Technische Universität Braunschweig

INSTITUTE OF COMPUTER AND NETWORK ENGINEERING

Response-Time Analysis for Task Chains with Complex Precedence and Blocking Relations

Johannes Schlatow, Rolf Ernst

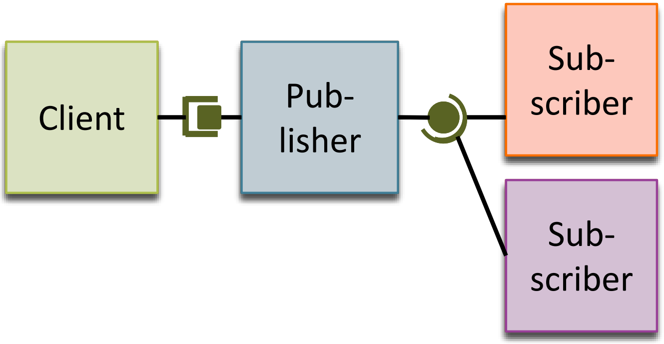
October 17, 2017

Introduction

From pure control algorithms (timing-centric) to ADAS (communication-centric).

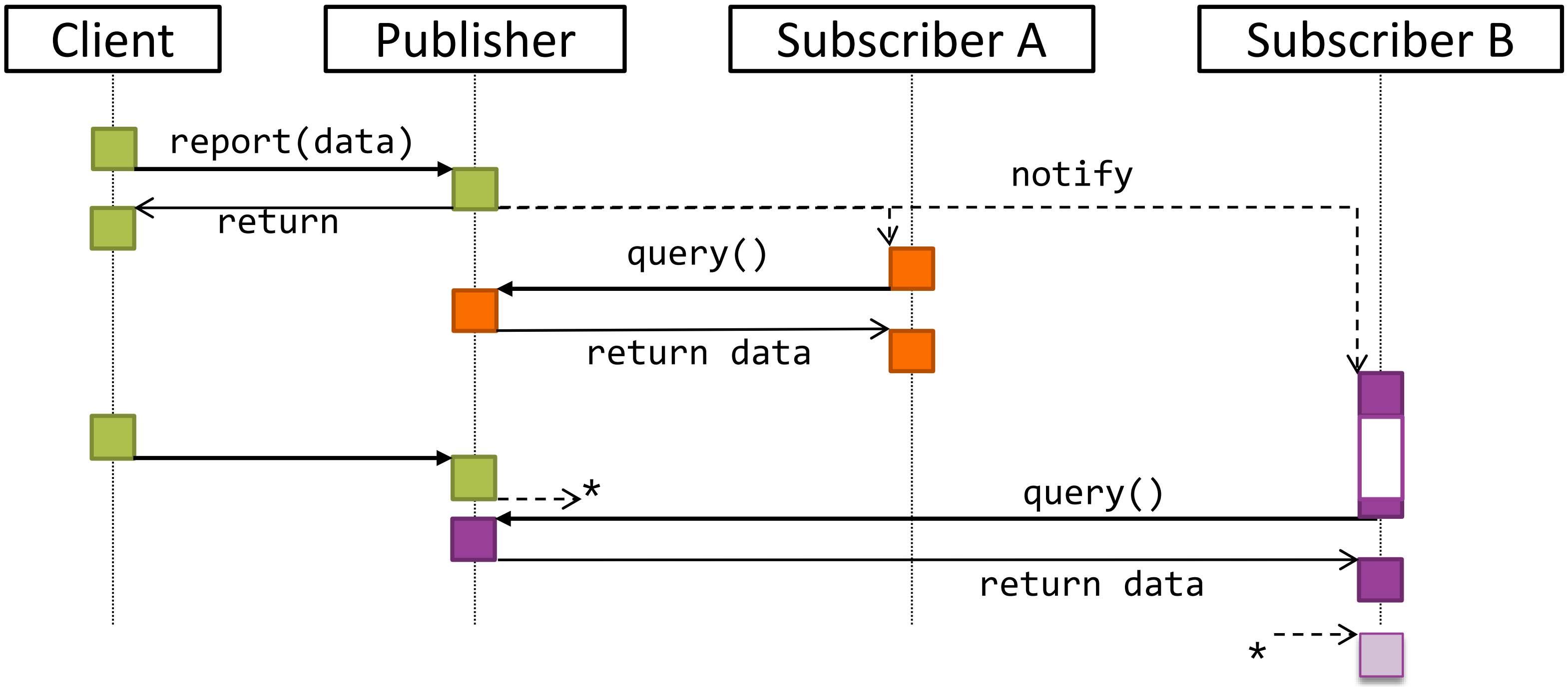
© *automotiveIT.com*

- object-oriented and **component-based** design for reusability and separation
- in particular, **microkernel** architectures (e.g. QNX Neutrino in automotive domain)
- focus on interaction of software components (**service-oriented architectures**)

