LQG-Based Scheduling and Control Co-Design Using Harmonic Task Periods

Xu, Yang; Cervin, Anton; Årzén, Karl-Erik

2016

Document Version:
Publisher's PDF, also known as Version of record

LQG-Based Scheduling and Control Co-Design Using Harmonic Task Periods

Yang Xu, Anton Cervin, Karl-Erik Årzén

LUND
UNIVERSITY

Department of Automatic Control

Technical Report TFRT-7646
ISSN 0280–5316

Department of Automatic Control
Lund University
Box 118
SE-221 00 LUND
Sweden

Printed in Sweden by Media-Tryck.
Lund 2016

Abstract

Harmonic task scheduling has many attractive properties, including a utilization bound of 100% under rate-monotonic scheduling and reduced jitter. At the same time, it places a severe constraint on the task period assignment for any application. In this paper, we explore the use of harmonic task scheduling for applications with multiple feedback control tasks. We investigate the properties of harmonic scheduling and give an efficient algorithm to calculate response times for harmonic tasks. We present two algorithms for finding harmonic task periods: one that minimizes the distance from an initial set of non-harmonic periods and one that finds all feasible harmonic periods within a given set of ranges. We apply the algorithms in a control and scheduling co-design procedure, where the goal is to optimize the total performance of a number of control tasks that share a common computing platform. The procedure is evaluated in simulated randomized examples, where it is shown that, in general, harmonic scheduling combined with release offsets gives better control performance than standard, non-harmonic scheduling.

1

Introduction

Industrial controllers are often implemented as periodic tasks executing under the control of a real-time operating system (RTOS) using fixed-priority scheduling. Furthermore, for cost-saving reasons it is not uncommon having multiple controllers sharing the same execution platform. This leads to the problem of control and scheduling co-design. Most controllers are designed assuming a constant sampling period and a negligible or constant input-output latency, also referred to as control delay, or simply delay. Due to the interference that high-priority tasks impose on lower-priority tasks these assumptions do not always hold. The key problem in control and scheduling co-design is to optimize the combined performance of all the control tasks in the systems, subject to a schedulability constraint [Seto et al., 1996]. This is done by assigning suitable task parameters and designing feedback controllers that take the resulting task schedules into account.

In previous work [Bini and Cervin, 2008] the optimal task period assignment problem was solved assuming controller cost functions that depended linearly on the period and the delay, and where the delay was assumed to be constant and estimated using approximative response-time analysis. In our paper [Xu et al., 2014] the delay was instead modeled by the statistical distribution of the task response time, and an optimization-based approach was used to find the task periods for a set of controllers designed using stochastic LQG control techniques. The response-time distributions were found by simulating the task schedule. In the follow-up paper [Xu et al., 2015] our approach was instead to perturb the task periods slightly in order to obtain a finite hyperperiod, and, hence, a periodic delay pattern in the control loops. This periodicity was then explicitly accounted for in the control design, by applying periodic LQG control techniques.

In the current paper the focus is again on perturbing the task periods, but in this case to make the task periods harmonic. Harmonic task sets have many attractive properties. As long as the total utilization is less than or equal to 1, the task set is schedulable under both rate-monotonic (RM) and earliest-deadline-first (EDF) scheduling. Also, assuming constant execution times,

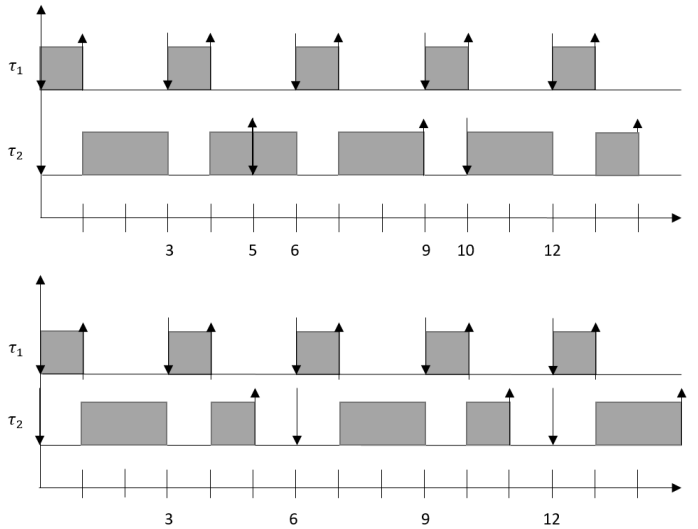

Figure 1.1 Two tasks with non-harmonic periods (upper plot) and harmonic periods (lower plot). The job arrivals are shown with down-arrows and the job finishing times with up-arrows.

the response times and start latencies are constant, something that leads to a particularly simple LQG control design. Consider the simple example shown in Fig. 1.1. In both cases we have two controller tasks running under RM priority assignment, where the sampling is performed at the job arrival times and the actuation is performed at the job finishing times, i.e., the control delay eauals the response times. In the upper plot the tasks have periods $\{3, 5\}$ and constant execution times $\{1, 3\}$. We assume that these periods have been chosen to give good total control performance. From the figure one can see that the response time of task τ_2 varies according to the pattern $5, 4, 4, 5, 4, 4 \ldots$. If one changes the period of task τ_2 to 6, i.e., harmonize the periods, then, as shown in the lower plot, the response time will be always equal to 5, i.e., more deterministic than in the first case. This will most likely lead to worse control performance since both the period and the average delay are larger than before. However, if one also introduces an offset of 1 for τ_2 then the response time will always be equal to 4. The question is then whether the decrease in control performance of the controller executing as task τ_2 caused by the longer sampling period is compensated for by the increased performance caused by the shorter and constant delay obtained through the period harmonization. This is the essence of the problem that we are investigating in this paper, and the evaluations performed show that

this is generally the case.

In the paper we present some new results on task period harmonization and scheduling analysis for harmonic tasks, with the twin goals of simplifying the control design and improving the performance of multi-loop control systems. The specific contributions of the paper are as follows:

- We propose a new analysis of response times under harmonic task scheduling. The algorithm is more efficient compared to previous methods, and, in contrast to previous work, it is applicable to real-numbered periods as well as real-numbered execution times.
- We give an algorithm for calculating the closest harmonic task periods (in the Euclidean sense) to a set of initial periods, under full utilization.
- We present a new algorithm for calculating all possible harmonic task periods given a set of allowed period ranges.
- We present a new control and scheduling co-design method based on the results above. The method strives to optimize the overall LQG control performance of a set of control tasks that share the same processor. The new method is compared to previous work in simulated randomized examples.

Outline of the Paper

In Section 2, related work is presented. In Section 3, we give the basic scheduling and control system model, including the metric we use to evaluate control performance. Section 4 provides the harmonic response-time analysis. The two algorithms to harmonize a non-harmonic task set are presented in Section 5. In Section 6, the harmonic task period assignment is evaluated with and without task offsets and is compared to non-harmonic period assignment. Finally, in Section 7, some concluding remarks are given.

2

Related Work

The control and scheduling co-design problem was first formulated in the seminal paper [Seto et al., 1996]. A cost function, which was a function of the task period, was introduced for each controller, and the design goal was to select periods such that the total control cost was minimized while keeping the task set schedulable. [Eker et al., 2000] analyzed how the cost function of a linear quadratic Gaussian (LQG) controller depends on the sampling period and proposed an on-line period adjustment algorithm. The theory behind LQG controller design is explained in, e.g., [Åström and Wittenmark, 1997].

