DMAC: Deadline-Miss-Aware Control

Paolo Pazzaglia, Claudio Mandrioli, Martina Maggio and Anton Cervin

Scuola Superiore Sant’Anna, Pisa, Italy
Department of Automatic Control, Lund University, Sweden

paolo.pazzaglia@santannapisa.it

ECRTS 2019, Stuttgart - July 9, 2019
• **Starting problem:** Optimal design of a control task to be run alongside a pre-existing real-time system

• **Co-design:** combining (conflicting) requirements from control theory and real-time systems

• **Hard deadline model** $\rightarrow$ periods are constrained to be longer than WCRT
An alternative approach

- Hard deadline requirements may be too tight for many real-world systems
- "Good control design" should guarantee robustness to a limited number of deadline misses
- ... And overload conditions are relatively rare

- **Idea**: Explore the interval of periods $T_d < WCRT$, explicitly taking into account the probability of **deadline misses**
Our proposal

- Leverage probability of **sequences of deadline miss and hits** to build an **optimal** (on average) **fixed** controller

- Sequences obtained by simulation, with formal guarantees coming from the **scenario theory**
### Task model

- **Initial taskset** \( \Gamma' = \{\tau_1, \tau_2, \ldots, \tau_N\} \), fixed priority

- **Control task** \( \tau_d \) to be added as the one with **lowest priority**

- **\( \tau_i = \{C_i, f_i^C, D_i, T_i\} \)**

- Execution time described as a **random independent** variable with known probability density, implicit deadline \((D_i = T_i)\)

- Period \( T_d \) of control task \( \tau_d \) is our design variable
System model

- Plant to be controlled is LTI MIMO, with white noise disturbance
- Periodic control task $\tau_d$ executes under **Logical Execution Time** paradigm

$$
\begin{align*}
\text{Read from sampler} & \quad \tau_d \\
kT_d & \quad (k + 1)T_d \\
\text{Write to actuator} & \quad u[k-1] \\
& \quad u[k]
\end{align*}
$$
Deadline miss handling

- Focus on three strategies: Kill – Skip Next – Queue(1)

- **Kill** strategy

  ![Kill strategy diagram]

- **Skip-next** strategy

  ![Skip-next strategy diagram]

- **Queue(1)** strategy

  ![Queue(1) strategy diagram]
Effects of deadline misses

- Deadline misses produce jitter in the control output pattern.
- The dynamic of the system behaves as a switched-linear system.
- Extract timing properties → Delay $\sigma_k$ and hold $h_k$.

- Delay ($\sigma_k$) is computed from response time.
- Hold ($h_k$) depends on the next update.

- Remark: Killed, skipped and overwritten jobs do not contribute to control!
Effects of deadline misses

• We associate \((\sigma_n, h_n)\) to each **valid control job**
  - depending on **specific following subsequence** and d.m. strategy

Ex: kill strategy

\[
\begin{align*}
\sigma_n &= T_d \\
h_n &= 2T_d
\end{align*}
\]

• What is the probability of having a specific \((\sigma_n, h_n)\)? We need to know how often each possible subsequence occurs

• Analytic approach is not available...

• **Focus on estimation with robustness**
• **Alternative approach**: Evaluation of probability of deadline misses using scenario theory

Optimization problem

\[ \tau_i = \{C_i, f_i^C, D_i, T_i\} \]

Task set model

\[ n_{sim} \]

Worst case sequence

Analysis approach

Approach by simulation

Sequence probabilities

Experienced sequence probabilities

Robustly guaranteed by scenario theory
Deadline-Miss-Aware Control

• Ideally, optimal control should be adaptive and clairvoyant $\rightarrow$ not realizable in real applications

• **Fixed robust control** based on statistical properties of the system: Deadline-Miss-Aware Control (DMAC)

$$ u(t_n) = -\bar{L} \hat{x}(t_n) $$

• Matrix $\bar{L}$ built using *stochastic Riccati equation*, based on the possible values of $(\sigma_n, h_n)$ and their probability

