i, k \in \{1, \dots, N\} \\ i \neq k}} \bigwedge_{j \in \{1, \dots, m_i\}} \bigwedge_{l \in \{1, \dots, m_k\}} (t_{ij} \geq 0) \wedge (t_{kl} \geq 0) \implies (t_{ij} + c_i \leq t_{kl}) \vee (t_{kl} + c_k \leq t_{ij}) \quad (30)$$

If the constraints are not satisfiable, there does not exist a scheduler for the given successful transmission rates. However, if the constraints are satisfiable we obtain a valid schedule. Further, the schedule obtained by repeating the static schedule every T units of time shows that the system $\{(C_i, r_i) \mid i \in \{1, \dots, N\}\}$ is schedulable with dropout.

4.2 Maximal Scheduler Synthesis

We now present how to find the maximal successful transmission rates for all the controllers that preserve schedulability. To solve this problem, we introduce variables x_i that denote the number of instances of task C_i that are scheduled in one basic cycle, and formulate a vector maximization problem [15] where the objective vector \mathbf{v} is given by $\mathbf{v}_i = x_i$. (The variables x_i are proportional to the rates.)

As in Section 3, we simplify the multi-objective optimization problem to a single-objective optimization problem by

choosing weights and taking the weighted sum of the vector. We assume the control designer additionally provides a priority for each control system. The weight of a control system is derived from its priority using the weight finding formulas introduced in Section 3. The single objective function is $\sum_{i=1}^N w_i x_i$. We formulate this optimization problem as a constraint solving problem, and find the optimal values for x_i by solving a series of feasibility problems.

First, we modify the set of constraints presented in Section 4.1 in the following way. The set of constraints in (27) are replaced by the following set of constraints:

$$\bigwedge_{i \in \{1, \dots, N\}} \sum_{j=1}^{m_i} s_{ij} \geq x_i \quad (31)$$

Moreover, we add the following constraint to the set of constraints:

$$\sum_{i=1}^N w_i \times x_i > \lambda, \quad (32)$$

where λ is a constant. If the constraints are infeasible then λ is certainly an upper bound for the objective function. We iteratively update λ and solve the set of constraints till we find the maximum λ for which the constraints are satisfiable. The satisfying assignment of those constraints gives the maximal rates x_i/T for which the system is schedulable with dropout.

4.3 Overall Algorithm

In this section we present the overall algorithm to synthesize a static schedule with dropout. The inputs of the algorithm are the systems C_i with their sampling times h_i , worst case execution times c_i , and priority $\pi_i \in \{1, \dots, N\}$. The output of the algorithm is a static schedule, if possible. The algorithm has the following steps.

In the first step, using Theorem 1, we find out the minimum successful transmission rates for all the controllers to achieve exponential stability. In the second step, we find out the maximal successful transmission rates by solving the optimization problem described in Section 4.2. If the minimum rate is greater than the maximal rate for some system, we stop and say unschedulable.

Now, for each system C_i , we have a range $[r_{\min, i}, r_{\max, i}]$ for the successful transmission rates, and any choice of r_i in this range ensures that the system is schedulable with dropout. For each controller, we find out the best eligible successful transmission rate in that range by discretizing the range $[r_{\min, i}, r_{\max, i}]$, solving the optimization problem (14) for rates chosen from this discretized space, and choosing the rate with minimum $\gamma(r)$.

Now, by solving the constraints in Section 4.1, we find a static schedule using the chosen rates.