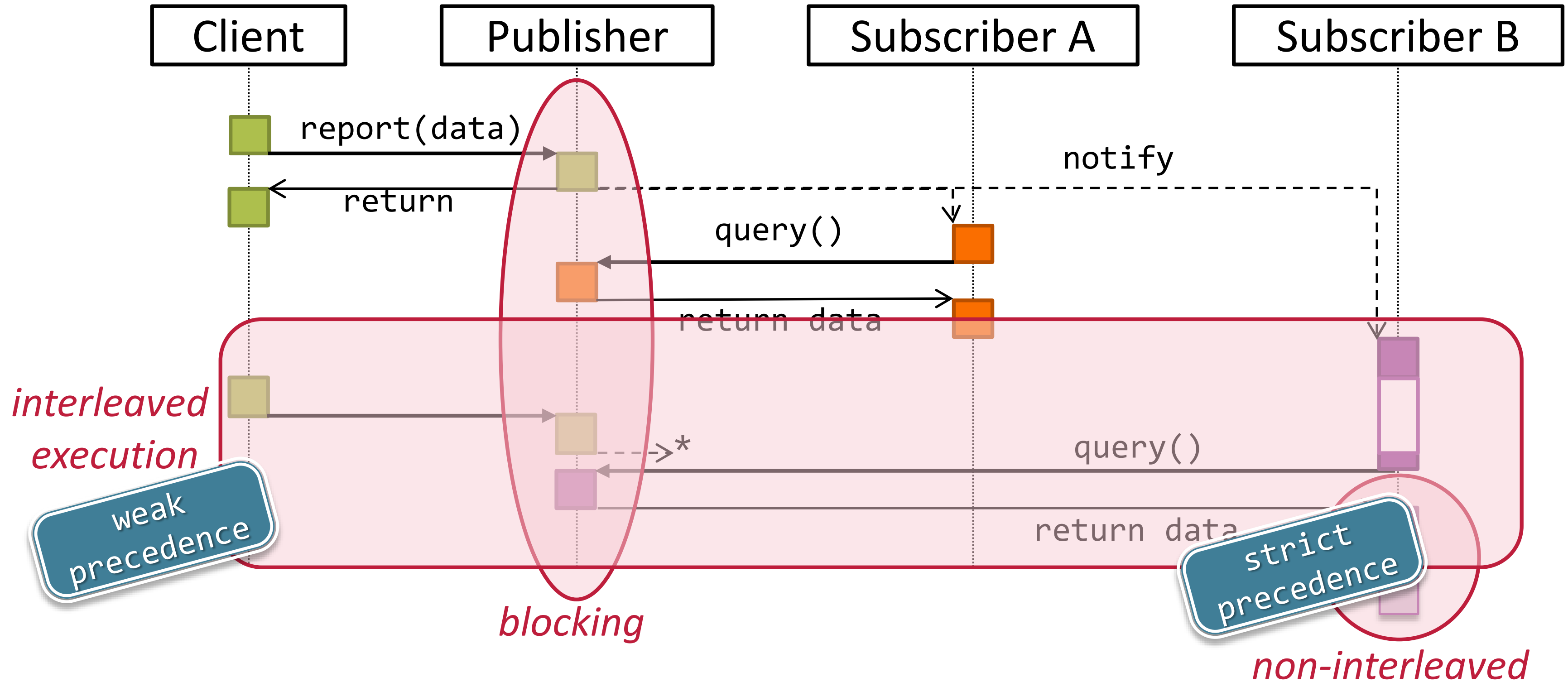


- precedence relations → **task chains**
- shared services → **blocking**
- here: software component = **thread**

Interaction and communication described by sequence diagrams:



Interaction and communication described by sequence diagrams:



Contribution

- Modelling and RTA of chains with mixed precedence relations and blocking.

Outline

- Introduction
- Modelling
- Response-time analysis (RTA)
- Related work
- Evaluation
- Conclusion

Modelling precedence and blocking relations

Idea: decouple implementation details from (timing) analysis model

- “standalone” model for single processor serves as **RTA input**
- incorporate knowledge about **OS implementation**
- independent from scheduling policy (**how vs. where** of scheduling decisions)

How do scheduling parameters propagate during communication?

- e.g. priority inheritance, thread migration, time-slice donation

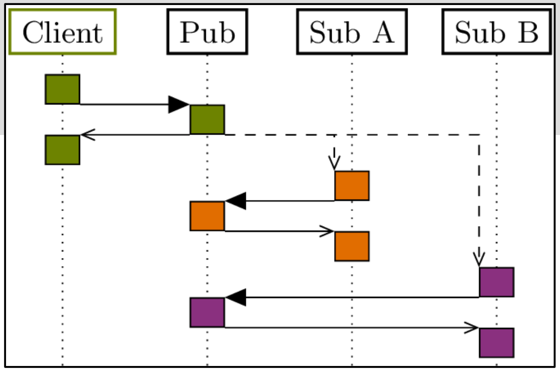
→ mapping of tasks to scheduling contexts (thread as **scheduled entity**)