Delay and jitter due to task scheduling can have a great impact on control performance [Wittenmark et al., 1995; Cervin et al., 2003]. In [Bini and Cervin, 2008] an integrated design method was proposed, where both periods and delays were taken into account when assigning controller task periods. The delay was—at design time—assumed to be constant and was found through an approximate response-time analysis. In [Xu et al., 2014], a stochastic LQG control and scheduling co-design method was proposed, in which controller response times were treated as independent random variables. However, the response-time of a periodic task actually appears in a periodic pattern. Hence, in [Xu et al., 2015], a periodic stochastic LQG control design method was proposed that takes the response-time distribution of each job in a hyperperiod into account. The resulting controller has time-varying parameters and its design is quite involved.

Cost functions for control tasks may be based on many different criteria, and the design problem can be formulated in a variety of ways. In [Zhang et al., 2008], an H_∞ design method for real-time controllers was proposed. [Samii et al., 2009a] proposed control scheduling co-design procedures for static-cyclic and priority-based scheduling. [Samii et al., 2009b] considered synthesis of multi-mode embedded control systems. [Goswami et al., 2012] synthesized schedules that optimize control performance which satisfy timing requirements by solving an integer linear programming problem.

Static scheduling approaches (such as RM scheduling) and dynamic scheduling approach (such as EDF), designed largely to achieve feasible tasks,

lead to conform pre-defined real-time system criteria. The most important real-time system criterion is the deadline constraint which means that all jobs of every task should have job response times less than the deadline. Then we have to analyze response times to check the schedulability. However, the response time calculation has been proved to be NP-hard [Eisenbrand and Rothvoss, 2008]. Furthermore, even though the response times are available, they are not constant. It leads to non-constant delay in control systems. Then it is hard to design a good controller which gives good performance due to the varying delays from job to job.

Harmonic task periods have give benefits both in real-time system scheduling and control system design [Busquets-Mataix et al., 1996; Shih et al., 2003]. The hyperperiods of harmonic tasks is finite and small [Ripoll and Ballester-Ripoll, 2013; Brocal et al., 2011]. RM scheduling of harmonic tasks is schedulable if the utilization is less than or equal to 1, e.g., [Han and Tyan, 1997]. [Bonifaci et al., 2013] gives an exact schedulability test for harmonic real-time tasks, which can be checked in polynomial time. [Nasri et al., 2014; Nasri and Fohler, 2015] proposed two efficient methods to assign harmonic periods to real-time tasks with period ranges: the forward approach in [Nasri et al., 2014] and the backward approach in [Nasri and Fohler, 2015].

3

Real-Time Control Co-Design System Model

3.1 Real Time System Model

We consider a real-time system, which is implemented by n tasks, running on a single processor under preemptive RM or earliest-deadline-first (EDF) scheduling. The ith task, denoted by τ_i, is defined by the 4-tuple (C_i, O_i, T_i^l, T_i^u), which is defined as follows:

- The *execution time* C_i is the length of time the task τ_i takes to execute. In this paper we will assume that this is constant. For simple control algorithms, including LQG controllers, executing on uniprocessors with simplistic memory hierarchies, this assumption is reasonably realistic.
- The *offset* O_i is the instant at which the first job of task τ_i is released. If no offset is specified, then $O_i = 0$ is assumed.
- The *period* T_i is the constant time between the release of two consecutive jobs of task τ_i. It can be chosen from the allowed time interval $[T_i^l, T_i^u]$.

In the RM scheduling case, the task *priority* is assumed to be implicitly assigned by the task ordering. A smaller task index i value means higher priority. The task ordering under both RM and EDF scheduling is decided by the periods, so that a task with a shorter period has a smaller task index i. If two or more tasks have the same period, then the order among them can be chosen arbitrarily.

For the given parameters mentioned above, the following characteristics can be calculated:

- The *hyperperiod* H is the least common multiplier of all the real numbers T_i, $i = 1 \cdots n$.

- The *response time* R_{ij} is the time that elapses from the job release time to the finish time of the jth job of task τ_i.
- The *start latency* S_{ij} is the time that elapses from the job release time to the start of the jth job of task τ_i.
- The task *utilization* $U_i = C_i/T_i$ measures the amount of computational resources required by the controller. We denote the *total utilization* of all control tasks by $U = \sum_i^n U_i$, which should be less than or equal to 1.

Each control task implements a LQG controller, for which we define the following timing parameters:

- The *sampling interval* h_{ij} is the time difference between the sampling operation of job j and job $j-1$ of task i. We assume that sampling jitter has been eliminated by enforcing sampling at the job release time; hence, $h_{ij} = T_i, \forall j$.
- The *delay* L_{ij} is the time interval between the sampling operation and the output operation of job j of task i. We assume that the output operation is performed when the job finishes; hence, $L_{ij} = R_{ij}$.

3.2 LQG Control Problem Formulation

For each task τ_i, a linear time-invariant (LTI) continuous time plant is defined in state-space form

$$\begin{aligned} \dot{x}_i(t) &= A_i x_i(t) + B_i u_i(t) + v_i(t) \\ y_i(t_k) &= C_i x_i(t_k) + e_i(t_k) \end{aligned} \tag{3.1}$$

where $x_i(t)$ is the state vector of the plant, $u_i(t)$ is the control input, $y_i(t)$ is the system output, and A_i, B_i, C_i are constant matrices. The disturbance $v_i(t)$ is continuous time white noise, while the measurement noise $e_i(t)$ is discrete time white noise.

For each plant, an LQG controller should be designed to minimize a quadratic cost function

$$J_i = \lim_{t\to\infty} \frac{1}{t} \int_0^t \left(x_i^T Q_{1ci} x_i + u_i^T Q_{2ci} u_i \right) \, \mathrm{d}\tau \tag{3.2}$$

where Q_{1ci} and Q_{2ci} are symmetric positive definite weighting matrices that penalizes state deviations and the control signal effort. The LQG controller is designed off-line, taking information about the expected delay (constant or random) into account. The control design gives rise to a linear controller with constant parameters.

4

Scheduling Analysis for Harmonic Tasks

Below we first give an algebraic characterization of the task periods under full utilization. After stating some general scheduling properties, we then give the new response-time analysis for harmonic tasks.

4.1 Characterization of Full-Utilization Harmonic Task Sets

Assume a harmonic task set consisting of n tasks and with full utilization, i.e.,

$$\frac{C_1}{T_1} + \frac{C_2}{T_2} + \cdots + \frac{C_n}{T_n} = 1$$

The fact that the task periods are harmonic can be expressed as $T_{k+1} = m_k T_k$, $m_k \in \mathbb{N}^+$, $k \in \{1, 2, \cdots, n-1\}$. Then one obtains

$$\frac{C_1}{T_1} + \frac{C_2}{m_1 T_1} + \frac{C_3}{m_1 m_2 T_1} + \cdots + \frac{C_n}{m_1 m_2 \cdots m_{n-1} T_1} = 1$$

Solving for T_1 gives

$$T_1 = C_1 + \frac{C_2}{m_1} + \frac{C_3}{m_1 m_2} + \cdots + \frac{C_n}{m_1 m_2 \cdots m_{n-1}}$$

From the harmonicity it follows that

$$T_2 = m_1 C_1 + C_2 + \frac{C_3}{m_2} + \frac{C_4}{m_2 m_3} + \cdots + \frac{C_n}{m_2 m_3 \cdots m_{n-1}}$$

$$\vdots$$

$$T_n = m_1 m_2 \cdots m_{n-1} C_1 + m_2 m_3 \cdots m_{n-1} C_2 + \cdots + C_n$$

This can be written in matrix form as

$$T = MC \tag{4.1}$$

with $T = \begin{bmatrix} T_1 & T_2 & \cdots & T_n \end{bmatrix}^T$ and with M being the reciprocal matrix given by

$$M = \begin{bmatrix} 1 \\ m_1 \\ \vdots \\ m_1 m_2 \cdots m_{n-1} \end{bmatrix} \begin{bmatrix} 1 & \frac{1}{m_1} & \cdots & \frac{1}{m_1 m_2 \cdots m_{n-1}} \end{bmatrix}$$

Algebraically, T is a linear combination of the execution times C. Geometrically, M is a linear projection of C onto a line L in the coordinate space, running through the origin. The vector $\begin{bmatrix} 1 & m_1 & m_1 m_2 & \cdots & m_1 m_2 \cdots m_{n-1} \end{bmatrix}^T$ lies on L. This is not an orthogonal projection. For given execution times $\{C_1, C_2, \cdots, C_n\}$ and $\{m_1, m_2, \cdots, m_{n-1}\}$ the harmonic periods can be calculated using Eq. (4.1).