5. EXPERIMENTS

5.1 Implementation

Figure 3 shows the toolbox that we have developed to synthesize static schedulers automatically for linear time invariant control systems with dropout. The inputs to our tool are (1) the mathematical description of the plants, (2) the linear controllers, (3) the sampling times, (4) the computation times for the control tasks, and (5) the ranking or priority of the control systems. We assume that the feedback

controllers and the sampling periods have been designed using standard control theoretic methods [4], and the execution times of the control tasks have been calculated using standard techniques [25]. We use a Matlab script to compute the lower bound on the successful transmission rates following Theorem 1. We automatically compute the upper bounds on the rates using Yices [8] from the sampling time, computation time, and the ranking of the control systems. Note that while the lower bounds on the rates are independent from each other, the upper bounds need to satisfy the scheduling constraints. It may be the case that the lower bound of the successful transmission rate is bigger than the upper bound for a control system. In this case, we cannot generate a schedule. The feasibility of a schedule is checked using a Matlab script. If there exists a feasible schedule, we find out the rates for which the performance of the control system becomes optimal. We choose a number of equally spaced points in the feasible range of the rates, and for each of them we find out the upper bound on the \mathcal{L}_∞ to RMS gain by solving optimization problem (14) using CVX [9]. The optimal rates are then used to form the constraints that are solved using Yices to get the optimal schedule.

To find out the upper bounds of the successful transmission rates, we solve a vector optimization problem using the weights calculated from the ranking of the control systems. The individual objective functions are the number of successfully transmitted packets (denoted by n_i) in a basic cycle with duration T , where $n_i = m_i \times r_i$, and m_i is the number of generated packets for the i -th controller in a basic cycle and r_i is the successful transmission rate of controller i . We solve this optimization problem by solving a number of feasibility problems and using the bisection method [5]. Let f denote the value of the objective function. As we want to find the maximum of successful transmission rates maintaining schedulability, we need to maximize the scalarized objective function. We add the constraint $f > C$ in the set of constraints, where C is a constant. We start with $C = 1$. As the sum of the weights is 1, and the values of n_i 's are greater than equal to 1, the constraints are satisfiable. We then iteratively find out the maximum value C such that the set of constraints are satisfiable. This is done as follows. At each step, if the constraints are satisfiable, then to choose the next C we multiply the current C by 2. If in any step, the constraints are unsatisfiable, then we choose the next C to be the average of the current value of C and the value of C for which the constraints were satisfiable for the last time. We stop in a step when the constraints are satisfiable and the difference between the previous and the current values of C is below a certain threshold (in our implementation, we choose 0.5). The values of $r_i (= \frac{n_i}{m_i})$'s in the last step provide the upper bounds on the successful transmission rates.

In the last step of our algorithm we need to find out the schedule corresponding to the chosen values of the successful transmission rates for different control systems. The constraints (27) in Subsection 4.1 are the corresponding set of constraints. As these constraints are equality constraints, they limit the feasible space of the constraints, and it is hard for the SMT solver to find a feasible solution. To alleviate this problem, for a control system, we choose a few different rates instead of just one rate, for which the performance of the control system is reasonably good. Now we replace a constraint set (27) in Subsection 4.1 by the disjunction of the constraints corresponding to different rates.

This increases the feasible search space, and Yices can find out a schedule relatively easily.

5.2 Example

We illustrate our technique by implementing controllers for multiple inverted pendulums sharing one processor. The example has been borrowed from [28]. The state-space representation of an inverted pendulum is given by:

$$\begin{aligned} \dot{x} &= Ax + B_1w + B_2u; \\ y &= Cx, \end{aligned}$$

where

$$\begin{aligned} A &= \begin{bmatrix} 0 & 1 \\ \frac{g}{l} & \frac{\rho}{ml^2} \end{bmatrix}, \quad B_2 = \begin{bmatrix} 0 \\ \frac{1}{ml} \end{bmatrix}, \\ B_1 &= \begin{bmatrix} 0.1 \\ 0 \end{bmatrix}, \quad C = [0.001, 0]. \end{aligned} \quad (33)$$