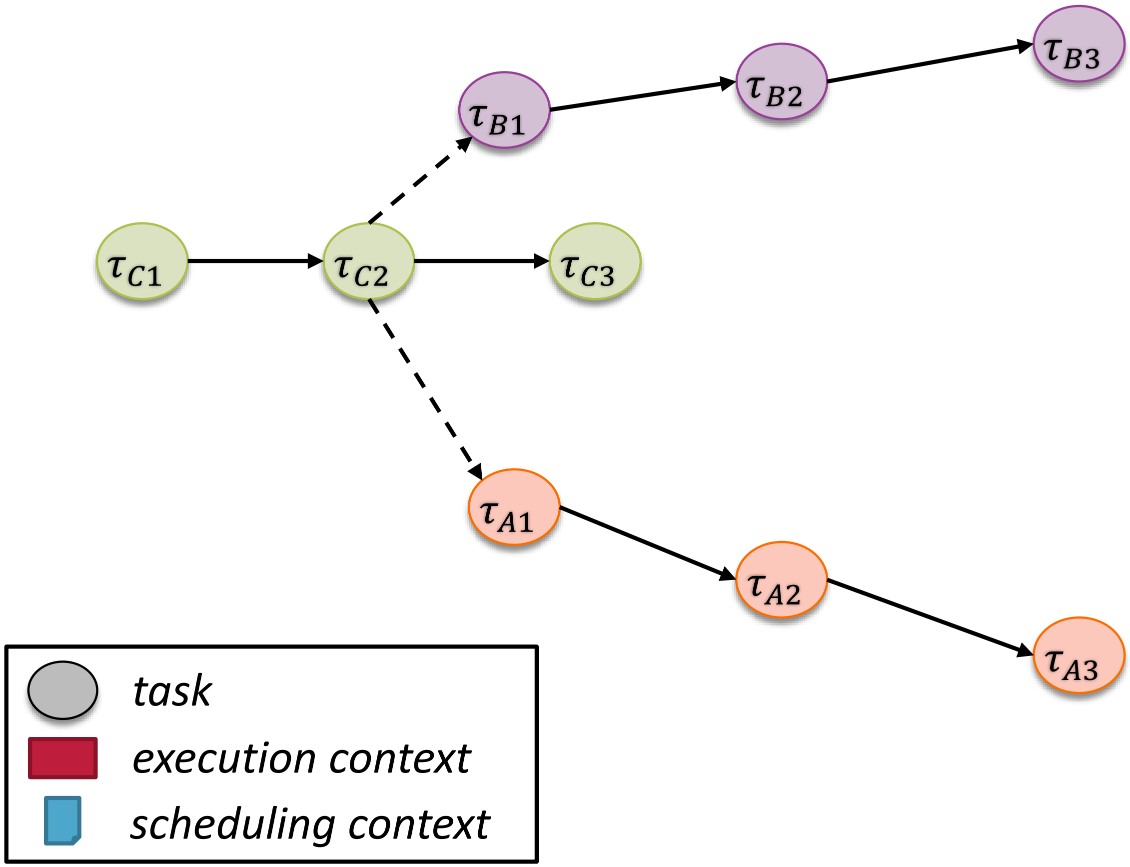
When can components be re-entered?

- wait for returns, ready to receive notifications

→ mapping of tasks to execution contexts (thread as **shared (local) resource**)