4.2 Scheduling Analysis for Harmonic Tasks

Harmonic tasks have several properties that are appealing from a real-time control perspective [Lehoczky et al., 1989].

1. Schedulability is guaranteed as long as the total utilization is less than or equal to 1.
2. RM and EDF scheduling yields the same start latencies and response times for each task. Under EDF, we assume that the tasks have implicit relative deadlines equal to their periods, and that, if task τ_i and task τ_j $(i < j)$ have a deadline tie, then EDF will schedule task τ_i.

Further, under the assumption that execution times are constant for each task, it holds that

3. The response time for each task is constant, which is denoted as R_i for task τ_i.
4. The start latency for each task is constant, which is denoted as S_i for task τ_i, and is given by the response time of task τ_{i-1}, with τ_i being the current task.

The fact that the start latency of task i is equal to the worst-case response time of task $i-1$ implies that the delay can be large, which may degrade the control performance. This can be remedied by adding an offset to each task, so that the job start time always coincides with the job release time.

The following theorem can be used to calculate the response times under both RM and EDF scheduling when the periods are harmonic, using the properties presented above.

THEOREM 1
For RM and EDF scheduling with harmonic periods, given a set of n tasks $\{\tau_1, \tau_2, \cdots, \tau_n\}$ with utilization $\sum U \leq 1$, there exists a smallest $\hat{m} \in \{1, 2, \cdots, \prod_{i=0}^{j-1} m_i\}$ such that

$$\sum_{i=1}^{j} \left(\left\lceil \frac{m}{\prod_{k=0}^{i-1} m_k} \right\rceil C_i \right) - mT_1 \leq 0$$

for any $m \geq \hat{m}$. Then the response time of task τ_j $(j \leq n)$ is

$$R_j = \sum_{i=1}^{j} \left(\left\lceil \frac{\hat{m}}{\prod_{k=0}^{i-1} m_k} \right\rceil C_i \right)$$

□

Proof Assume $\hat{m} = \prod_{i=0}^{j-1} m_k$ and the utilization bound

$$\sum_{i=1}^{j} U_i = \sum_{i=1}^{j} \frac{C_i}{T_i} \leq 1.$$

Then

$$\begin{aligned} &\sum_{i=1}^{j} \left(\left\lceil \frac{\hat{m}}{\prod_{k=0}^{i-1} m_k} \right\rceil C_i \right) - \hat{m}T_1 \\ &= \left(\sum_{i=1}^{j} \frac{C_i}{\left(\prod_{k=0}^{i-1} m_k \right) T_i} - 1 \right) \hat{m}T_1 \\ &= \left(\sum_{i=1}^{j} \frac{C_i}{T_i} - 1 \right) \hat{m}T_1 \leq 0 \end{aligned}$$

□

and, hence, the inequality is proved.

If R_i is the response time of task τ_i, then when $0 \leq t \leq R_i$, the CPU is fully loaded, so

$$\sum_{i=1}^{j} \left(\left\lceil \frac{m}{\prod_{k=0}^{i-1} m_k} \right\rceil C_i \right) - mT_1 > 0$$

when $m < \hat{m}$. When $m = \hat{m}$,

$$\sum_{i=1}^{j} \left(\left\lceil \frac{m}{\prod_{k=0}^{i-1} m_k} \right\rceil C_i \right) - mT_1 \leq 0$$

and the response time of task τ_j is

$$R_j = \sum_{i=1}^{j} \left(\left\lceil \frac{\hat{m}}{\prod_{k=0}^{i-1} m_k} \right\rceil C_i \right)$$

The procedure to calculate the response time of task τ_j follows from Theorem 1. When m is chosen from 1, 2, ..., $\prod_{i=0}^{j-1} m_i$, we evaluate

$$\sum_{i=1}^{j} \left(\left\lfloor \frac{m}{\prod_{k=0}^{i-1} m_k} \right\rfloor C_i \right) - mT_1$$

and find $\hat{m}$ as the first m that makes this term negative. The procedure is summarized in Algorithm 1.

Algorithm 1 Response time calculation

procedure ResponseTime
  calculate R_j for $m_0, m_1, \cdots, m_{j-1}, C_1, C_2, \cdots, C_j, T_1$
  $T' \leftarrow MC$
  $T \leftarrow \frac{T_1}{T_1'} T'$
  $m \leftarrow \left\lfloor \frac{\left(1 - \sum_{i=1}^{j-1} \frac{C_i}{T_i}\right) T_j}{C_j} \right\rfloor \prod_{i=0}^{j-2} m_i$
  while $\sum_{i=1}^{j} \left(\left\lceil \frac{m}{\prod_{k=0}^{i-1} m_k} \right\rceil C_i \right) - mT_1 > 0$ **do**
    $m \leftarrow m + 1$
  end while
  $\hat{m} \leftarrow m$
  $R_j = \sum_{i=1}^{j} \left(\left\lceil \frac{\hat{m}}{\prod_{k=0}^{i-1} m_k} \right\rceil C_i \right)$
  return R_j ▷ Response time R_j
end procedure

In Algorithm 1, a linear search is used. The initial value of m is given by

$$m_{\text{initial}} = m_{\text{lower}} = \left\lfloor \frac{\left(1 - \sum_{i=1}^{j-1} \frac{C_i}{T_i}\right) T_j}{C_j} \right\rfloor \prod_{i=0}^{j-2} m_i$$

and it is increased by 1 every step, until $\hat{m}$ is found. By defining

$$m_{\text{upper}} = \left\lceil \frac{\left(1 - \sum_{i=1}^{j-1} \frac{C_i}{T_i}\right) T_j}{C_j} \right\rceil \prod_{i=0}^{j-2} m_i$$

it follows that $m_{\text{lower}} \leq \hat{m} \leq m_{\text{upper}}$. Furthermore, binary search on $[m_{\text{lower}}, m_{\text{upper}}]$ can be applied to improve the algorithm performance.

Example 1
Assume that we have three tasks with execution times $C = \begin{bmatrix} 0.9 & 6.3 & 9.1 \end{bmatrix}$ and that $T_1 = 7.7$, $m_1 = 2$, and $m_2 = 3$, i.e., $T = \begin{bmatrix} 7.7 & 15.4 & 46.2 \end{bmatrix}$. Assume that we want to calculate the response time of task τ_3. Using Algorithm 1, one then obtains

$$m_{\text{initial}} = \left\lfloor \frac{\left(1 - \sum_{i=1}^{2} \frac{C_i}{T_i}\right) T_3}{C_3} \right\rfloor \prod_{i=0}^{1} m_i = 2$$

The search in the while loop in Algorithm 1 gives, when $m = 3$,

$$\sum_{i=1}^{3} \left(\left\lceil \frac{m}{\prod_{k=0}^{i-1} m_k} \right\rceil C_i \right) - mT_1 = 2.2 > 0$$

and when $m = 4$,

$$\sum_{i=1}^{3} \left(\left\lceil \frac{m}{\prod_{k=0}^{i-1} m_k} \right\rceil C_i \right) - mT_1 = -5.5 < 0$$

This implies that $\hat{m} = 3$ and that

$$R_3 = \sum_{i=1}^{3} \left(\left\lceil \frac{\hat{m}}{\prod_{k=0}^{i-1} m_k} \right\rceil C_i \right) = 25.3$$

The time complexity of Algorithm 1 can be compared with the with the fixed priority scheduling response-time calculation algorithm in [Bonifaci et al., 2013]. In [Bonifaci et al., 2013], the authors assume that the task periods and the task execution times are integers. The harmonic periods are defined as $T_i | T_j$ (T_i divides T_j) or $T_j | T_i$ (T_j divides T_i) for all $i, j = 1, 2, \cdots, n$. The algorithm correctly computes the response time of task τ_n in $\mathcal{O}(n \log n + n \log P)$ where n is the number of tasks and $P = \max_{i=1}^{n} T_i$. In Algorithm 1 task periods and execution times may all be real-valued. The while loop in the algorithm will, in the worst case, run $\prod_{k=0}^{i-2} m_k$ times. The time complexity is $\mathcal{O}\left(\prod_{k=0}^{i-1} m_k\right)$ in the normal case and $\mathcal{O}\left(\log \prod_{k=0}^{i-1} m_k\right)$ if binary search is used.

