A Generic Coq Proof of Typical Worst-Case Analysis

Pascal Fradet, **Maxime Lesourd**,
Jean-François Monin, Sophie Quinton

RTSS'18
December 13, 2018

Context: Weakly Hard Real Time Systems

Hard Real Time Systems

Schedulability: All activations of a task meet their deadline.

Context: Weakly Hard Real Time Systems

Hard Real Time Systems

Schedulability: All activations of a task meet their deadline.

What if we don't need *all* jobs to meet their deadline?

Context: Weakly Hard Real Time Systems

Hard Real Time Systems

Schedulability: All activations of a task meet their deadline.

What if we don't need *all* jobs to meet their deadline?

Weakly Hard Real Time Systems

(m, k) *guarantees*: Out of k consecutive activations of a task, at most m miss their deadline.

Context: Typical Worst Case Analysis

Typical Worst Case Analysis (TWCA) is a family of analyses providing (m, k) guarantees.

Context: Typical Worst Case Analysis

Typical Worst Case Analysis (TWCA) is a family of analyses providing (m, k) guarantees.

Variants:

- Fixed Priority (Non) Preemptive Scheduling
- Earliest Deadline First (EDF)
- Weighted Round Robin
- Extensions for task chains, multiprocessors, ...

Problem Statement and Contribution

Problem: Extending/improving an analysis is tricky & tedious

- Risk of introducing mistakes when reusing an analysis on a "similar" model
- Proofs must be redone from scratch
- Complex analyses often require complex proofs

Problem Statement and Contribution

Problem: Extending/improving an analysis is tricky & tedious

- Risk of introducing mistakes when reusing an analysis on a "similar" model
- Proofs must be redone from scratch
- Complex analyses often require complex proofs

Contribution:

- A *formal proof* of a *generic methodology* to build TWCA based on a simple set of requirements.

Typical Worst Case Analysis

Generic Typical Worst Case Analysis

Generic Typical Worst Case Analysis in Coq

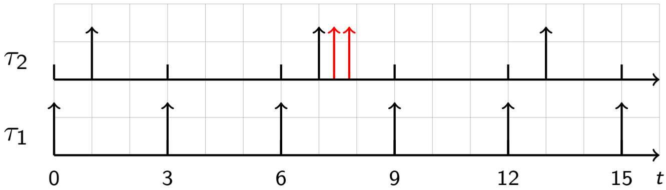
Table of Contents

System Model (FPP)

- Uniprocessor
- Task arrival modeled by arrival curves
  - Maximal number of activations for a given duration
  - Maximal separation between k consecutive activations
- Fixed Priority Preemptive (FPP) scheduling

System Model (FPP)

- Uniprocessor
- Task arrival modeled by arrival curves
  - Maximal number of activations for a given duration
  - Maximal separation between k consecutive activations
- Fixed Priority Preemptive (FPP) scheduling
- Typical and overload components



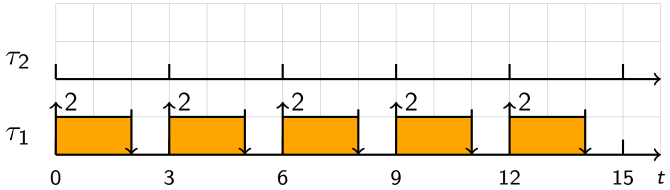
Principle of Typical Worst Case Analysis

Objective:

Given a task τ and k, compute an (m, k) guarantee for τ.

Assumption:

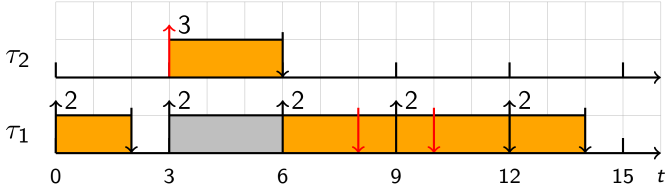
In the absence of overload the system is schedulable.