Task model

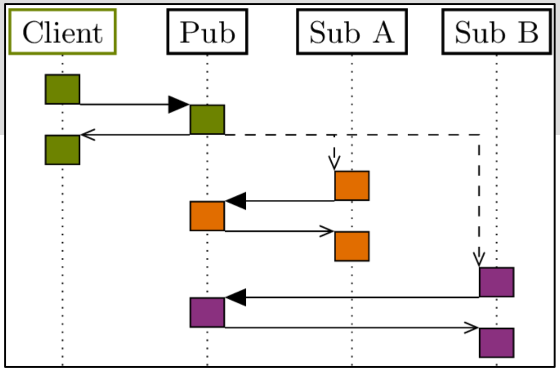


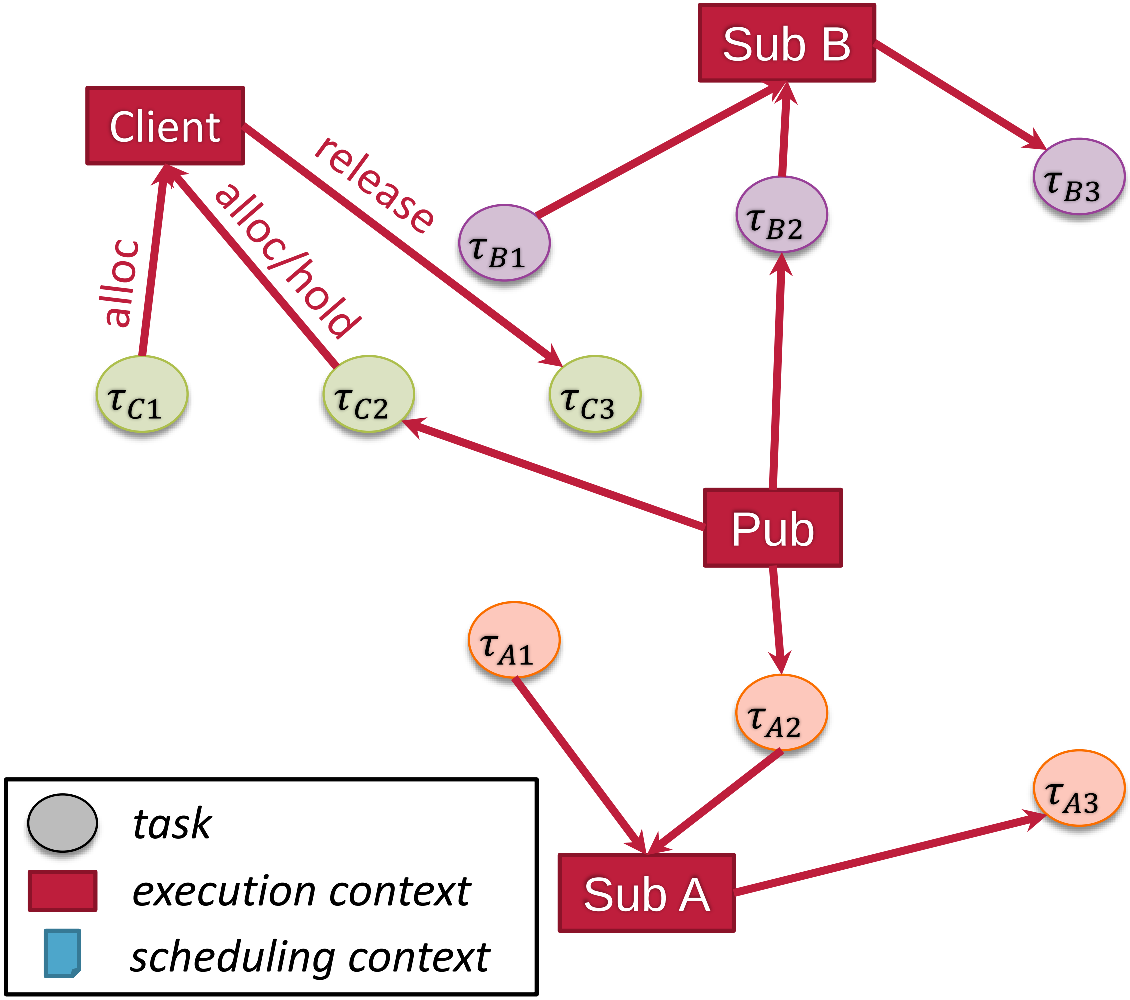


Task graph

- directed, acyclic

Task model





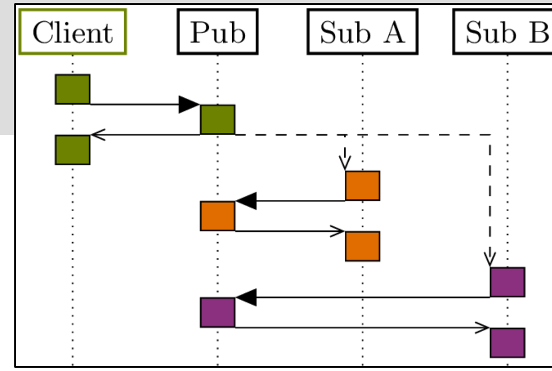
Task graph

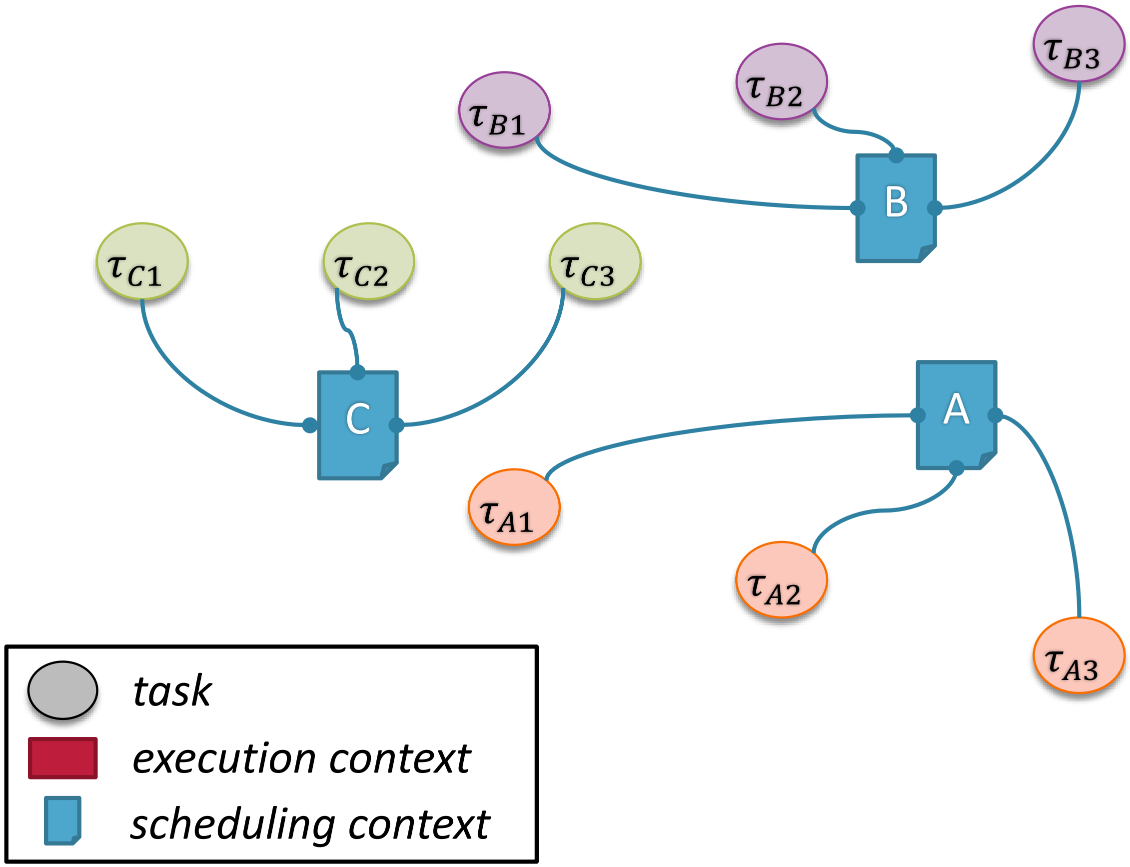
- directed, acyclic

Allocation graph

- bipartite, directed

Task model





Task graph

- directed, acyclic

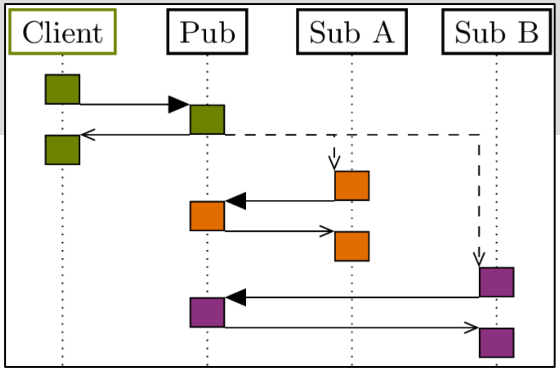
Allocation graph

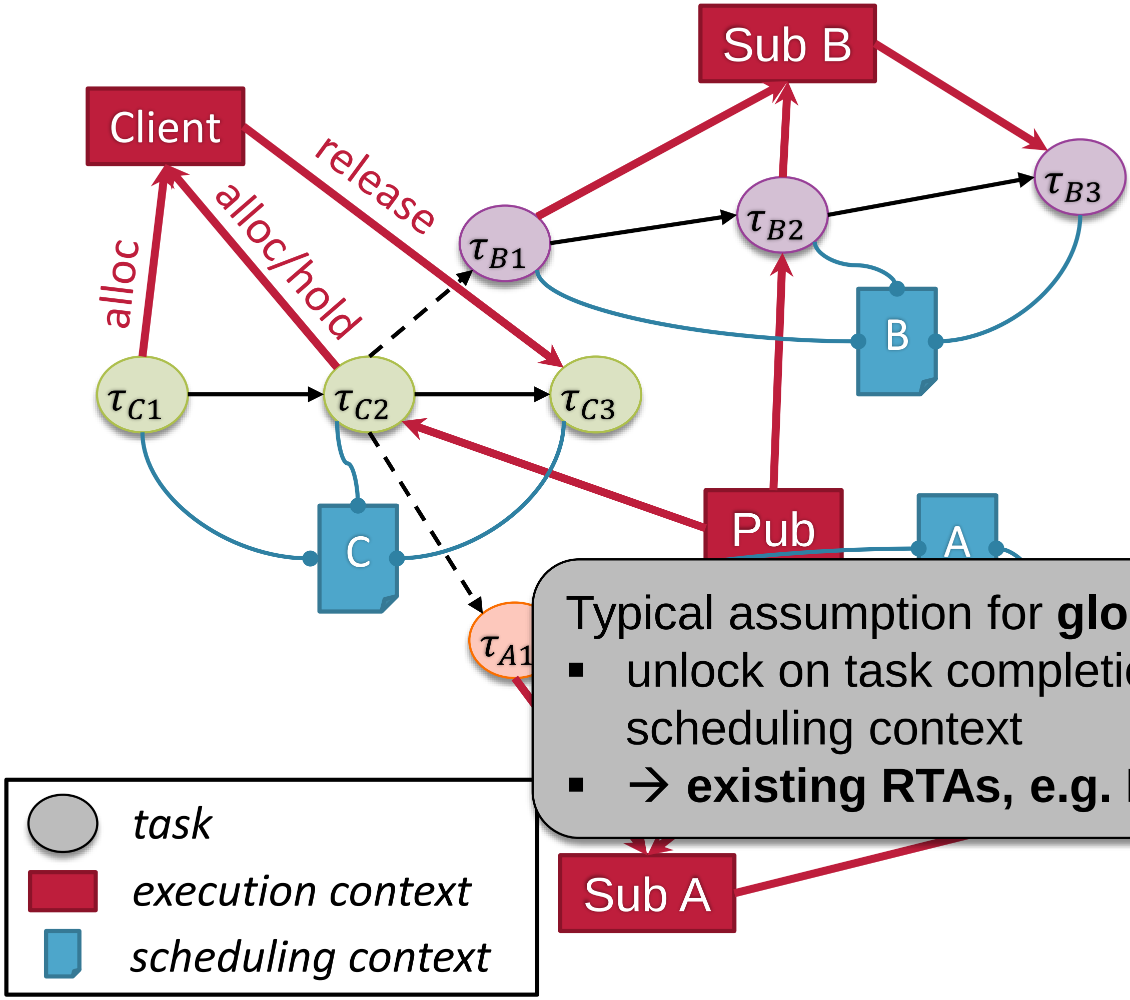
- bipartite, directed

Mapping graph

- bipartite, undirected

Task model





