A ROS 2 Response-Time Analysis Exploiting Starvation Freedom and Execution-Time Variance

T. Blass, D. Casini, S. Bozhko, B. Brandenburg

SIC Saarland Informatics Campus

Sant'Anna Scuola Universitaria Superiore Pisa

MAX PLANCK INSTITUTE FOR SOFTWARE SYSTEMS

BOSCH Invented for life

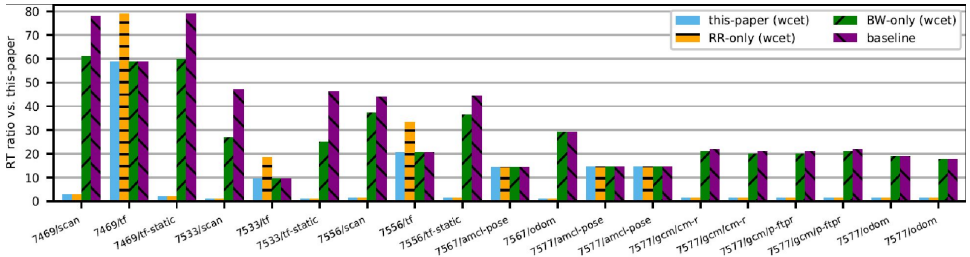
This Paper in a Nutshell

ROS 2 is one of the most popular robotics frameworks, with peculiar timing properties.

We **improve** upon existing response-time analyses with three techniques.

1. Address large **execution-time variance** over time
2. Exploit **starvation-freedom** in the callback scheduler
3. Improve activation-curve **propagation within executors**

Experiments show significant improvements (10-80x) in **real-world ROS packages**.



This Paper in a Nutshell

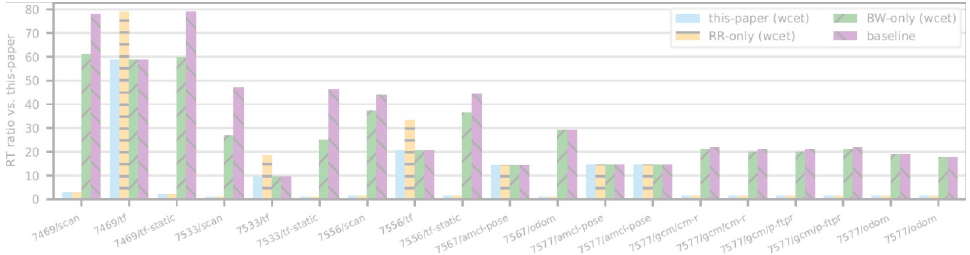
ROS 2 is one of the most popular robotics frameworks, with peculiar timing properties.

ROS

We **improve** upon existing response-time analyses with three techniques.

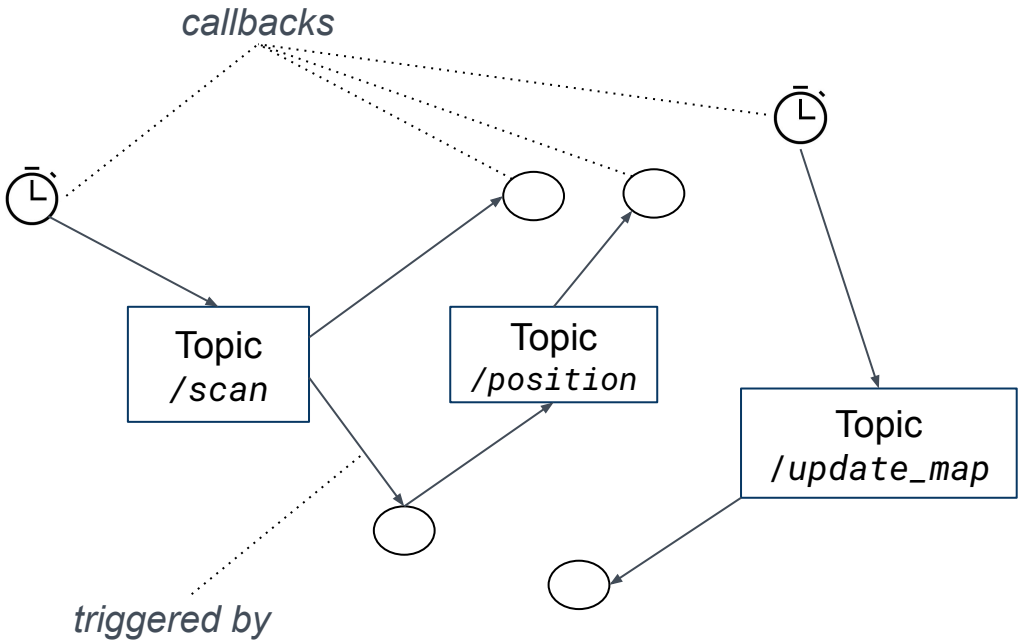
1. Address large **execution-time variance** over time
2. Exploit **starvation-freedom** in the callback scheduler
3. Improve activation-curve **propagation within executors**