Due to the different setups for the response time calculation in [Bonifaci et al., 2013] and this paper, one has to make adjustments of the proposed method. In order to make a fair comparison one may assume that the periods are all integers. Then it follows that $\prod_{k=0}^{i} m_k \leq P$ and $\prod_{k=0}^{i-1} m_k \leq P$, from

which it follows that $\mathcal{O}\left(\log \prod_{k=0}^{i-1} m_k\right) \leq \mathcal{O}(n \log n + n \log P)$ for any n, i.e. the complexity of the proposed algorithm is lower than what has been previously presented.

5

Finding Harmonic Control Task Periods

In control–scheduling co-design, it is typically assumed that the period of each control task can be chosen as a real value within a (possibly infinite) period range. In a prototypical problem formulation, the performance of each controller is described by a cost function $J(T)$, which is assumed to be an increasing function of the task period T, i.e., the lower the cost, the better the performance will be. The goal is to optimize the combined performance of all control tasks subject to a utilization constraint U_b, e.g.,

$$\underset{T_1,\ldots,T_n}{\text{minimize}}\ J = \sum_{i=1}^{n} J(T_i),\ \text{subject to}\ \sum \frac{C_i}{T_i} \leq U_b$$

If the function $J(T^{-1})$ is convex, efficient numerical methods are available to find the global optimum [Eker et al., 2000].

Here, we are interested in solving a similar co-design problem, but we want to restrict the possible task periods to be harmonic. However, optimization problems involving integers (i.e., the harmonic factors $m_1, \ldots, m_{n-1}$ in our case) are in general NP-hard [Papadimitriou, 1981], meaning that the optimal harmonic period assignment co-design problem cannot be solved efficiently. Hence, we propose the following two heuristic approaches to harmonic control task period assignment: 1) finding the closest harmonic period assignment to a set of initial periods, and 2) finding all possible harmonic period assignments that satisfy given task period ranges. In both cases, all feasible candidate solutions are then evaluated with regards to the combined control performance and the best solution is chosen.

5.1 Finding the Closest Harmonic Task Periods

We assume that a set of full-utilization initial non-harmonic task periods are given as $T^0 = \begin{bmatrix} T_1^0 & T_2^0 & \cdots & T_n^0 \end{bmatrix}^T$. The problem is then to find a set of harmonic periods that minimizes the Euclidean distance between this set and the initial periods:

$$\underset{T_1,\ldots,T_n}{\text{minimize}} \ \left\| T - T^0 \right\|$$
$$\text{subject to} \ \sum_{i=1}^{n} \frac{C_i}{T_i} = 1, \ \frac{T_{k+1}}{T_k} \in \mathbb{N}^+, \ k \in \{1, 2, \cdots, n-1\}$$

THEOREM 2
Let

$$T^* = \begin{bmatrix} T_1^* & T_2^* & \cdots & T_n^* \end{bmatrix}^T$$

be the solution of the above optimization problem. Then $T^* \in \{MC\}$, where M is defined in Eq. (4.1), $m_i \in \left\{ \left\lfloor \frac{T_{i+1}^0}{T_i^0} \right\rfloor, \left\lceil \frac{T_{i+1}^0}{T_i^0} \right\rceil \right\}$. □

Proof Let $f : \mathbb{R}^{n-1} \to \mathbb{R}^n$ be defined as

$$T = MC = f([m_1 \ m_2 \ \ldots \ m_{n-1}])$$

where the matrix M is given in Equation (4.1), and let the initial harmonic periods be $T^* = f(m^*_{\text{vector}})$, in which $m^*_{\text{vector}} = \begin{bmatrix} m_1^* & m_2^* & \ldots & m_{n-1}^* \end{bmatrix}$. Further define

$$\overline{m}_{\text{vector},j} = \left[m_1^* \ m_2^* \ \ldots \ \left\lceil \frac{T_{j+1}^0}{T_j^0} \right\rceil + 1 \ \ldots \ m_{n-1}^* \right]$$

$$\underline{m}_{\text{vector},j} = \left[m_1^* \ m_2^* \ \ldots \ \left\lfloor \frac{T_{j+1}^0}{T_j^0} \right\rfloor - 1 \ \ldots \ m_{n-1}^* \right]$$

Let $\underline{T} = f(\underline{m}_{\text{vector},j})$. Now consider Fig. 5.1 where the curve represents the utilization bound. Since this curve is convex, in the triangle $T^*T^0\underline{T}$, the angle between T^*T^0 and $T^*\underline{T}$ is greater than 90°. Then

$$\left\| T - T^0 \right\| < \left\| f(\underline{m}_{\text{vector},j}) - T^0 \right\|$$

and, similarly,

$$\left\| T - T^0 \right\| < \left\| f(\overline{m}_{\text{vector},j}) - T^0 \right\|$$

The two inequalities above also apply for the high-dimensional case. By choosing different value of m_i, one can prove that for each $i \leq n-1$, the two inequalities are valid. In the $n-$dimensional case, we need to check 2^{n-1} inequalities. □

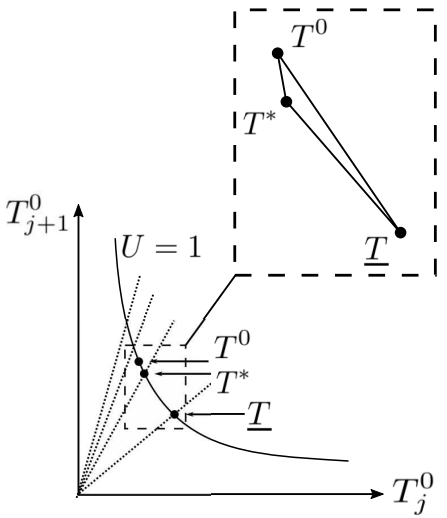

Figure 5.1 Finding the closest periods set to the initial periods set.

Example 2
Assume that we have three tasks with the same execution times as in Example 1, i.e., $C = \begin{bmatrix} 0.9 & 6.3 & 9.1 \end{bmatrix}$. Let the full-utilization initial periods be

$$T^0 = \begin{bmatrix} 12.3 & 13.7 & 19.4 \end{bmatrix}, \text{ with } \frac{T^0_2}{T^0_1} = 1.1, \ \frac{T^0_3}{T^0_2} = 1.4,$$

so $m_1 = 1$ or 2 and $m_2 = 1$ or 2. Four sets of harmonic periods can be calculated from Equation (4.1). The 2-norms of $T - T^0$ are given as follows:

$$\left\| T - T^0 \right\| = \begin{cases} 5.62 & \text{if } m_1 = 1, \ m_2 = 1 \\ 4.60 & \text{if } m_1 = 1, \ m_2 = 2 \\ 5.57 & \text{if } m_1 = 2, \ m_2 = 1 \\ 8.52 & \text{if } m_1 = 2, \ m_2 = 2 \end{cases}$$

The closest harmonic periods are given by $m_1 = 1$, $m_2 = 2$ with $T^* = \begin{bmatrix} 11.75 & 11.75 & 23.5 \end{bmatrix}$.