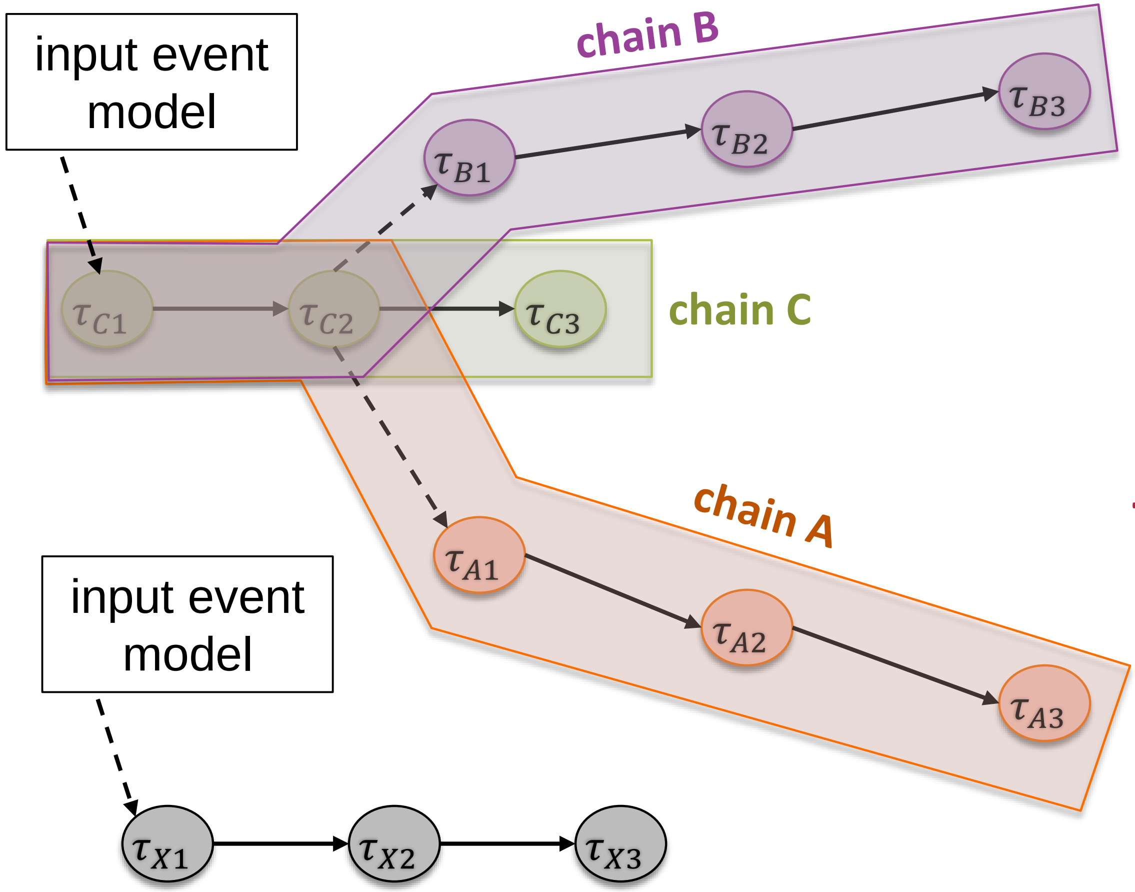
Task graph

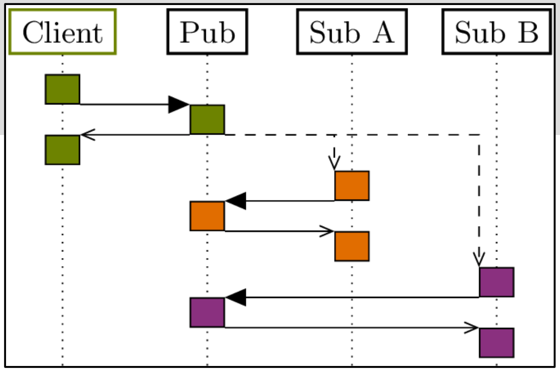
- directed, acyclic

Typical assumption for **global** shared resources:

- unlock on task completion or when leaving scheduling context
- → **existing RTAs, e.g. MAST, not applicable here**

Task chains





- **sequence** of directly connected tasks
- **arbitrarily** defined

for RTA:

- given task model
- every task must belong to **at least one chain**
- known input **event model(s)**
- known **scheduling policy**

Response-time analysis (RTA)

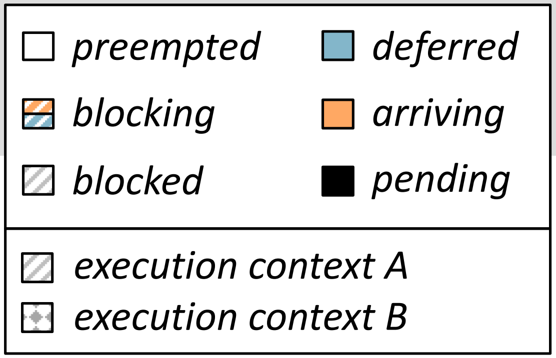