• On average, it works as the ideal adaptive clairvoyant controller
Evaluating the performance: JitterTime

- The performance of the controlled system for a given schedule is computed using **JitterTime** [*]

- Matlab-based analysis tool inspired by **Jitterbug** and **TrueTime**

- Used to analyze performance in scenarios with non-ideal timing, continuous and discrete blocks

- Transitions with arbitrary rules

JitterTime is freeware! Online manual: [http://www.control.lth.se/jittertime](http://www.control.lth.se/jittertime)

Experimental evaluation

- Starting taskset randomly generated with UUnifast
- Generate WCET and probability distributions for all tasks
- Target task $\tau_d$ with WCRT $\approx 2$ sec, interval of interest of $T_d = [0.5, 2]$ sec
- Scenario theory parameters: $\epsilon = 0.003$, $\beta = 0.01 \rightarrow n_{sim} = 1533$
- Scheduling obtained using an ad-hoc simulator using the three different deadline miss strategies – kill, skip-next, queue(1)
- Design controller with DMAC using worst-case sequence
- Performance computed using JitterTime
DMAC design **outperforms classic control design** for all chosen deadline miss strategies

- Limited gap between maximum and minimum $\rightarrow$ good robustness
Experimental evaluation

- Testing DMAC with different initial taskset configurations

- It is not simple to define which deadline miss handling strategy is the best one
  - Depends on the system to be controlled

- Choosing the worst-case sequence differently may affect the overall control performance
  - Require more tests
Conclusion

- **Problem:** optimal design of controller that can miss some deadline, with probabilistic execution times

- Three deadline miss strategies: kill, skip-next and queue(1)

- Deadline miss probabilities of subsequences of jobs extracted using Scenario Theory

- Proposed **DMAC:** Deadline-Miss-Aware Control design

- Experimental testing showed that it easily outperforms standard design techniques with good robustness

**Giveaway message:** control systems may be efficaciously designed to be robust to deadline misses
Any questions?

Thank you!

paolo.pazzaglia@santannapisa.it

Want to know more details?
Check our paper! →

DMAC: Deadline-Miss-Aware Control

Paolo Pazzaglia
Scuola Superiore Sant’Anna, Pisa, Italy
Department of Automatic Control, Lund University, Sweden
paolo.pazzaglia@sassp.it

Claudio Mandrioli
Department of Automatic Control, Lund University, Sweden
claudio.mandrioli@control.lth.se

Martina Maggio
Department of Automatic Control, Lund University, Sweden
martina.maggio@control.lth.se

Anton Cervin
Department of Automatic Control, Lund University, Sweden
anton.cervin@control.lth.se

Abstract
The real-time implementation of periodic controllers requires solving a co-design problem, in which the choice of the controller sampling period is a crucial element. Classic design techniques limit the period exploration to safe values, that guarantee the correct execution of the controller alongside the remaining real-time load, i.e., ensuring that the controller worst-case response time does not exceed its deadline. This paper presents DMAC, the first formally-grounded controller design strategy that explores shorter periods, thus explicitly taking into account the possibility of missing deadlines. The design leverages information about the probability that specific sub-sequences of deadline misses are experienced. The result is a fixed controller that on average works as the ideal clairvoyant time-varying controller that knows future deadline hits and misses. We obtain a safe estimate of the hit and miss events using the scenario theory, that allows us to provide probabilistic guarantees. The paper analyzes controllers implemented using the Logical Execution Time paradigm and three different strategies to handle deadline miss events: killing the job, letting the job continue but skipping the next activation, and letting the job continue using a limited queue of jobs. Experimental results show that our design proposal, i.e., exploring the space where deadlines can be missed and handled with different strategies, greatly outperforms classical control design techniques.

2012 ACM Subject Classification Computing methodologies → Computational control theory; Computer systems organization → Embedded software; Software and its engineering → Real-time systems software; Theory of computation → Stochastic control and optimization

Keywords and phrases Weakly-Hard Real-Time Systems, Deadline Miss Handling, Control Design