The time complexity for calculating all feasible candidates for the closest harmonic periods is $\mathcal{O}\left(2^{n-1}\right)$.

5.2 Harmonic Period Assignment with Period Ranges

In many real-time control applications, there exist lower and upper bounds on the possible task periods. For instance, a commonly quoted rule of thumb

[Åström and Wittenmark, 2013] states that the period T of a control task should be chosen such that

$$0.2 \leq \omega_b T \leq 0.6$$

where ω_b is the bandwidth of the closed-loop system.

In more general terms, we require that $T_i \in \left[T_i^l,\, T_i^u\right]$ where T_i^l and T_i^u are the lower and upper period bounds of task τ_i. We assume that $T_i^l \leq T_{i+1}^u$. We want to find all harmonic periods that satisfy the requirements above for all tasks. We have the following theorem.

THEOREM 3
There exist $\tilde{m}_i$, $i \in \{1, 2, \cdots, n-1\}$, satisfying the harmonic period requirement above if and only if

1.
$$\left\lceil \frac{T_j^l}{T_i^u} \right\rceil \leq \prod_{k=i}^{j-1} \tilde{m}_k \leq \left\lfloor \frac{T_j^u}{T_i^l} \right\rfloor$$

for all $i < j$; and

2.
$$\sum_{i=1}^{n} \frac{C_i}{\alpha \prod_{j=0}^{i-1} \tilde{m}_j} \leq 1, \text{ where } \alpha = \min_i \frac{T_i^u}{\prod_{j=0}^{i-1} \tilde{m}_j}.$$

Proof Condition 1) can be understood as an existence condition. As shown in subsection 4.1, the harmonic periods are a function of $\tilde{m}_i$, $i \in \{1, 2, \cdots, n-1\}$, for given execution times $[C_1, C_2, \cdots, C_n]$. Thinking in the n-dimensional space spanned by T_1, T_2, ..., T_n, each set of harmonic periods corresponds to a line through the origin. The existence condition can be understood as an n-dimensional cuboid in that space. Any line that intersects with the cuboid can be represented by a function of $\tilde{m}_i$.

Condition 2) states that the total utilization should be less than or equal to 1. Part of the intersection between the line representing the harmonic condition and the cuboid must satisfy this utilization condition. □

When using Theorem 3 to find all harmonic periods, we can start with 1) to find out all possible sets of $\tilde{m}_i$, and then use condition 2) to verify all of them. However, if

$$\left\lceil \frac{T_j^l}{T_i^u} \right\rceil \ll \left\lfloor \frac{T_j^u}{T_i^l} \right\rfloor$$

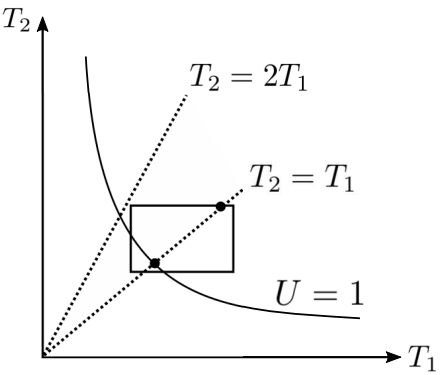

Figure 5.2 Harmonic periods selection for $n = 2$

then there exists a large number of sets of $\tilde{m}_i$ satisfying condition 1). We could choose an equivalent to condition 1), which is

$$\left\lceil \frac{T_{i+1}^l}{T_i^u} \right\rceil \leq \tilde{m}_i \leq \left\lfloor \frac{T_{i+1}^u}{T_i^l} \right\rfloor \text{ and } \alpha \prod_{j=0}^{i-1} \tilde{m}_j \geq T_i^l$$

where α has been defined in condition 2).

We give an intuitive explanation for Theorem 3 when $n = 2$ in Fig. 5.2. The curve represents the utilization bound. For given C_1 and C_2, $\frac{C_1}{T_1} + \frac{C_2}{T_2} \leq 1$ is the region on the right upper side of this curve. The lines through the origin represent different harmonic relations between T_1 and T_2, while the rectangle represents the allowed period ranges. The line segment between the two marked points give all possible harmonic period assignments with utilization ≤ 1.

Considering n tasks, the point that is closest to the origin on each line segment is $T_0 = \tilde{M}C$, where $\tilde{M}$ is calculated by Eq. (4.1) with $\tilde{m}_i$. The utilization when $T = T_0$ is 1. The other point is

$$T_f = \alpha \begin{bmatrix} 1 & \tilde{m}_1 & \tilde{m}_1\tilde{m}_2 & \cdots & \tilde{m}_1\tilde{m}_2\cdots\tilde{m}_n \end{bmatrix} \tag{5.1}$$

The utilization when $T = T_f$ is less than 1. Hence, the harmonic periods can be selected anywhere in $(1 - a)T_0 + aT_f$, with $0 \leq a \leq 1$.

Example 3

Assume that we have three tasks with the same execution times as in Examples 1 and 2, i.e., $C = \begin{bmatrix} 0.9 & 6.3 & 9.1 \end{bmatrix}$. Further assume that we have specified the allowable period ranges as

$$\left[T_1^l, T_1^u\right] = \left[6, 12\right], \ \left[T_2^l, T_2^u\right] = \left[7, 21\right], \ \left[T_3^l, T_3^u\right] = \left[9, 27\right]$$

We then have

$$\left\lceil \frac{T_2^l}{T_1^u} \right\rceil = 0.58, \quad \left\lfloor \frac{T_2^u}{T_1^l} \right\rfloor = 3.5 \Rightarrow m_1 \in \{1,2,3\}$$
$$\left\lceil \frac{T_3^l}{T_2^u} \right\rceil = 0.43, \quad \left\lfloor \frac{T_3^u}{T_2^l} \right\rfloor = 3.86 \Rightarrow m_2 \in \{1,2,3\}$$
$$\left\lceil \frac{T_3^l}{T_1^u} \right\rceil = 0.75, \quad \left\lfloor \frac{T_3^u}{T_1^l} \right\rfloor = 4.5 \Rightarrow m_1 m_2 \in \{1,2,3,4\}$$

We should choose $[m_1, m_2] \in \{[1,2],[2,1],[2,2],[3,1]\}$. When $[m_1, m_2] \in \{[1,1],[1,3]\}$ the utilization condition is not satisfied. We then use Equation (4.1) and (5.1) to calculate the harmonic periods T_0 and T_f. The corresponding period sets are

$$a\begin{bmatrix}0.25 & 0.25 & 0.5\end{bmatrix} + \begin{bmatrix}11.75 & 11.75 & 23.5\end{bmatrix}$$
$$a\begin{bmatrix}0.4 & 0.8 & 0.8\end{bmatrix} + \begin{bmatrix}8.6 & 17.2 & 17.2\end{bmatrix}$$
$$a\begin{bmatrix}0.425 & 0.85 & 1.7\end{bmatrix} + \begin{bmatrix}6.325 & 12.65 & 25.3\end{bmatrix}$$
$$a\begin{bmatrix}0.967 & 2.9 & 2.9\end{bmatrix} + \begin{bmatrix}6.033 & 18.1 & 18.1\end{bmatrix}$$

for any $a \in [0,1]$.

Note that the first solution with $a = 0$ corresponds to the closest periods solution in Example 2. Here we have obtained more solutions that could be further evaluated with regard to, e.g., overall system performance.