Problem statement

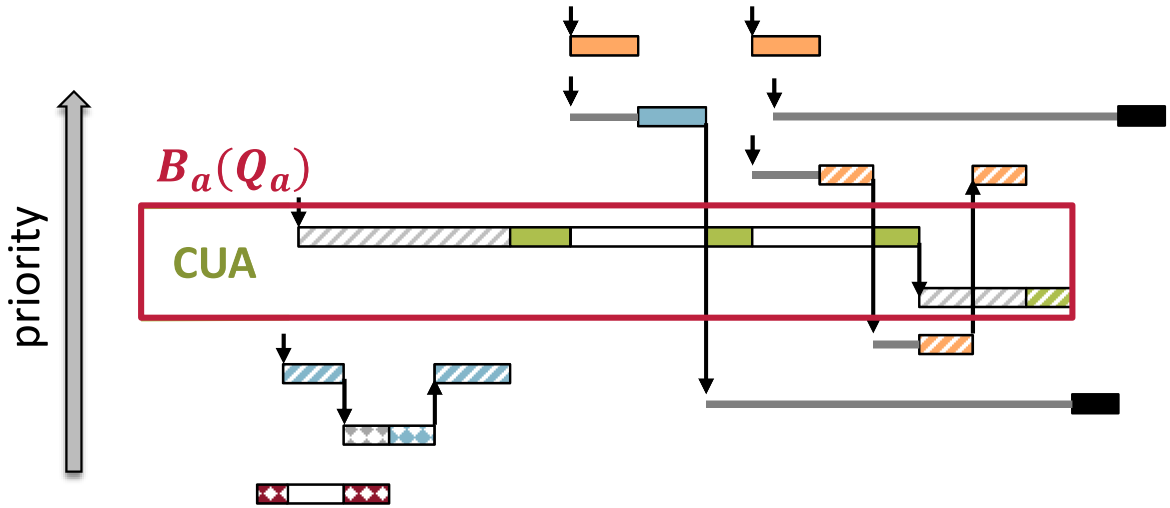
- find worst-case interference scenario for chain under analysis (CUA)
- static-priority preemptive (SPP)
- arbitrary event models: arrival curves $\eta^+(\Delta t)/\eta^-(\Delta t)$

q-event task-chain busy window $B_a(q)$

- "[…] denotes the maximum time a processor may be **busy processing q-events** of the CUA T_a. […]"
  - after **maximum busy window** $B_a(Q_a)$ there are no pending activation of T_a
  - but: other activations can be pending (**deferred load**)

