Beyond the Weakly Hard Model: Measuring the Performance Cost of Deadline Misses

Authors:

Paolo Pazzaglia, Luigi Pannocchi, Alessandro Biondi, Marco Di Natale

Scuola Superiore Sant'Anna, Pisa

paolo.pazzaglia@santannapisa.it

Barcelona, ECRTS, July 6th, 2018

Introduction

- Embedded systems with control tasks may face **overload** conditions (e.g. automotive)

- Common (practical) approach: running at a high rate and allowing some **deadline miss** is an acceptable compromise

How to study performance evolution under overload conditions?

- **Weakly Hard real-time systems**: allowing a limited number of deadline misses
  - **(m,k)**: at most **m** deadlines are missed every **k** activations

- (m,k) constraints can be extracted with TWCA

Weakly hard model limitations

- (m,k) constraint is not enough descriptive…

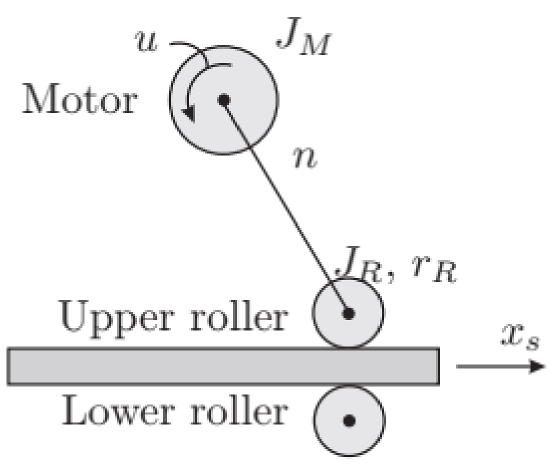
- (m,k) constraint leads to a **binary** model (either pass or fail)
  - Easy to define stability guarantees
  - No information about **performance** of different patterns
  - Difficult to extract an **ordering** between constraints

- No relation with the system state**:**
  - Deadline misses may have different effects (transients vs steady state)

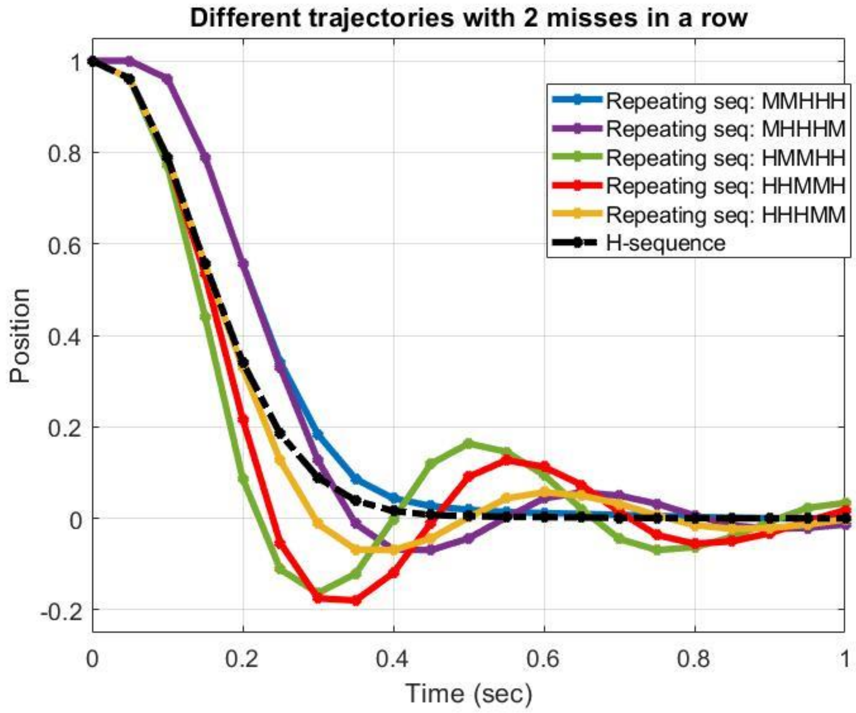
Weakly hard model limitations

Changing the pattern of H/M deadlines may lead to different performance values!