Principle of Typical Worst Case Analysis

Properties:

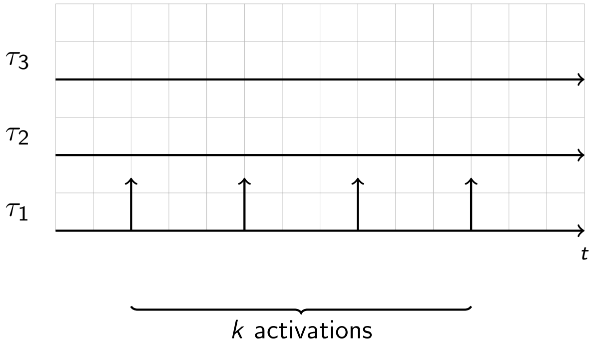
- Overload activations can only affect their busy window
- The size of busy windows can be bounded
- Knowing which overload tasks are activated in a busy window, we can bound the number of deadline misses in that window.



Principle of Typical Worst Case Analysis

Analysis:

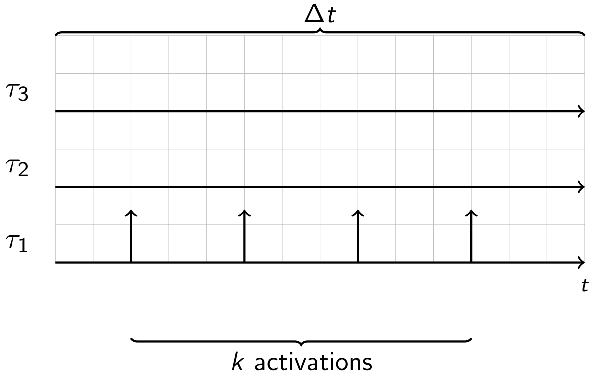
1. Compute how many overload activations can impact a sequence of k consecutive activations of τ.
2. Analyze possible packings of overload activations into busy windows spanning these activations by a reduction to ILP.



Principle of Typical Worst Case Analysis

Analysis:

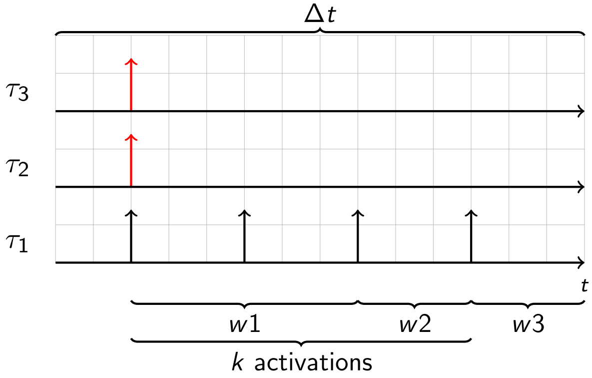
1. Compute how many overload activations can impact a sequence of k consecutive activations of τ.
2. Analyze possible packings of overload activations into busy windows spanning these activations by a reduction to ILP.



Principle of Typical Worst Case Analysis

Analysis:

1. Compute how many overload activations can impact a sequence of k consecutive activations of τ.
2. Analyze possible packings of overload activations into busy windows spanning these activations by a reduction to ILP.

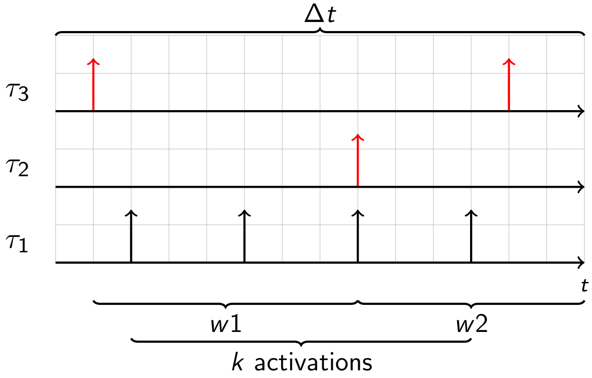


	w_1	w_2	w_3
τ_3	$\top$	$\bot$	$\bot$
τ_2	$\top$	$\bot$	$\bot$
N_1	1	0	0

Principle of Typical Worst Case Analysis

Analysis:

1. Compute how many overload activations can impact a sequence of k consecutive activations of τ.
2. Analyze possible packings of overload activations into busy windows spanning these activations by a reduction to ILP.



	w_1	w_2
τ_3	$\top$	$\top$
τ_2	$\bot$	$\top$
N_1	1	1

Table of Contents

How Generic?

- Generic *w.r.t.* the scheduling policy
- Generic *w.r.t.* the activation model
- Generic *w.r.t.* the type of combinations
- Generic *w.r.t.* a notion of schedulability analysis

How Generic?

- Generic *w.r.t.* the scheduling policy
  We need some notion of busy window
- Generic *w.r.t.* the activation model
  Today we focus on arrival curves
- Generic *w.r.t.* the type of combinations
  Finer grained abstraction
- Generic *w.r.t.* a notion of schedulability analysis
  It only needs to be "correct"

A Recipe for TWCA

Objective:

Given a task τ and k, compute an (m, k) guarantee for τ.

A Recipe for TWCA

Objective:

Given a task τ and k, compute an (m, k) guarantee for τ.

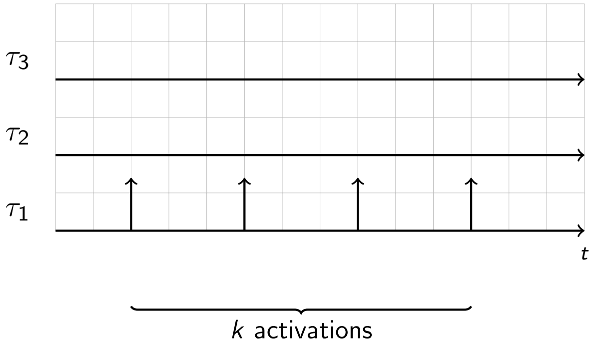
Ingredients:

- A way to decompose traces into isolated intervals: *Analyzable windows*
- An abstraction of the activations inside analyzable windows: *Combinations*
- A local deadline miss analysis based on combinations: *Local analysis*
- A bound on the size of analyzable windows of task τ
- An upper bound on the separation of activations of task τ

Principle of GTWCA

Analysis:

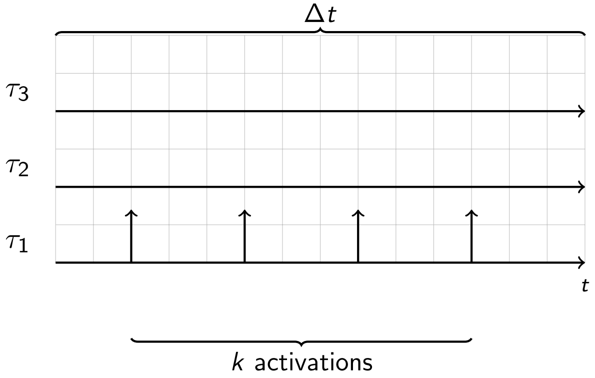
1. Compute how many overload activations can impact a sequence of k consecutive activations of τ.
2. Analyze possible packings of activations into analyzable windows spanning these activations by a reduction to ILP.