Possible interference scenarios





- ✓ lower-priority blocking
- ✓ transitive blocking
- ✓ higher-priority interference
- ✓ deferred interference
- ✓ priority inversion

- **arriving** interference
  - no pending activations after $B_a(Q_a)$
  - bounded by arrival curves
- **deferred** interference
  - pending activations
  - not dependent on arrival curves

Observation:
interference by a task depends on **how often** its predecessors can execute within B_a

Introducing event-count bounds

q-event busy window for chain T_a :

$$\forall q \in [1, Q_a]: \quad B_a(q) = \sum_{\tau_i} n_{a,i}(q) \cdot C_i^+$$

WCET

with

lower bound

$$n_{a,i}(q) = \max(\zeta_{a,i}(q), \min_k \vartheta_{a,i}^{(k)}(q))$$

k event-count upper bounds

Lower bound (starting point):

$$\zeta_{a,i}(q) = \begin{cases} q & \forall \tau_i \in T_a \\ 0 & else \end{cases}$$

Upper bounds:

- $\vartheta_{a,i}^{(k)}(q)$: k-th upper bound for task τ_i in $B_a(q)$
- → optimsation problem ($\min_k$)

Upper event-count bounds $\boldsymbol{\vartheta}_{\boldsymbol{a,i}}^{(\boldsymbol{k})}(\mathrm{q})$

- each bound focusses on different effects
- i.e. tighter for particular τ_i, conservative for others

Preconditions for $\boldsymbol{\vartheta}_{\boldsymbol{a,i}}^{(\boldsymbol{k})}$:

- must include all interference effects (→ conservative bounds)
  - preemptions from predecessors in CUA ("self interference")
  - transitive blocking
  - priority inversion
- no mutual exclusion → bounds must always hold
- may depend on results from other bounds (fixed-point problem, propagation)

What bounds can we formulate?

Defining event-count bounds

$\min_k \vartheta_{a,i}^{(k)}(q)$

Arrival function ($\forall \tau_i \in T_b$)

$$\vartheta_{a,i}^{(1)}(q) = \eta_b^+\left(B_a(q)\right)$$

Self-interference (for last task of T_a and its strict predecessors)

$$\vartheta_{a,i}^{(2)}(q) = q$$

Deferred interference ($\forall \tau_i$ with lower-priority or strict predecessor)

$$\vartheta_{a,i}^{(3)}(q) = \begin{cases} 1 & if\ n_{a,j}(q) = 0 \\ \infty & else \end{cases}$$

Lower-priority ($\forall \tau_i$ not blocking, not higher priority, $\notin T_a$)

$$\vartheta_{a,i}^{(4)}(q) = \begin{cases} 0 & if\ lowest\ priority \\ 0 & if\ \nexists lower\ priority\ \tau_j : n_{a,j}(q) > 0 \\ \infty & else \end{cases}$$

Related work

[Gonzales Harbour et al. 1994]

- subtask model, no blocking relations, only strict precedence, mutual exclusion

task-chain analyses: [Schlatow2016], [Hammadeh2017]

- no blocking relations
- precedence relations can vary between (not within) chains

MAST/MARTE UML (offset-based analyses)

- similar modelling concepts: scheduling servers, shared resources
- locks must not be hold across scheduling server boundaries

Blocking effects / shared resources:

- transitive blocking [Biondi2016]
- focus on global shared resources
  - typical restrictions: locks are released upon task completion

Evaluation

Caveat: RTA targets new task model → limited comparability

a) client-publisher-subscriber example

- not comparable with other work

→ details in the paper/poster

b) modified case study from [Schlatow2016]

- compare with MAST

c) synthetic benchmarks

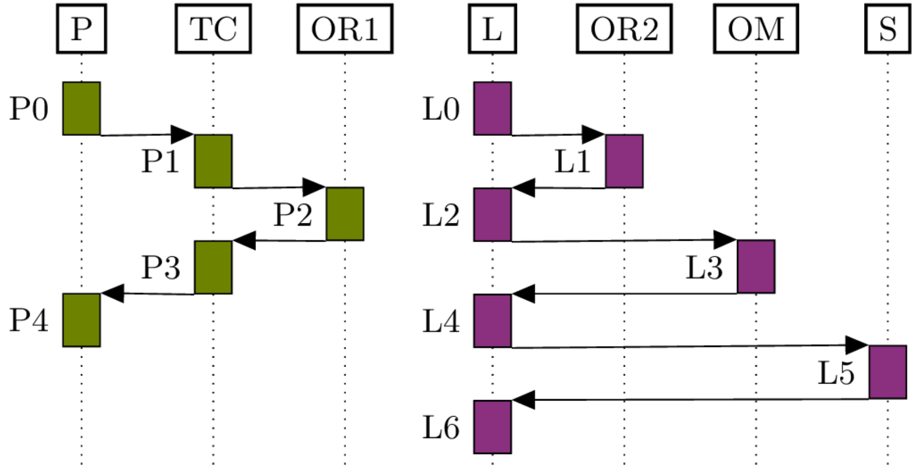
- test analyzability and scalability

→ details in the paper/poster

Evaluation of ADAS use case from [Schlatow2016]

Setup

- park and lane assist chain
- original setup:
  - 7 scheduling contexts
  - no blocking



Results

- requires **additional candidate search** to achieve same results (mutual exclusion):
  - $\max(\min_k \vartheta^{(k)}, \min_l \vartheta^{(l)}, \dots)$