In this model, $x = [x_1, x_2]^T$ is the state of the system, with x_1 the angular position and x_2 the angular velocity of the point mass, m is the mass, l is the length of the rod, $g = 9.8\text{m/s}^2$ is acceleration due to gravity, ρ is the rotational friction coefficient, u is the applied force (control input), and w is the disturbance input. We show the result of our technique on three inverted pendulums. As in [28], we assume that all pendulums have mass $m = 0.5$, and rotational friction coefficient $\rho = 0.6$. The pendulums differ from each other in their lengths, chosen as $[l_1, l_2, l_3] = [0.20, 0.35, 0.50]$, their sampling times, chosen as $[h_1, h_2, h_3] = [0.010\text{s}, 0.015\text{s}, 0.020\text{s}]$, and their controllers, designed as $K_1 = [5.10, -2.50]$, $K_2 = [5.25, -1.1893]$, and $K_3 = [5.40, -0.45]$. We assume that the computation time for all the controllers is the same and equal to 0.005s . All constants and variables are expressed in SI units. The parameters of the control systems are summarized in Table 1.

Using Theorem 1, we obtain $r_{\min,1} = 0.7651$, $r_{\min,2} = 0.6375$, and $r_{\min,3} = 0.6589$, which guarantee that by choosing successful transmission rates bigger than these rates, the inverted pendulums, with no disturbances, are exponentially stable. The obtained maximal successful transmission rates are $r_{\max,1} = 1.00$, $r_{\max,2} = 0.90$, and $r_{\max,3} = 0.70$, when the ranks of the control systems are chosen to be 1, 2, 3, respectively, and we use *rank reciprocal weights*. Since $r_{\min,i} < r_{\max,i}$ for $i = 1, 2, 3$, feasible ranges of the successful transmission rates exist for all the pendulums. We now find out the successful transmission rates for the three controllers corresponding to the best possible upper bound on the \mathcal{L}_∞ to RMS gain in the feasible ranges of the rates. The optimal rates for which we can find schedules are 0.85, 0.85 and 0.70 respectively for controllers $i = 1, 2, 3$. The optimal rates have been chosen as multiples of 0.05. We synthesize the schedule for these rates which can be roughly the schedule corresponding to the optimal performance. The experimental results are summarized in Table 2.

To check the scalability of our tool, we conduct the following experiments. We execute our tool to synthesize optimal schedule for eight different scenarios, where we gradually increase the number of pendulums from 3 to 10. In all cases, the computation time for the controllers is 0.005s . In each scenario, the sampling time for all the pendulums are the same. However, the sampling times are chosen in such a way that all the controllers are not schedulable together. More precisely, if the computation time is denoted by C , and if N

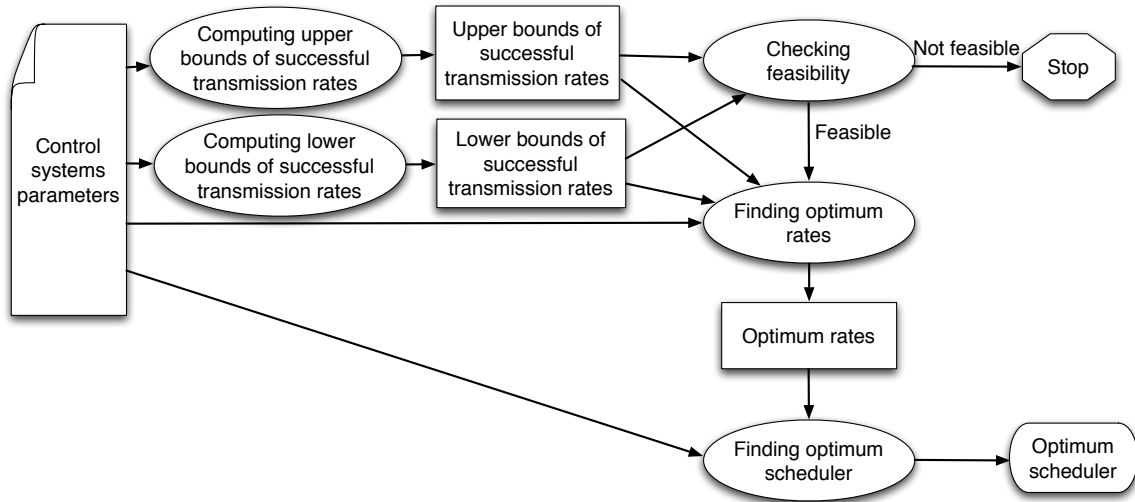


Figure 3: Scheduler synthesis toolbus

Systems	Mass (kg)	Length(m)	Priority	Controller gain	Sampling time (s)	Computation time (s)
System 1	0.50	0.20	1	[5.1, -2.5]	0.010	0.005
System 2	0.50	0.35	2	[5.25, -1.1893]	0.015	0.005
System 3	0.50	0.50	3	[5.4, -0.45]	0.020	0.005