Assumption: When a deadline is missed, the control output is not updated



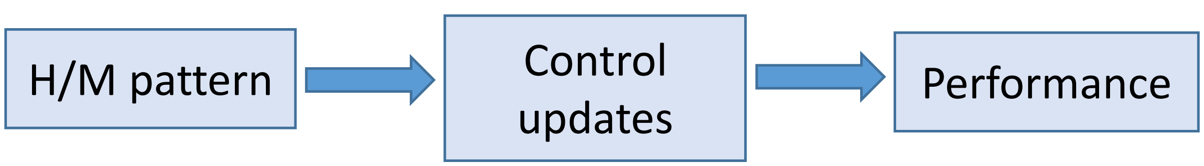
$T = 50\,ms;\ \ D = 0.7 * T$



A new model for performance analysis

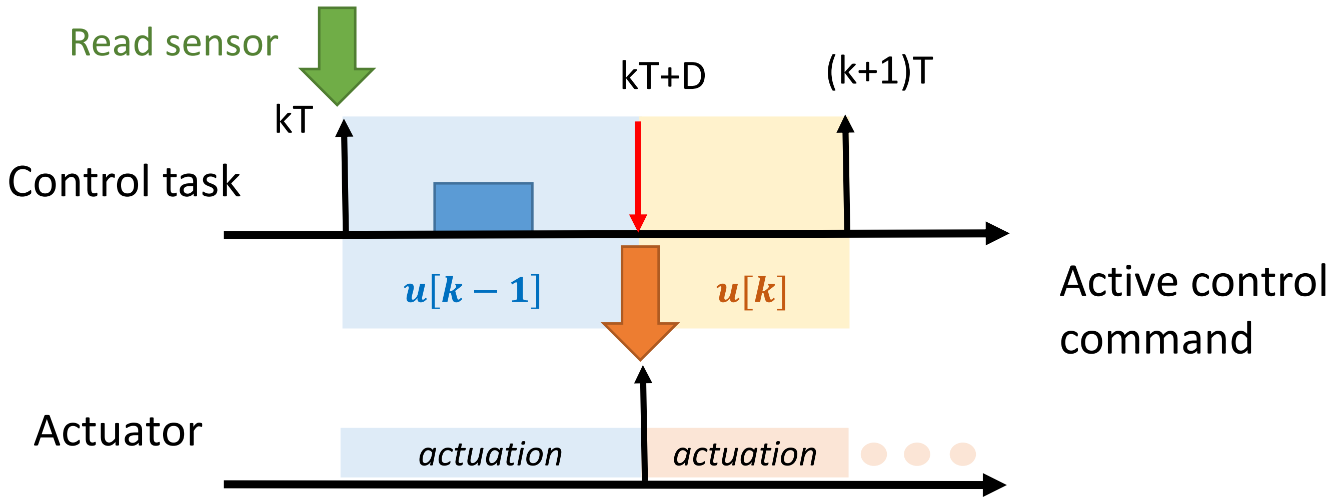
- Goal: Developing a new model for studying:
  - How the **performance** change with different **patterns of missed deadlines** that satisfy a given (m,k) constraint
  - **Worst guaranteed performance**
  - Different policy at deadline miss (continue or kill?)

- Merging real-time analysis with control system dynamics and performance analysis



System model

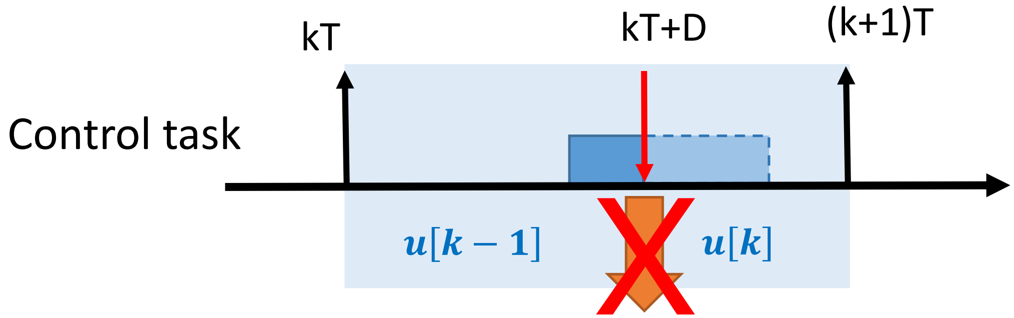
- **Linear Time Invariant** plant, MIMO
- Periodic control of period T_i and deadline $D_i \leq T_i$
- State-feedback control: $u[k] = K(r[k] - x[k])$



State update function: $\mathrm{x}[\mathrm{k}+1] = \mathrm{A_d x[k]} + \mathrm{B_{d1}} u[k-1] + B_{d2} u[k]$

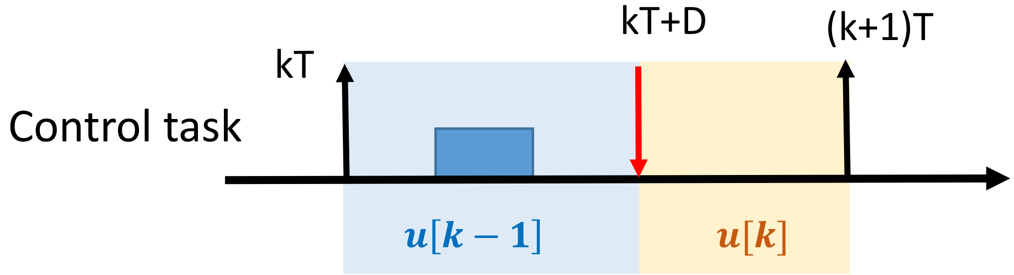
- Similar to **LET** model: trading jitter for latency

Missing a deadline



- Missing a deadline means **missing** an actuator command **update**
- Chosen strategy: **keep** the previous actuation value
- Problem: The actuator uses a control output that is not related with the current state
  - Control output is no more «fresh»
- **The system dynamics changes!**

Update freshness: definition



- **Update freshness Δ** of the control output
  - $\Delta = 0$ if job completes before the deadline
  - Otherwise, Δ equals to the «ageing steps» of the control output