Principle of GTWCA

Analysis:

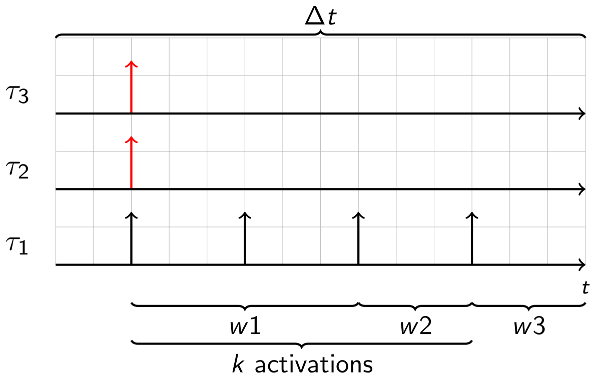
1. Compute how many overload activations can impact a sequence of k consecutive activations of τ.
2. Analyze possible packings of activations into analyzable windows spanning these activations by a reduction to ILP.



Principle of GTWCA

Analysis:

1. Compute how many overload activations can impact a sequence of k consecutive activations of τ.
2. Analyze possible packings of activations into analyzable windows spanning these activations by a reduction to ILP.

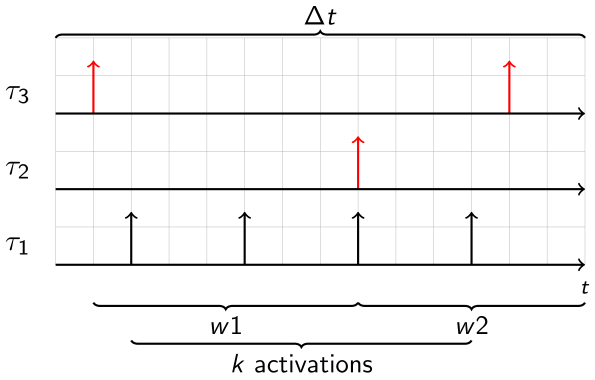


	w_1	w_2	w_3
τ_3	1	0	0
τ_2	1	0	0
N_1	1	0	0

Principle of GTWCA

Analysis:

1. Compute how many overload activations can impact a sequence of k consecutive activations of τ.
2. Analyze possible packings of activations into analyzable windows spanning these activations by a reduction to ILP.



	w_1	w_2
τ_3	1	1
τ_2	0	1
N_1	1	1

Instantiations of GTWCA

Instantiation to several scheduling Policies:

- FPP
- FPNP
- EDF

Instantiations of GTWCA

Instantiation to several scheduling Policies:

- FPP
- FPNP
- EDF

Bonus: derivation of the existing FPP analysis

Principle: work on combinations

- Remove higher priority tasks
- Distinguish overload tasks
- Use boolean combinations

Table of Contents

Real Time Systems Analysis in Coq

Coq

- Formal language
- Interactive proof system
- Increasingly used in industry

Real Time Systems Analysis in Coq

Coq

- Formal language
- Interactive proof system
- Increasingly used in industry

Prosa

- Proofs of schedulability analyses
- Readable specifications of real-time systems
- Written in Coq

Artifact

We have reused from Prosa:

- Fundamental definitions
- System model
- Schedulability analysis for FPP

Artifact

We have reused from Prosa:

- Fundamental definitions
- System model
- Schedulability analysis for FPP

We have formalized in Coq:

- The recipe for GTWCA
- A proof of the reduction to ILP
- An instantiation to the arrival curves model
- An instantiation to the FPP scheduling policy

Lessons Learned

The effort to formalize in Coq is not negligible...

- TWCA for FPP: 2 months
- GTWCA: 3 weeks
- Instantiation to FPP: 2 weeks

Lessons Learned

The effort to formalize in Coq is not negligible...

- TWCA for FPP: 2 months
- GTWCA: 3 weeks
- Instantiation to FPP: 2 weeks

...but:

- Being explicit about hypotheses helps generalizing safely
- Generic proofs allow us to reuse model specific results

Conclusion & Future Work

What we have:

- A generic framework for TWCA
- A simple methodology to instantiate the analysis
- A formalization and proof in Coq of our results

Conclusion & Future Work

What we have:

- A generic framework for TWCA
- A simple methodology to instantiate the analysis
- A formalization and proof in Coq of our results

Future work:

- Extend GTWCA to more complex system models
- Investigate other types of combinations