Table 1: Control systems parameters

denotes the number of pendulums, then the sampling times of all the controllers are chosen to be $(N - 1) \times C$. Table 3 summarizes the experimental results. t_m denotes the time required to find the maximal successful transmission rates for all control systems guaranteeing schedulability (using the method described in Section 4.2). t_s denotes the time required to synthesize a scheduler statically based on the chosen successful transmission rates between the minimum and maximum successful transmission rates for all control systems (solving the set of constraints described in Section 4.1). t_m is considerably larger than t_s , since computing the maximal successful transmission rates requires solving a number of feasibility problems.

6. CONCLUSION

In this paper, we develop theoretical results as well as a tool for controller-scheduler co-design. Co-design lets us relax constraints on the hard real-time scheduling problem, while potentially getting better performance from the system. There are several possible generalizations. While we focus on the effect of schedulability, our techniques can be generalized with other sources of error, such as quantization errors [2] or additional network effects [24, 1]. Second, we only consider static schedules. It is worth exploring how dynamic scheduling policies interact with our control-theoretic analysis. Third, it will be interesting to extend our results to more complex hybrid systems with several discrete modes. Fourth, we have considered a very simple architecture, where the computation is shared. We leave the extension to more complex architectural constraints to future work.

7. REFERENCES

- [1] R. Alur, A. D’Innocenzo, K. Johansson, G. Pappas, and G. Weiss. Modeling and analysis of multi-hop control networks. In *Proceedings of RTAS*, pages 223–232, 2009.
- [2] A. Anta, R. Majumdar, I. Saha, and P. Tabuada. Automatic verification of control system implementations. In *Proceedings of EMSOFT*, pages 9–18, 2010.
- [3] K. Årzén, A. Cervin, J. Eker, and L. Sha. An introduction to control and scheduling co-design. In *Proceedings of CDC*, pages 4865–4870, 2000.
- [4] K. J. Åström and B. Wittenmark. *Computer-controlled systems: theory and design*. Prentice-Hall, Inc., 2nd edition, 1990.
- [5] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.
- [6] M. S. Branicky, S. M. Phillips, and W. Zhang. Scheduling and feedback co-design for networked control systems. In *Proceedings of CDC*, pages 1211–1217, 2002.
- [7] A. Cervin. *Integrated Control and Real-Time Scheduling*. PhD thesis, Lund University, 2003.
- [8] B. Dutertre and L. de Moura. A fast linear-arithmetic solver for DPLL(T). In *Proceedings of CAV*, pages 81–94, 2006.
- [9] M. Grant and S. Boyd. CVX: Matlab software for disciplined convex programming, version 1.21. <http://cvxr.com/cvx>, Jan 2011.
- [10] A. Hassibi, S. P. Boyd, and J. P. How. Control of

Systems	r_{min}	r_{max}	Optimal upper bound of the \mathcal{L}_∞ to RMS gain	Transmission rate used in static scheduler synthesis
System 1	0.7651	1.00	16.155	0.85
System 2	0.6375	0.90	10.5784	0.85
System 3	0.6589	0.70	24.12	0.70

Table 2: Experimental results

Number of pendulums	Sampling time	Computation time	t_m	t_s
3	10ms	5ms	3.548s	0.313s
4	15ms	5ms	5.948s	0.591s
5	20ms	5ms	1m34.576s	1.003s
6	25ms	5ms	5m20.364s	1.702s
7	30ms	5ms	11m5.501s	5.945s
8	35ms	5ms	12m39.703s	3.026s
9	40ms	5ms	25m10.479s	5.123s
10	45ms	5ms	11m0.143s	6.485s