Experiments show significant improvements (10-80x) in **real-world ROS packages**.

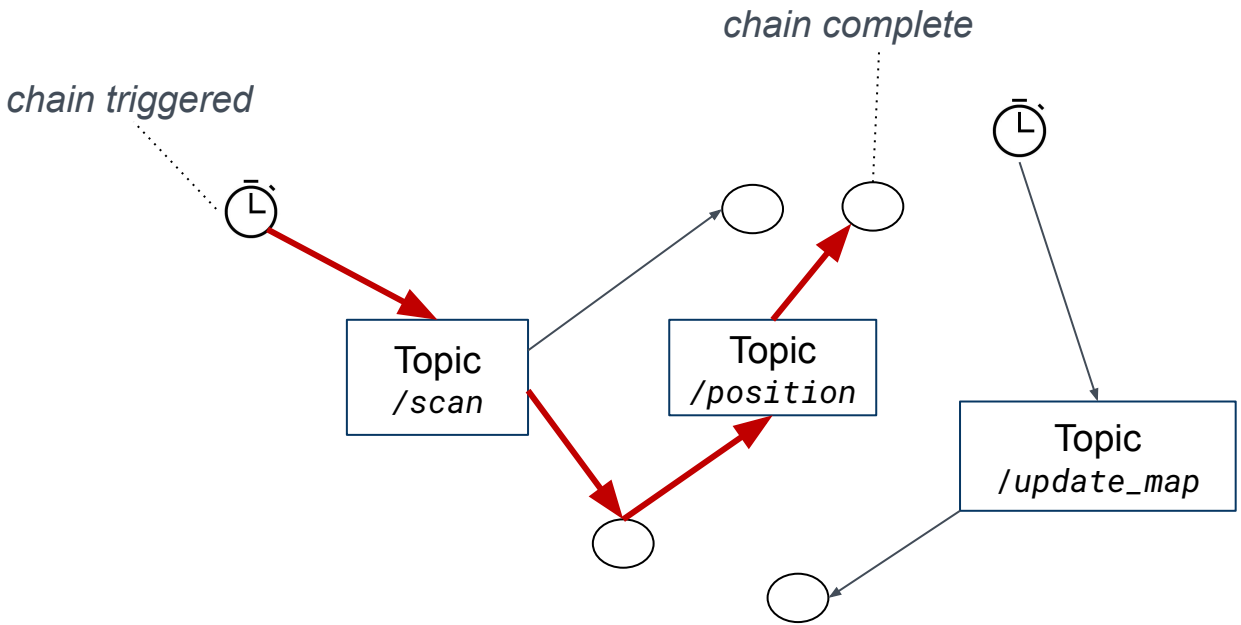


BOSCH

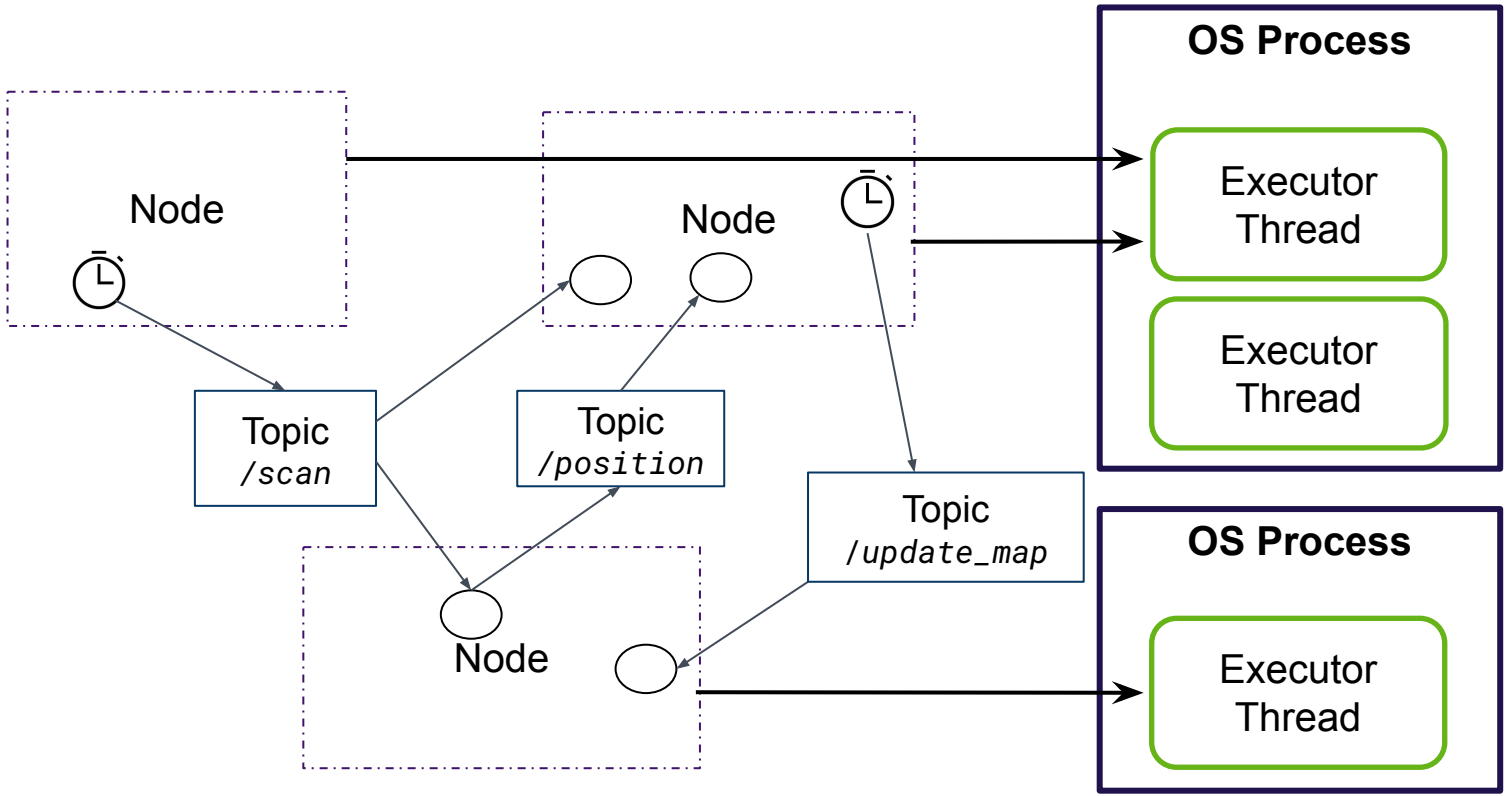
Background: ROS Callback Graph



Background: Processing Chains



Background: Executing a Callback Graph