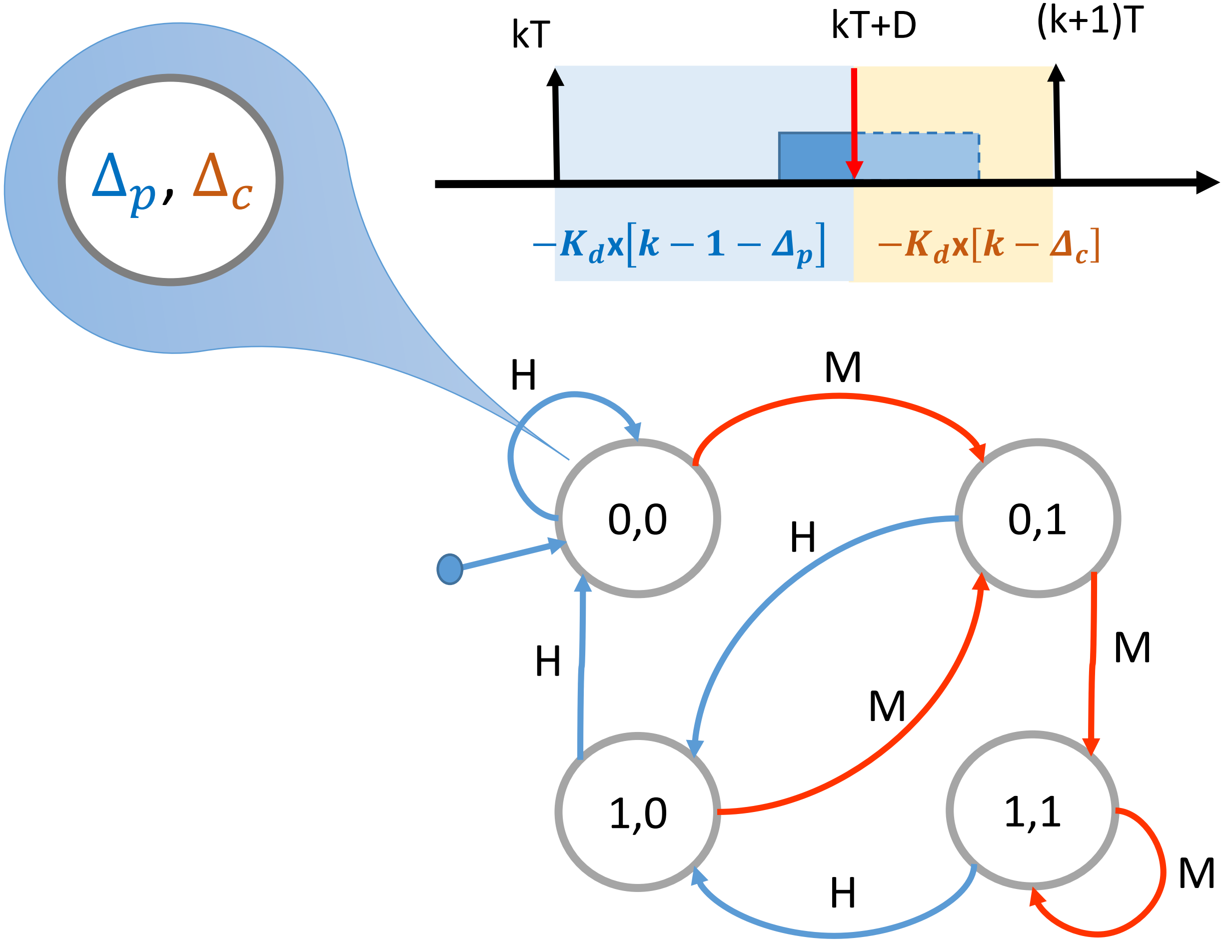
$$x[k+1] = A_d x[k] + B_{d1} u[k-1] + B_{d2} u[k]$$

$$u[k-1] = -K_d x[k-1-\Delta_p]$$

$$u[k] = -K_d x[k-\Delta_c]$$

- Freshness is independent of control law and controlled system!
- Different effects changing *deadline miss handling*

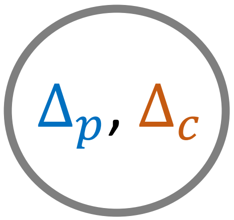
Update freshness: Continue strategy

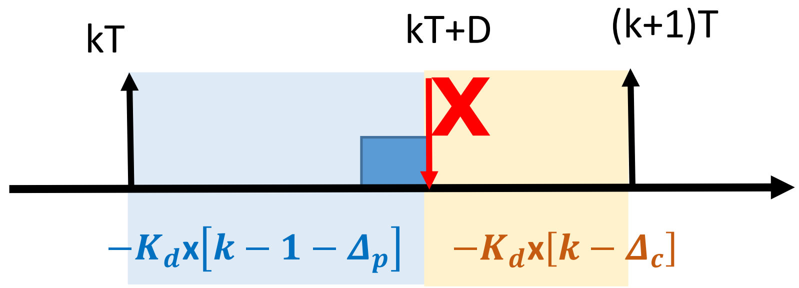


* $BCRT \leq D_i$
* $WCRT < T_i + D_i$

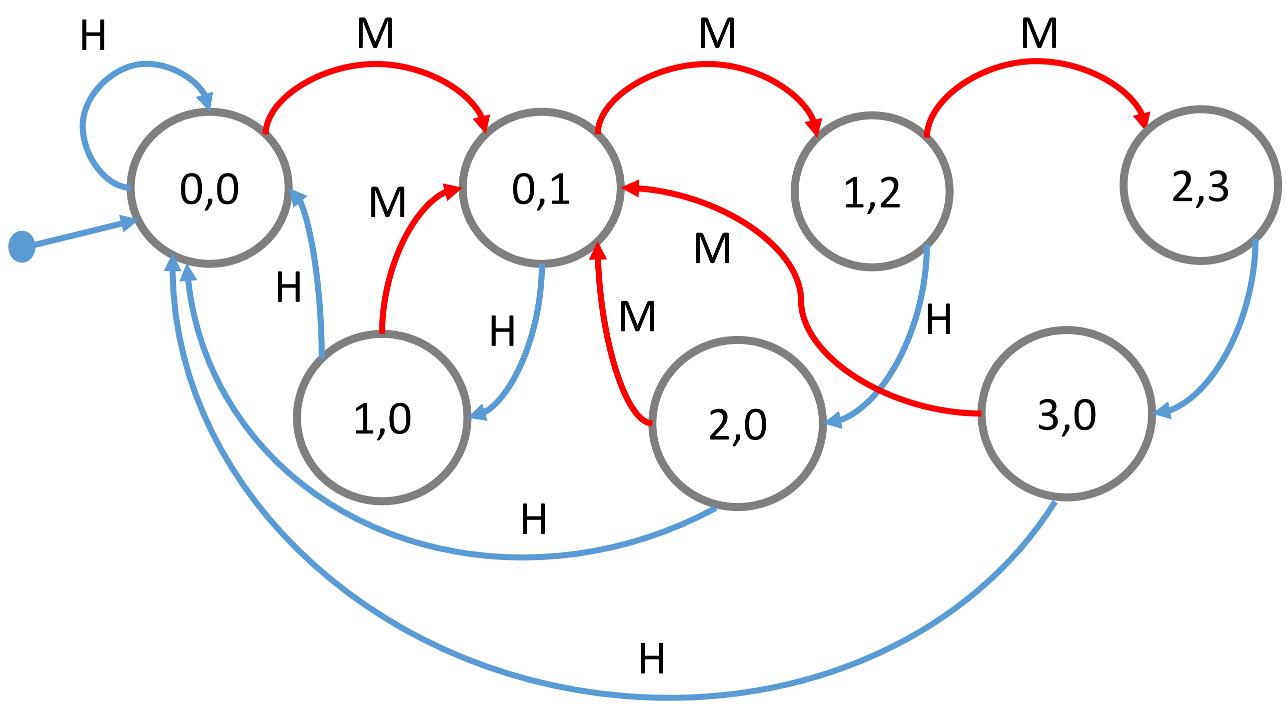
See Algorithm 1 in the paper for more details