Table 3: Time required to find maximal successful transmission rates and static schedule

- asynchronous dynamical systems with rate constraints on events. In *Proceedings of CDC*, volume 2, pages 1345–1351, 1999.
- [11] H. Kellerer, U. Pferschy, and D. Pisinger. *Knapsack Problems*. Springer Verlag, 2004.
- [12] J. Legriel and O. Maler. Meeting deadlines cheaply. Technical Report TR-2010-1, Verimag Research Report, 2010.
- [13] C. L. Liu and J. W. Layland. Scheduling algorithms for multiprogramming in a hard-real-time environment. *Journal of ACM*, 20(1):46–61, 1973.
- [14] J. W. S. Liu. *Real-Time Systems*. Prentice-Hall, Inc., 2000.
- [15] R. T. Marler and J. S. Arora. Survey of multi-objective optimization methods for engineering. *Structural and Multidisciplinary Optimization*, 26(6):369–395, 2004.
- [16] S. Prajna, A. Papachristodoulou, P. Seiler, and P. A. Parrilo. SOSTOOLS: Control applications and new developments. *Proceedings of IEEE International Symposium on Computer Aided Control Systems Design*, pages 315–320, 2004.
- [17] H. Reh binder and M. Sanfridson. Integration of off-line scheduling and optimal control. In *Proceedings of ECRTS*, pages 137–143, 2000.
- [18] M. Ryu, S. Hong, and M. Saksena. Streamlining real-time controller design: From performance specifications to end-to-end timing constraints. In *Proceedings of RTAS*, pages 91–99, 1997.
- [19] D. Seto, J. Lehoczky, L. Sha, and K. Shin. On task schedulability in real-time control systems. In *Proceedings of RTSS*, pages 13–21, 1996.
- [20] P. Sinha and A. A. Zoltners. The multiple-choice knapsack problem. *Operations Research*, 27(3):503–515, 1979.
- [21] W. Steiner. An evaluation of SMT-based schedule synthesis for time-triggered multi-hop networks. In *Proceedings of RTSS*, pages 375–384, 2008.
- [22] W. G. Stillwell, D. A. Seaver, and W. Edwards. A comparison of weight approximation techniques in multiple utility decision making. *Organizational Behavior and Human Performance*, 28:62–77, 1981.
- [23] G. Weiss and R. Alur. Automata based interfaces for control and scheduling. In *Proceedings of HSCC*, pages 601–613, 2007.
- [24] G. Weiss, S. Fischmeister, M. Anand, and R. Alur. Specification and analysis of network resource requirements of control systems. In *Proceedings of HSCC*, pages 381–395. Springer, 2009.
- [25] R. Wilhelm, J. Engblom, A. Ermedahl, N. Holsti, S. Thesing, D. Whalley, G. Bernat, C. Ferdinand, R. Heckmann, T. Mitra, F. Mueller, I. Puaut, P. Puschner, J. Staschulat, and P. Stenström. The worst-case execution-time problem – overview of methods and survey of tools. *ACM Trans. Embed. Comput. Syst.*, 7:36:1–36:53, 2008.
- [26] F. Xia and Y. Sun. Control-scheduling codesign: A perspective on integrating control and computing. *Dynamics of Continuous, Discrete and Impulsive Systems - Series B: Applications and Algorithms, Special Issue on ICSCA’06*, pages 1352–1358, 2006.
- [27] K. P. Yoon and C. Hwang. *Multiple attribute decision making: an introduction*. Sage Publications, Inc., 1995.
- [28] F. Zhang, K. Szwaykowska, W. Wolf, and V. Mooney. Task scheduling for control oriented requirements for cyber-physical systems. In *Proceedings of RTSS*, pages 47– 56, 2008.
- [29] W. Zhang, M. S. Branicky, and S. M. Phillips. Stability of networked control systems. *IEEE Control Systems Magazine*, 21:84–99, 2001.