The harmonic period assignment using period ranges has previously been solved in [Nasri et al., 2014] and in [Nasri and Fohler, 2015]. In [Nasri et al., 2014] which uses the so called forward calculation approach, the computational complexity is $\mathcal{O}(n)$ when all integer multiples intersect, while the complexity is pseudo-polynomial when all integer multiples do not intersect. In [Nasri and Fohler, 2015], which uses an alternative backward calculation approach, the complexity is pseudo-polynomial when all integer multiples intersect, and when all integer multiples do not intersect the complexity is $\mathcal{O}(n \log n)$.

Theorem 3 provides a unified method for both the case when all integer multiples intersect and when all integer multiples do not intersect, or a combination of those two situations. Since the algorithm is only run once from 1 to n to calculate $\tilde{m}_i$ and to evaluate

$$\sum_{i=1}^{n} \frac{C_i}{\alpha \prod_{j=0}^{i-1} \tilde{m}_j},$$

the time complexity is $\mathcal{O}(n)$ for each such calculation. Assuming that $\tilde{m}_i$ can take at most m different values, the time complexity to check the utilization condition is $\mathcal{O}(n^m)$. The overall time complexity is hence $\mathcal{O}(n^m)$.

6

Co-Design and Evaluation

In this section we apply harmonic period assignment in control–scheduling co-design and compare the resulting performance to state-of-the-art non-harmonic co-design.

6.1 Co-Design Procedure

As a starting point for the co-design, we find a set of non-harmonic, real-valued task periods and corresponding controllers using the sequential search optimization method in [Xu et al., 2014]. We then harmonize these periods using Theorem 2 or Theorem 3 and enumerate all possible combinations of the harmonic factors. For each harmonic period assignment, we redesign each controller based on the new period and the new (now constant) control delay. Finally, we evaluate the combined control performance of all cases and pick the best result.

All LQG controllers are designed using the `lqgdesign` command in Jitterbug [Cervin et al., 2002]. To make a fair comparison between harmonic and non-harmonic designs, we make the following assumptions for the LQG control design:

- The harmonic control design takes the constant delay into account.
- The non-harmonic control design takes the delay distribution due to task scheduling into account, resulting in a jitter-robust controller with fixed parameters [Xu et al., 2014]. The delay distribution is found through a schedule simulation in TrueTime [Cervin et al., 2002].
- The CPU utilization is 1 for both harmonic scheduling and non-harmonic scheduling. In non-harmonic scheduling, if the response time is greater than the period, the response time distribution used in the control design is truncated to the period length.

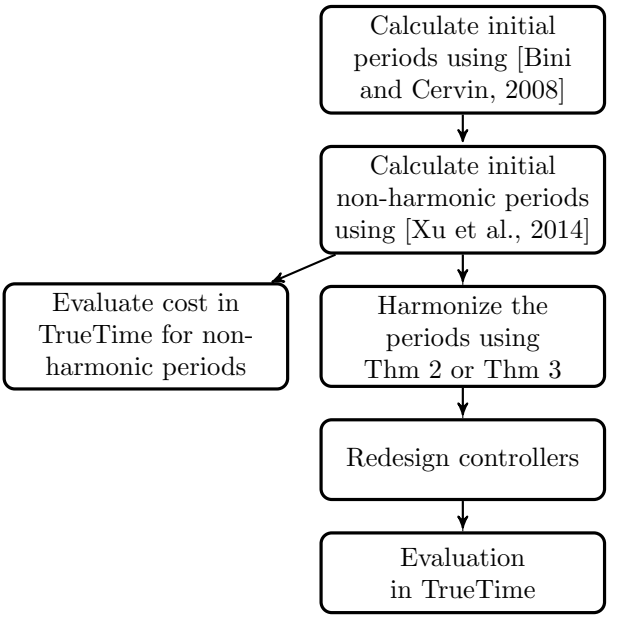

Figure 6.1 Co-design and evaluation procedure for non-harmonic and harmonic designs.

The cost function (Eq. (3.2)) for each controller under each scheduling scenario is evaluated using the TrueTime toolbox [Cervin et al., 2002]. Using TrueTime it is possible to simulate real-time kernels with tasks executing, e.g., controller, code under the control of an arbitrary scheduling policy and interacting with continuous-time dynamic models representing the physical process under control. It is also possible to numerically evaluate the same quadratic cost functions that are evaluated analytically using Jitterbug. However, using TrueTime one is not, as in Jitterbug, restricted to scenarios in which the delays are independent, e.g., in the non-harmonic controller evaluations.

The design and evaluation procedure is summarized in the flow diagram in Fig. 6.1.

6.2 A Simple Co-Design Example

For a simple co-design example, we choose three plants

$$P_1 = \frac{2}{s^2},\ P_2 = \frac{1}{s^2 - 3},\ P_3 = \frac{1}{s(s+1)}$$

to be controlled by three tasks τ_1, τ_2, τ_3. τ_1 has the highest priority and τ_3 has the lowest priority. The execution times are given as $C_1 = 0.1$, $C_2 = 0.12$, $C_3 = 0.14$. The LQG control cost parameters are given as $Q_{1c} = BB^T$, and $Q_{2c} = 0.01\text{tr}\,(Q_{1c})$.

Initial non-harmonic periods are calculated by the following initialization procedure (cf. [Xu et al., 2014]):

1. For task i, assume that the LQG cost can be approximated by a linear function of the period T_i and of the delay L_i,

$$J_i = \alpha_i T_i + \beta_i L_i$$

   Evaluate the sensitivity coefficients α_i and β_i at the point where $T_i = C_i$ and $L_i = C_i$ using numerical linearization and Jitterbug. Then use the period assignment method in [Bini and Cervin, 2008] to minimize the LQG cost under the simplifying assumption that these are the true cost functions.

2. Use the Sequential Search method in [Xu et al., 2014] to find the stochastic LQG periods.

The initial non-harmonic periods are $T_1^* = 0.3017$, $T_2^* = 0.4089$, $T_3^* = 0.4478$ with initial cost $J^* = 2.01$. Using Theorem 2, we find the harmonic factors

$$m_1 \in \left\{ \left\lfloor \frac{T_2^*}{T_1^*} \right\rfloor, \left\lceil \frac{T_2^*}{T_1^*} \right\rceil \right\} = \{1, 2\}$$
$$m_2 \in \left\{ \left\lfloor \frac{T_3^*}{T_2^*} \right\rfloor, \left\lceil \frac{T_3^*}{T_2^*} \right\rceil \right\} = \{1, 2\}$$

The LQG cost is then evaluated in TrueTime for the following cases:

- **No offset.** Assume a constant delay equal to the job response time R_i to design and evalute the LQG controllers.
- **Offset.** Add the start latency S_i as a release offset to each task; then design and evalute the LQG controllers for the constant delay $R_i - S_i$.

The resulting periods and LQG costs are shown in the table below. No offset means sampling happens at job release time, while offset means that sampling happens at the job start.

$\{m_1, m_2\}$	T_1	T_2	T_3	$J_{\text{no offset}}$	J_{offset}
$\{1, 1\}$	0.36	0.36	0.36	1.99	1.57
$\{1, 2\}$	0.29	0.29	0.58	2.11	1.57
$\{2, 1\}$	0.23	0.46	0.46	1.90	1.33
$\{2, 2\}$	0.20	0.39	0.78	2.56	1.75

The result shows that when we approximate the initial non-harmonic task periods with harmonic ones, the costs can be better or worse than the initial cost. However, when we utilize release offsets, the cost is clearly better than the initial cost in all the cases, with the best case obtained for $\{m_1, m_2\} = \{2, 1\}$.

Continuing the same example, we also show how to find harmonic periods when there are constraints on the allowable period ranges, and then evaluate the LQG control performance, using the same plants with the same execution times as before. Assuming that the period T_i can only be chosen from $[0.6T_i^*, 1.7T_i^*]$, it follows from Theorem 3 that the possible periods are when $\{m_1, m_2\}$ is $\{1, 1\}$, $\{1, 2\}$, $\{2, 1\}$ (as shown above), or $\{3, 1\}$ (as shown below).