Background: Compositional Performance Analysis

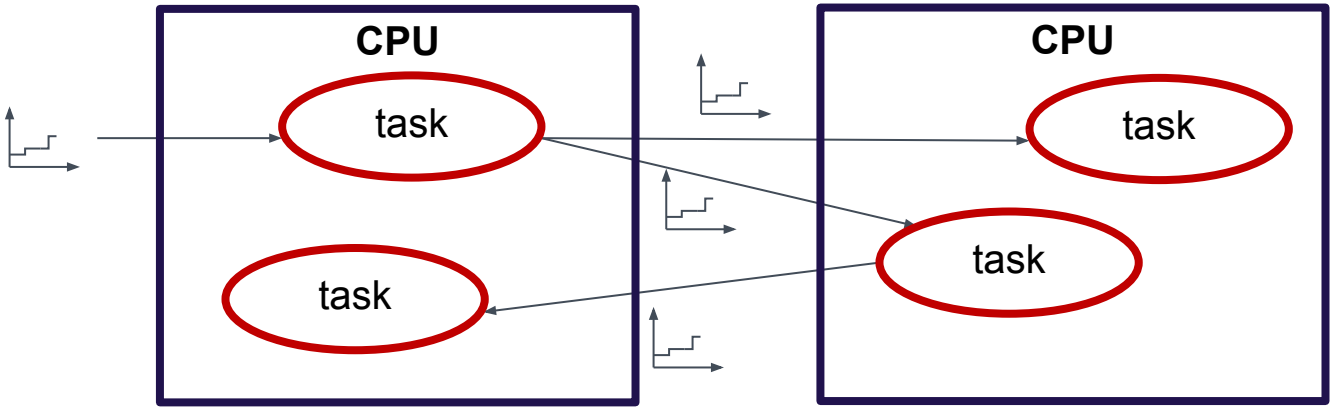
Per-CPU Analysis

Computes task response times given event arrival curves

Fixed-point search

Arrival-Curve Propagation

Computes event arrival curves given task response times



Background: Compositional Performance Analysis