Next step: modify to include blocking

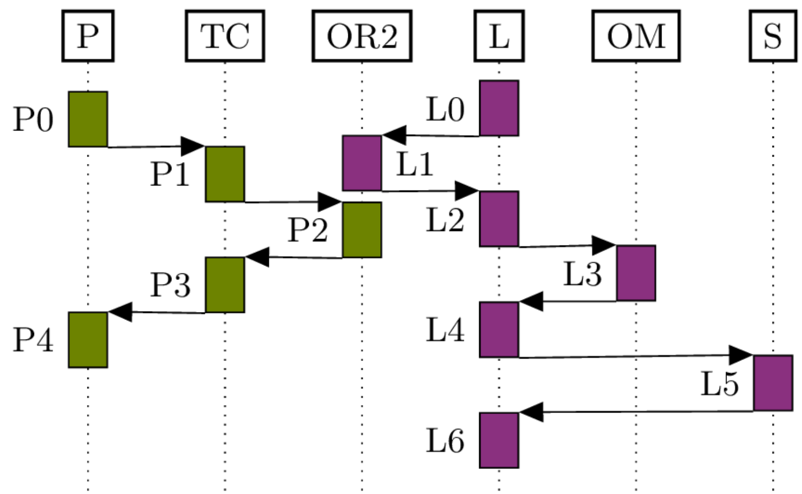
Evaluation of modified ADAS use case

Modified setup:

- one shared execution context
- priority inheritance

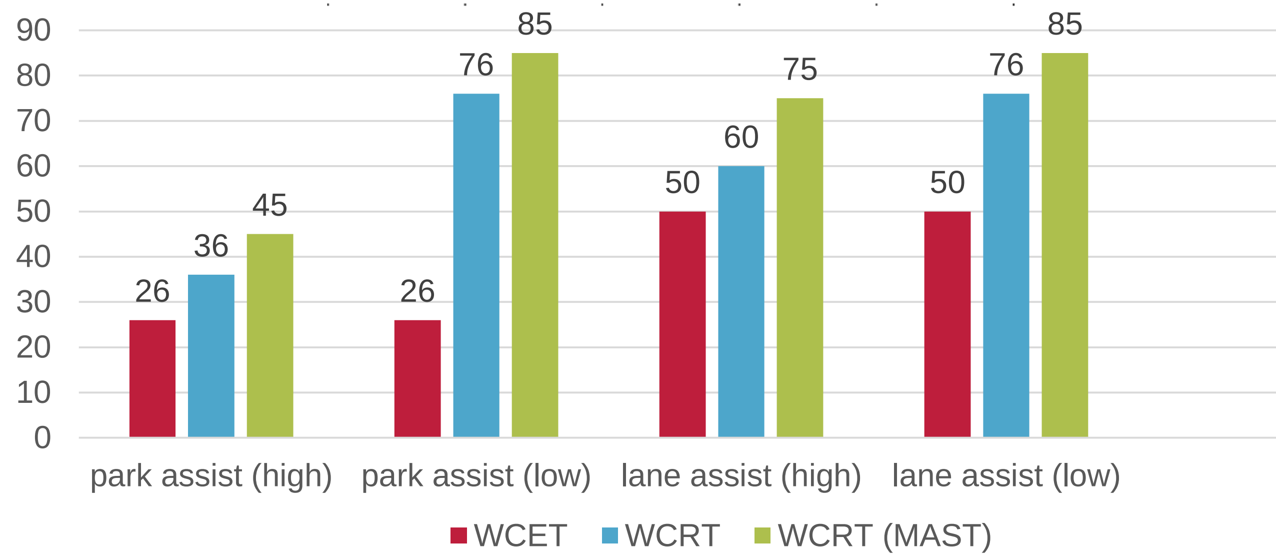
→two scheduling contexts

→comparable with MAST



Results:

- high-priority chain blocked by L1/P2
- low-priority chain = sum of all WCETs
- pessimistic results from MAST



Conclusion

- comprehensive **timing model** for inter-component communication
- RTA of scenarios **not possible before**
- covering **priority inversion**, **transitive blocking** and **deferred activations** in single framework by conservative bounds (no restrictions)
- **tight results** when combined with candidate search
- outperforms **(py)CPA** and **MAST** (where comparable)
- **scalability** (convergence of analysis) up to 99% load (see paper)

Thank you for your attention.

In case of **questions**, please ask **now** or at the **poster**.

Code available at https://bitbucket.org/pycpa/pycpa_taskchain

References

- [Maki-Turja et al. 2008] Jukka Mäki-Turja and Mikael Nolin, “Efficient implementation of tight response-times for tasks with offsets.” Real-Time Systems 40, 2008.
- [Palencia et al. 1999] J. C. Palencia and M. G. Harbour, “Exploiting precedence relations in the schedulability analysis of distributed real-time systems,” in Real-Time Systems Symposium, 1999.
- [Schlatow et al. 2016] Johannes Schlatow and Rolf Ernst, “Response-Time Analysis for Task Chains in Communicating Threads.” Real-Time Embedded Technology and Applications Symposium (RTAS), 2016.
- [Hammadeh et al. 2017] Zain A. H. Hammadeh, Sophie Quinton, Rafik Henia, Laurent Rioux und Rolf Ernst, "Bounding Deadline Misses in Weakly-Hard Real-Time Systems with Task Dependencies“, Design Automation and Test in Europe (DATE), 2017.
- [Biondi et al.] Alessandro Biondi, Björn B Brandenburg, and Alexander Wieder, “A Block-ng Bound for Nested FIFO Spin Locks.” Real-Time Systems Symposium (RTSS), 2016.
- [Gonzales Harbour et al. 1994] M. Gonzalez Harbour, M. H. Klein, J.P. Lehoczky, “Timing analysis for fixed-priority scheduling of hard real-time systems”, IEEE Transactions on Software Engineering, 1994.