Update freshness: Kill strategy





$* \; BCRT \leq D_i$



In this example, maximum number of consecutive deadline misses is equal to 3

State update matrix

- System dynamics as a function of freshness pairs

$$x[k+1] = A_d x[k] - B_{d1} K_d x\left[k - 1 - \Delta_p\right] - B_{d2} K_d x[k - \Delta_c]$$

- Augmented state vector $\xi[k]$

$$\xi[k] = [x[k]; x[k-1]; \ldots. x[k - \Delta_{max} - 1]]$$

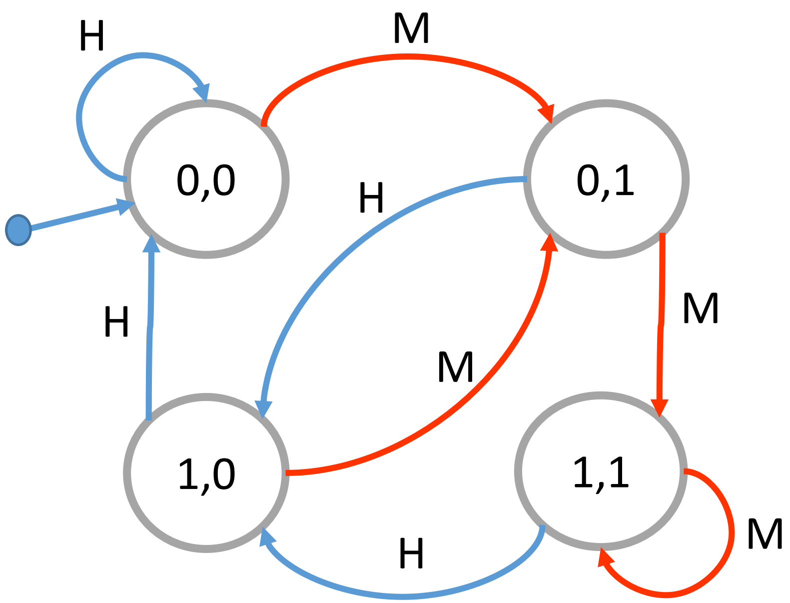
- We can write the system dynamics as: $\xi[k+1] = \Phi(\Delta_p, \Delta_c)\, \xi[k]$

- **State update matrix** $\Phi(\Delta_p, \Delta_c)$

$$\Phi(\Delta_p, \Delta_c) = \begin{bmatrix} A_d & \cdots & -B_{d2}K_d & \cdots & -B_{d1}K_d & \cdots \\ I_n & 0_n & \cdots & \cdots & \cdots & \\ 0_n & I_n & 0_n & & \cdots & \cdots \\ \vdots & \vdots & \vdots & \ddots & \cdots & \cdots \end{bmatrix}$$

State update matrix: an example

Example:



$$\mathbf{\Phi}(0,0) = \begin{bmatrix} A_d - B_{d2}K_d & -B_{d1}K_d & \mathbf{0}_n \\ \mathbf{I}_n & \mathbf{0}_n & \mathbf{0}_n \\ \mathbf{0}_n & \mathbf{I}_n & \mathbf{0}_n \end{bmatrix}$$

$$\mathbf{\Phi}(0,1) = \begin{bmatrix} A_d & -(B_{d1}+B_{d2})K_d & \mathbf{0}_n \\ \mathbf{I}_n & \mathbf{0}_n & \mathbf{0}_n \\ \mathbf{0}_n & \mathbf{I}_n & \mathbf{0}_n \end{bmatrix}$$

$$\mathbf{\Phi}(1,0) = \begin{bmatrix} A_d - B_{d2}K_d & \mathbf{0}_n & -B_{d1}K_d \\ \mathbf{I}_n & \mathbf{0}_n & \mathbf{0}_n \\ \mathbf{0}_n & \mathbf{I}_n & \mathbf{0}_n \end{bmatrix}$$

$$\mathbf{\Phi}(1,1) = \begin{bmatrix} A_d & -B_{d2}K_d & -B_{d1}K_d \\ \mathbf{I}_n & \mathbf{0}_n & \mathbf{0}_n \\ \mathbf{0}_n & \mathbf{I}_n & \mathbf{0}_n \end{bmatrix}.$$