$\{m_1, m_2\}$	T_1	T_2	T_3	$J_{\text{no offset}}$	J_{offset}
$\{3, 1\}$	0.19	0.56	0.56	3.36	2.25

It should, however, be noted that the harmonic period assignment with the lowest control cost is not necessarily restricted to the above cases. The control cost function could have a form so that the harmonic period assignment with the lowest cost is not among those assigments that are close to the initial non-harmonic periods in the Euclidean sense. However, as shown in the general evaluation the proposed approach obtains gives considerably better control performance than the non-harmonic case.

6.3 Randomly Generated Example

To see if the good results shown in the simple example above holds in more general cases, we randomly generate sets of three plants from the following three plant families:

- Family I: All plants have two stable poles and each plant is drawn from $P_1(s)$ and $P_2(s)$ with equal probability where

$$P_1(s) = \frac{1}{(s + a_1)(s + a_2)}, \; P_2(s) = \frac{1}{s^2 + 2\zeta\omega s + \omega^2}$$

  with $a_1, a_2, \zeta \in \text{unif}(0, 1)$, $\omega \in \text{unif}(0, 1)$.

- Family II: All plants have two stable or unstable poles, with each plant drawn with equal probability from

$$P_3(s) = \frac{1}{(s + a_1)(s + a_2)}, \; P_4(s) = \frac{1}{s^2 + 2\zeta\omega s + \omega^2}$$

  with $a_1, a_2, \zeta \in \text{unif}(-1, 1)$, $\omega \in \text{unif}(0, 1)$.

- Family III: All plants have three stable or unstable poles, with each plant drawn with equal probability from

$$P_5(s) = \frac{1}{(s+a_1)(s+a_2)(s+a_3)}$$

$$P_6(s) = \frac{1}{(s^2+2\zeta\omega s+\omega^2)(s+a_3)}$$

with $a_1, a_2, a_3, \zeta \in \text{unif}(-1,1)$, $\omega \in \text{unif}(0,1)$.

We randomly generated 10 sets of plants for each family. For the LQG controllers, we used the design parameters $Q_{1c} = BB^T$, and $Q_{2c} = 0.01\text{tr}\,(Q_{1c})$. The task execution times were randomly generated from $C_1 \in \text{unif}(0.09, 0.11)$, $C_2 \in \text{unif}(0.11, 0.13)$, $C_3 \in \text{unif}(0.13, 0.15)$. The nominal task utilizations U_i^{nom} were generated using an n-dimensional uniform distribution with total utilization 1. Task 1 has the highest priority, while task 3 has the lowest priority.

The optimization procedure to assign initial non-harmonic periods is the same as in the previous section. We find the four closest harmonic period sets to the initial periods using Theorem 2. For the harmonic periods case, the response time of each task is constant. Using this constant response time as delay, the LQG controllers are designed and the corresponding costs are evaluated. In the table below, the minimum cost, out of the four cases with harmonic periods, is given. We then add an offset to each task in order to obtain a shorter delay for task 2 and 3. The length of the offset is the start latency of each task. The constant delay is $R_i - S_i$. The we design controller and evaluate the LQG costs for this constant delay. The overall results, averaged over 10 generated plant sets for each family, are summarized below.

Family	I	II	III
non-harmonic tasks	2.92	8.61	29.46
harmonic tasks	2.53	5.17	17.76
harmonic tasks with offsets	2.03	3.91	15.43

In the above table, we compare design the LQG controllers and evaluate the LQG costs as follows:

- **Non-harmonic tasks.** The delay distribution is truncated to the interval $\left[R_i^{\text{best}}, R_i^{\text{best}} + T_i\right]$. The probability of a response time greater than $R_i^{\text{best}} + T_i$ is added to the probability mass function (PMF) at $R_i^{\text{best}} + T_i$. We then use the truncated delay distribution to design stochastic LQG controllers. The LQG cost is evaluated in TrueTime.

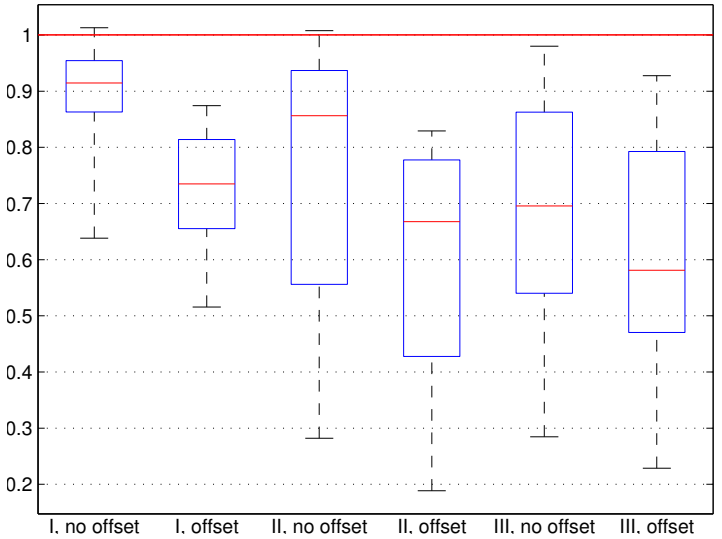

Figure 6.2 Normalized costs for harmonic tasks without and with offsets

- **Harmonic tasks.** We calculate 2^{n-1} sets of harmonic periods. For each set, LQG controllers with constant delays equal to R_i are designed and evaluated in TrueTime. The period set giving the smallest cost is selected.

- **Harmonic tasks with offsets.** For each set of harmonic periods, LQG controllers with constant delays equal to $R_i - S_i$ are designed and evaluated in TrueTime. The period set giving the smallest cost is selected.

We normalize the costs for non-harmonic tasks to 1 for each plant set, then normalize each cost for harmonic tasks without or with offsets, compared with corresponding non-harmonic tasks cost. The box plot is shown in Fig. 6.2.

In Family III, the likelihood that the plants are unstable, and, hence, more sensitive to delays and delay jitter, is larger, and therefore the control cost is considerably higher than for Family I and II. When the periods are harmonic without offset, the costs are higher than in the non-harmonic case. The reason for this is that the increase in cost caused by the period perturbation is not compensated for by the decrease in cost caused by the jitter-free, but possibly large delays. The best results are obtained for the harmonic tasks with offsets. In this case the increase in cost caused by the period perturbation is small compared to the decrease in cost caused by the smaller and jitter-free delays.

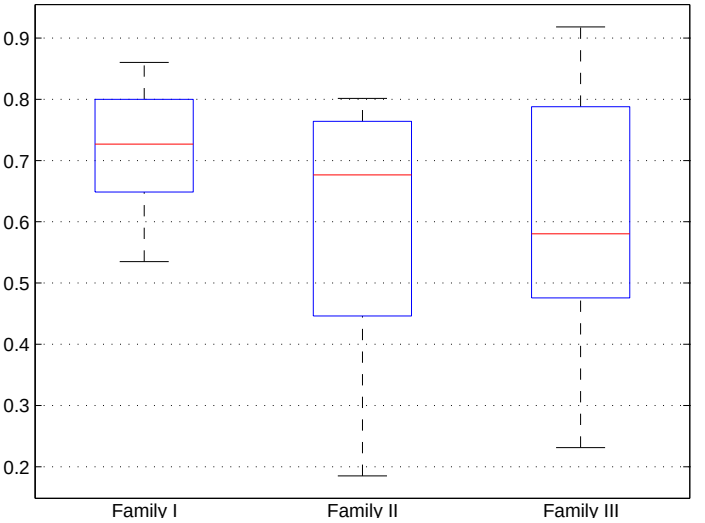

Figure 6.3 Normalized costs for non-constant execution times (different scale with Fig. 6.2)

The evaluation above is based on the assumption that the execution time are constant. However, in reality this is seldom the case. To investigate the effect of varying execution times we design the controllers using the harmonic task with offset method assuming that the execution times are constant. When we evaluate the performance we let the execution time vary from job to job according to Unif$(0.9C_i, C_i)$. The costs now become

Family	I	II	III
Cost	2.01	3.88	15.33

i.e., even smaller than before. We also make the box plot of costs for non-constant execution times compared with non-harmonic tasks costs in Fig. 6.3. This is, however, not surprising since the average delay is shorter than the constant delay in the constant execution time case.