- Every combination of (Δ_p, Δ_c) is mapped to a specific dynamic of the system through the matrix $\Phi(\Delta_p, \Delta_c)$

Missing deadlines: effects on control

$$\xi[k+1] = \Phi(\Delta_p, \Delta_c)\, \xi[k]$$

- Every $\Phi(\Delta_p, \Delta_c)$ represents an **operating mode** of the system
  - Different dynamics
  - Constraints on transitions due to (m,k)

- **Constrained switched linear system**

- *Even if some operating modes can be* ***unstable,*** global stability can be still ensured with state of the art analysis

Hypothesis:

- Every combination of mode switches leads to a stable behavior
- **Exponential stability**: bounded by an exponential function

Performance analysis

- Assign a **performance value** for each sequence of N jobs
- Value of N is determined by the exponential bound on the dynamics
- **Sum of quadratic error**

$$P(s) = \sum_{i=0}^{N-1} \xi[i]^T \xi[i]$$
$$= \xi[0]^T \left(\mathbf{I} + \boldsymbol{\Phi}_0^T \boldsymbol{\Phi}_0 + \boldsymbol{\Phi}_0^T \boldsymbol{\Phi}_1^T \boldsymbol{\Phi}_1 \boldsymbol{\Phi}_0 + \ldots + \boldsymbol{\Phi}_0^T \boldsymbol{\Phi}_1^T \cdots \boldsymbol{\Phi}_{N-1}^T \boldsymbol{\Phi}_{N-1} \cdots \boldsymbol{\Phi}_1 \boldsymbol{\Phi}_0 \right) \xi[0]$$
$$= \xi[0]^T \boldsymbol{\Psi}(s) \xi[0]$$

- Matrix elements of $\Psi(s)$ depends on the **ordered** sequence of H/M

Performance analysis

- $\boldsymbol{P}(\boldsymbol{s}) = \xi[\mathbf{0}]^{\boldsymbol{T}}\Psi(\boldsymbol{s})\xi[\mathbf{0}]$

- **Scalar performance index** independent from initial state

$$\prod(s) = ||\Psi(s)||_2$$

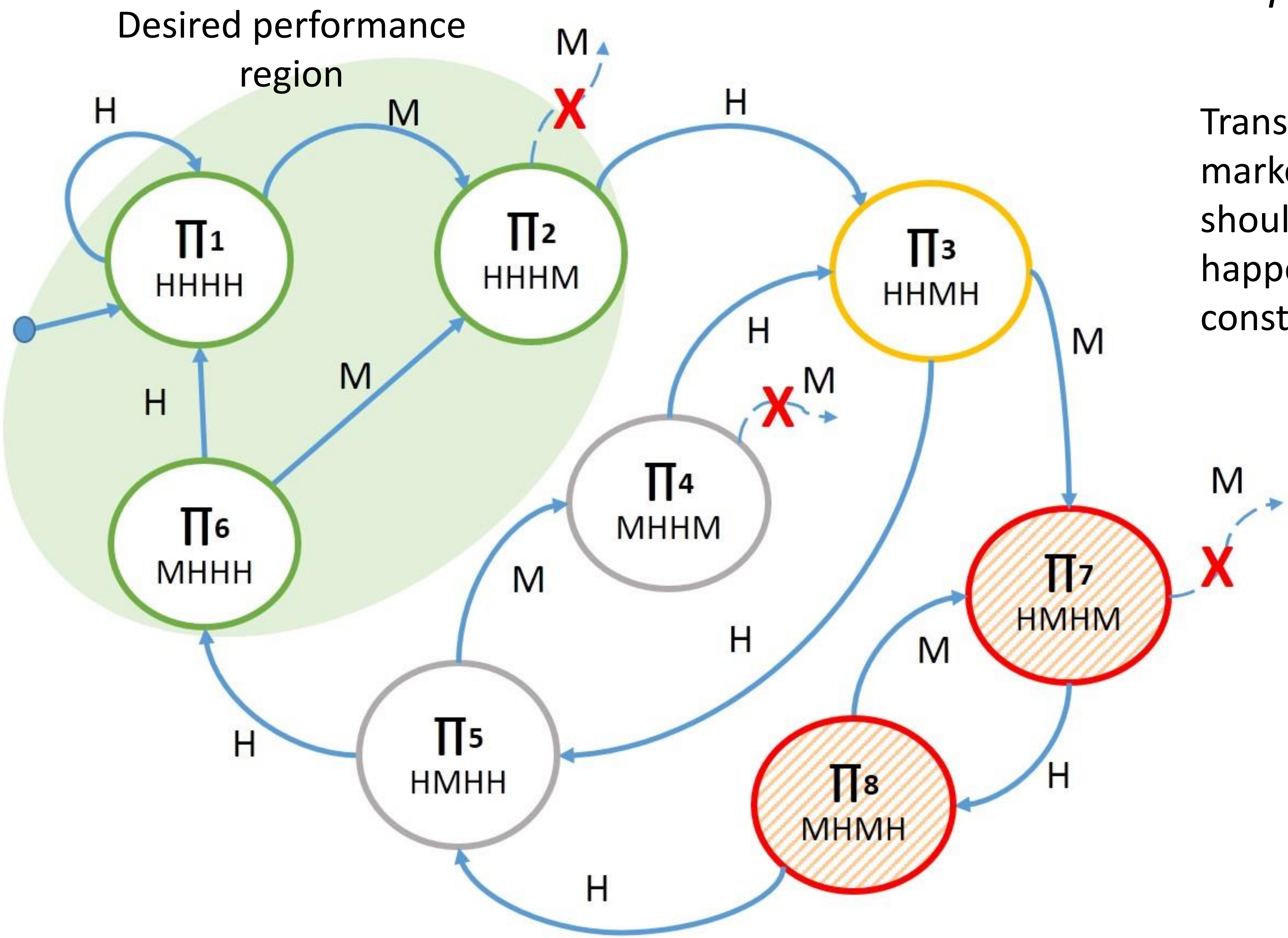
- It is possible to extract one single value representing the worst value for each (m,k) constraint:

- **Worst Case Normalized Performance:** $WCPn = \dfrac{max_s \prod(s)}{\prod(all\ hits)}$

Performance state machine

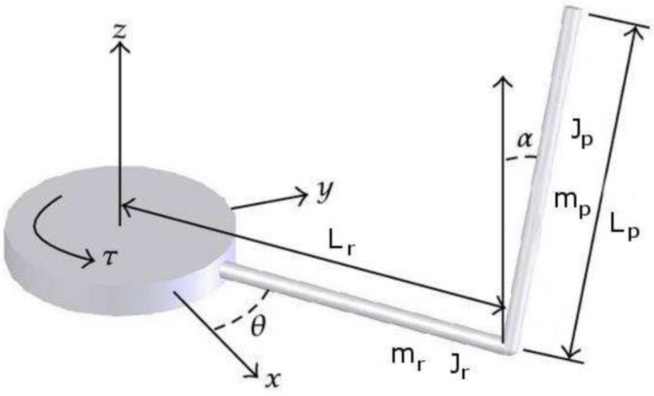
*WH constraint **(1,2)***
N = 4 steps

Transitions marked with **X** should never happen for (m,k) constraints



Case study: Furuta pendulum

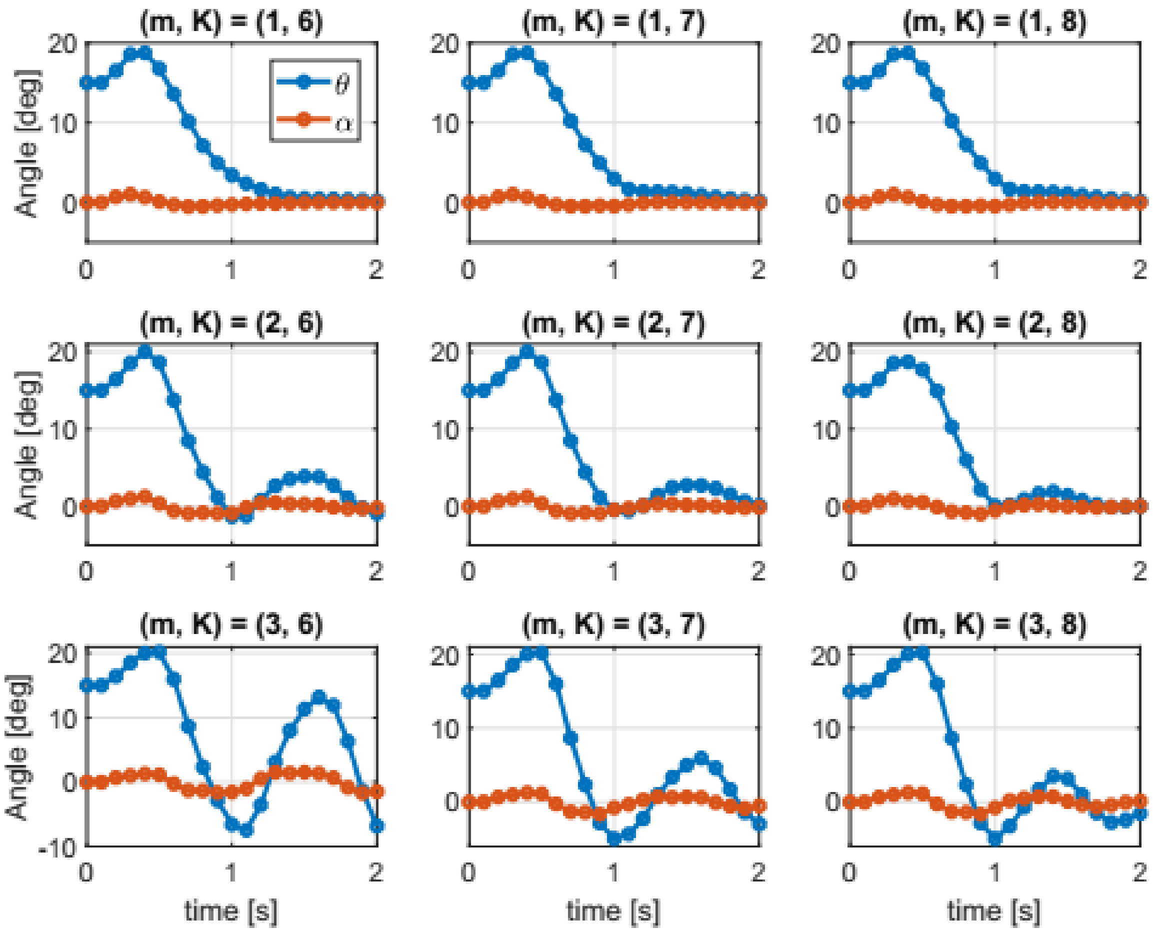
- **Furuta pendulum**: rotary inverted pendulum

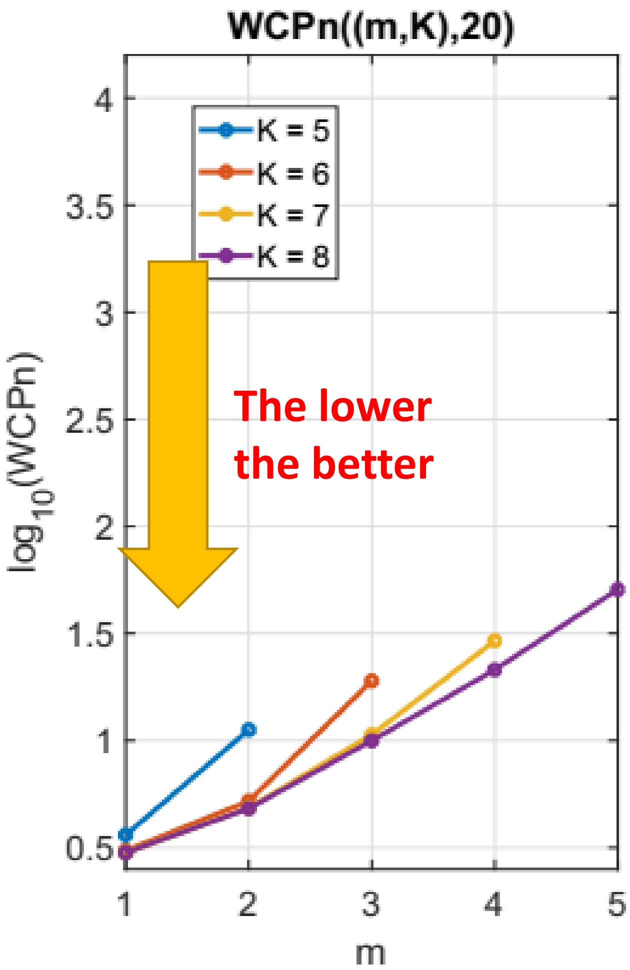


- **Linearized** model in the neighbourhood of the upward position
- Feedback control with $T_i = 0.1sec$ and $D_i = 0.2 * T_i$

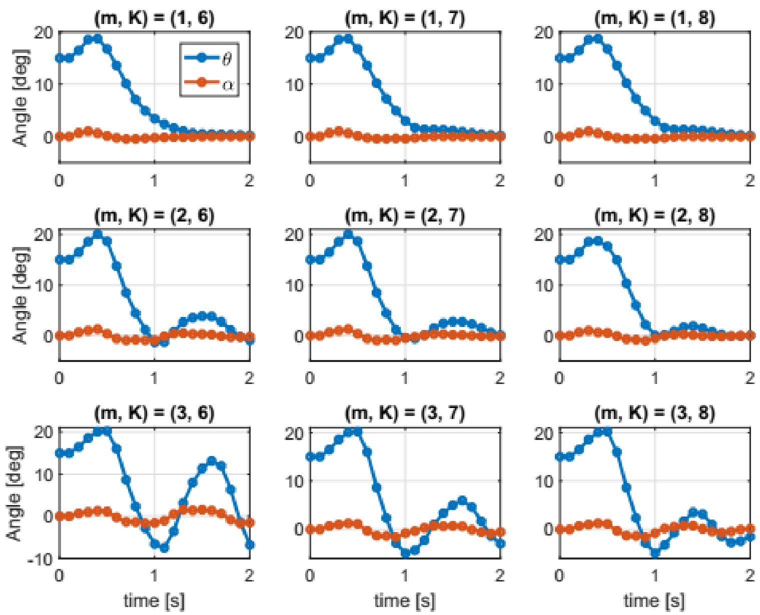
- Testing different (m,K) values and studying how Worst Case performance changes