7

Conclusion

In this paper we have investigated the harmonic scheduling and control co-design problem. Through an extensive evaluation it was shown that co-design using harmonic controller task periods gives better control performance than using non-harmonic periods, when task offsets are added. The reason for this is the fact that under harmonic scheduling the response times and start latencies for each task are constant, assuming that the task execution times are constant. This can be exploited in the LQG control design through the constant control delays that it gives rise to. However, as shown in the evaluation also in the case when the task execution times vary slightly from job to job the proposed co-design method gives good results.

In order to implement the co-design method it is necessary to be able to find the harmonic periods and to calculate the task response times. A new method for calculating the response times has been presented that has lower complexity than earlier methods. Also, two heuristic approaches to harmonic task period assignment have been presented. One method for finding the closest harmonic periods to a set of initial periods and one method for finding all possible harmonic period assignments that satisfy constraints on allowable task period ranges.

Acknowledgments

The authors would like to thank the LCCC Linnaeus Center and the ELLIIT Excellence Center at Linköping and Lund.

Bibliography

Åström, K. J. and B. Wittenmark (2013). *Computer-controlled systems: theory and design*. Courier Corporation.

Åström, K. J. and B. Wittenmark (1997). *Computer-controlled systems : theory and design (3 ed.)* Prentice-Hall.

Bini, E. and A. Cervin (2008). "Delay-aware period assignment in control systems". In: *Proceedings of the* 29^{th} *IEEE Real-Time Systems Symposium*. Barcelona, Spain, pp. 291–300.

Bonifaci, V., A. Marchetti-Spaccamela, N. Megow, and A. Wiese (2013). "Polynomial-time exact schedulability tests for harmonic real-time tasks". In: *Real-Time Systems Symposium (RTSS), 2013 IEEE 34th*, pp. 236–245. DOI: 10.1109/RTSS.2013.31.

Brocal, V., P. Balbastre, R. Ballester, and I. Ripoll (2011). "Task period selection to minimize hyperperiod". In: *Emerging Technologies & Factory Automation (ETFA), 2011 IEEE 16th Conference on*. IEEE, pp. 1–4.

Busquets-Mataix, J. V., J. J. Serrano, R. Ors, P. Gil, and A. Wellings (1996). "Using harmonic task-sets to increase the schedulable utilization of cache-based preemptive real-time systems". In: *Real-Time Computing Systems and Applications, 1996. Proceedings., Third International Workshop on*. IEEE, pp. 195–202.

Cervin, A., D. Henriksson, B. Lincoln, and K.-E. Årzén (2002). "Jitterbug and TrueTime: Analysis tools for real-time control systems". In: *Proceedings of the 2nd Workshop on Real-Time Tools*. Copenhagen, Denmark.

Cervin, A., D. Henriksson, B. Lincoln, J. Eker, and K.-E. Årzén (2003). "How does control timing affect performance?" *IEEE Control Systems Magazine* **23**:3, pp. 16–30.

Eisenbrand, F. and T. Rothvoss (2008). "Static-priority real-time scheduling: response time computation is np-hard". In: *Real-Time Systems Symposium, 2008*, pp. 397–406. DOI: 10.1109/RTSS.2008.25.

Eker, J., P. Hagander, and K.-E. Årzén (2000). "A feedback scheduler for real-time controller tasks". *Control Engineering Practice* **8**:12.

Goswami, D., M. Lukasiewycz, R. Schneider, and S. Chakraborty (2012). "Time-triggered implementations of mixed-criticality automotive software". In: *Proceedings of the Conference on Design, Automation and Test in Europe*. Dresden, Germany, pp. 1227–1232.

Han, C.-C. and H.-y. Tyan (1997). "A better polynomial-time schedulability test for real-time fixed-priority scheduling algorithm". In: *Proceedings of the* 18^{th} *IEEE Real-Time Systems Symposium*. San Francisco, (CA), U.S.A., pp. 36–45.

Lehoczky, J. P., L. Sha, and Y. Ding (1989). "The rate-monotonic scheduling algorithm: exact characterization and average case behavior". In: *Proceedings of the* 10^{th} *IEEE Real-Time Systems Symposium*. Santa Monica (CA), U.S.A., pp. 166–171.

Nasri, M. and G. Fohler (2015). "An efficient method for assigning harmonic periods to hard real-time tasks with period ranges". In: *Real-Time Systems (ECRTS), 2015 27th Euromicro Conference on*, pp. 149–159. DOI: 10.1109/ECRTS.2015.21.

Nasri, M., G. Fohler, and M. Kargahi (2014). "A framework to construct customized harmonic periods for real-time systems". In: *Real-Time Systems (ECRTS), 2014 26th Euromicro Conference on*, pp. 211–220. DOI: 10.1109/ECRTS.2014.31.

Papadimitriou, C. H. (1981). "On the complexity of integer programming". *Journal of the ACM (JACM)* **28**:4, pp. 765–768.

Ripoll, I. and R. Ballester-Ripoll (2013). "Period selection for minimal hyperperiod in periodic task systems". *Computers, IEEE Transactions on* **62**:9, pp. 1813–1822.

Samii, S., A. Cervin, P. Eles, and Z. Peng (2009a). "Integrated scheduling and synthesis of control applications on distributed embedded systems". In: *Proc. Design, Automation & Test in Europe (DATE)*.

Samii, S., P. Eles, Z. Peng, and A. Cervin (2009b). "Quality-driven synthesis of embedded multi-mode control systems". In: *Design Automation Conference, 2009. DAC'09. 46th ACM/IEEE*. IEEE, pp. 864–869.

Seto, D., J. P. Lehoczky, L. Sha, and K. G. Shin (1996). "On task schedulability in real-time control systems". In: *Proceedings of the* 17^{th} *IEEE Real-Time Systems Symposium*. Washington, DC, USA, pp. 13–21.

Shih, C.-S., S. Gopalakrishnan, P. Ganti, M. Caccamo, and L. Sha (2003). "Scheduling real-time dwells using tasks with synthetic periods". In: *Real-Time Systems Symposium, 2003. RTSS 2003. 24th IEEE*. IEEE, pp. 210–219.

Wittenmark, B., J. Nilsson, and M. Törngren (1995). "Timing problems in real-time control systems". In: *In Proceedings of the American Control Conference*. Citeseer.

Xu, Y., K.-E. Årzén, E. Bini, and A. Cervin (2014). "Response time driven design of control systems". In: *Proceedings of the* 19^{th} *World Congress of the International Federation of Automatic Control*. Cape Town, South Africa, pp. 6098–6104.

Xu, Y., K.-E. Årzén, A. Cervin, E. Bini, and B. Tanasa (2015). "Exploiting job response-time information in the co-design of real-time control systems". In: *Embedded and Real-Time Computing Systems and Applications (RTCSA), 2015 IEEE 21st International Conference on*, pp. 247–256. DOI: 10.1109/RTCSA.2015.23.

Zhang, F., K. Szwaykowska, W. Wolf, and V. Mooney (2008). "Task scheduling for control oriented requirements for cyber-physical systems". In: *Real-Time Systems Symposium, 2008*. IEEE, pp. 47–56.