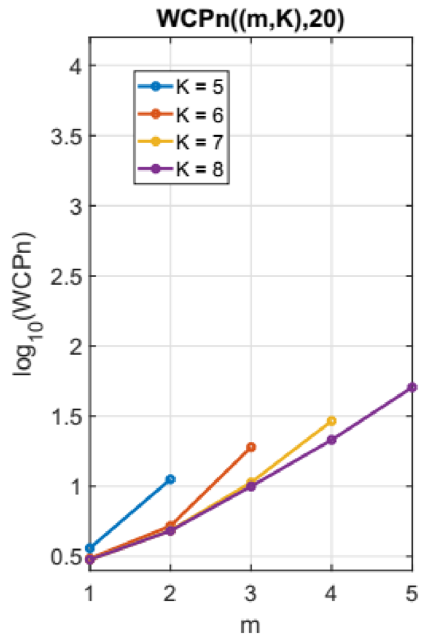
Case study: Furuta pendulum





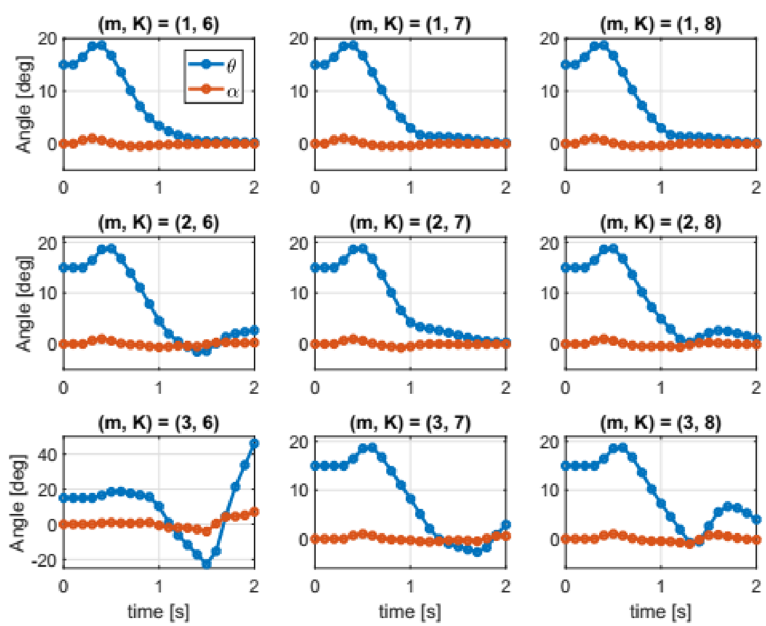
Case study: Furuta pendulum

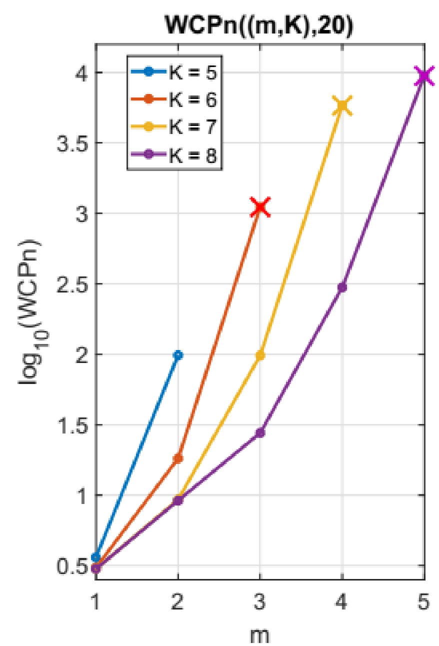




Continue job strategy

Kill job strategy

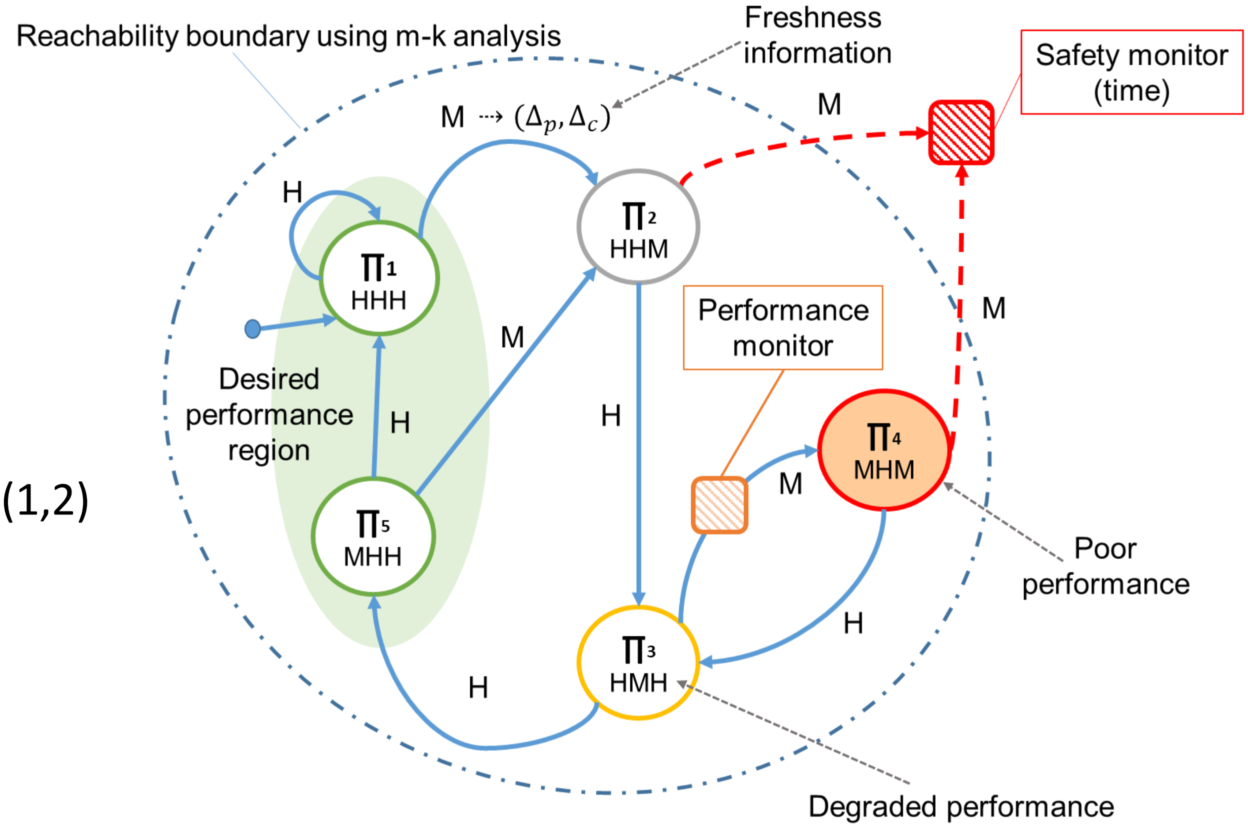




Possible applications

- This new model can be used as a **time contract** between software designers and control engineers
- Possibility of inserting **run-time monitors**

(m,k) = (1,2)



Summary

- New model for studying performance evolution under overload conditions

1. Creating a state machine for computing **freshness** of outputs, applicable to different patterns and handling of deadline misses
2. Intergrating freshness information with state evolution of the controlled system: different **operating modes**
3. Creating a state machine for computing **performance** values realted to patterns of H/M deadlines
   - Worst case performance guarantees
   - Runtime monitors for performance evolution

- **Case study**: Furuta pendulum

Future work

- Extensions:
  - Including additional performance metrics
  - Extending the case study to WCRT>T+D, allowing multiple pending jobs at deadline

- Finding **optimal controller** for a system under (m,K) constraints, for achieveing a given performance

- More complex case studies:
  - Testing non linear systems performance by simulation
  - More complex deadline miss handlings

Any questions?

Thank you!

paolo.pazzaglia@santannapisa.it