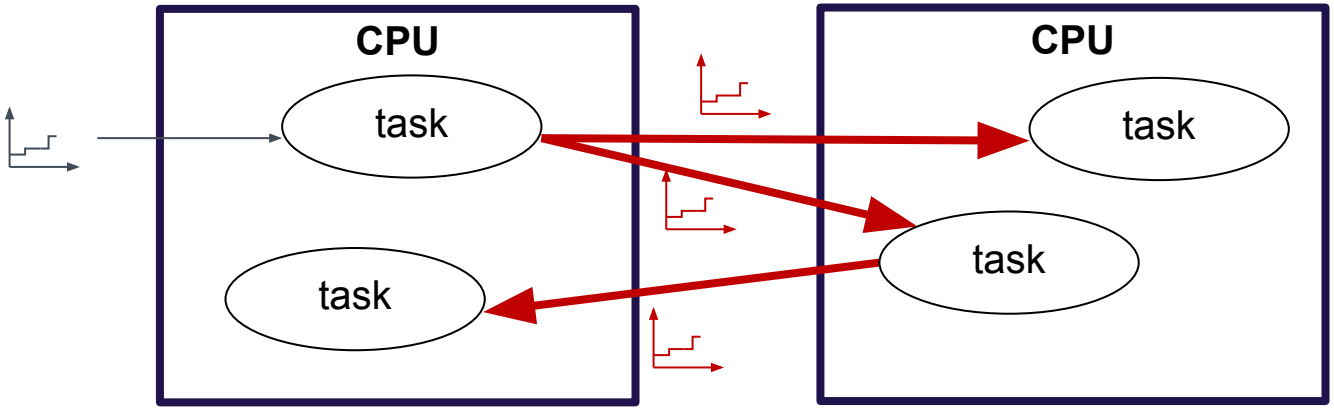
Per-CPU Analysis

Computes task response times given event arrival curves

Fixed-point search

Arrival-Curve Propagation

Computes event arrival curves given task response times



Background: Compositional Performance Analysis

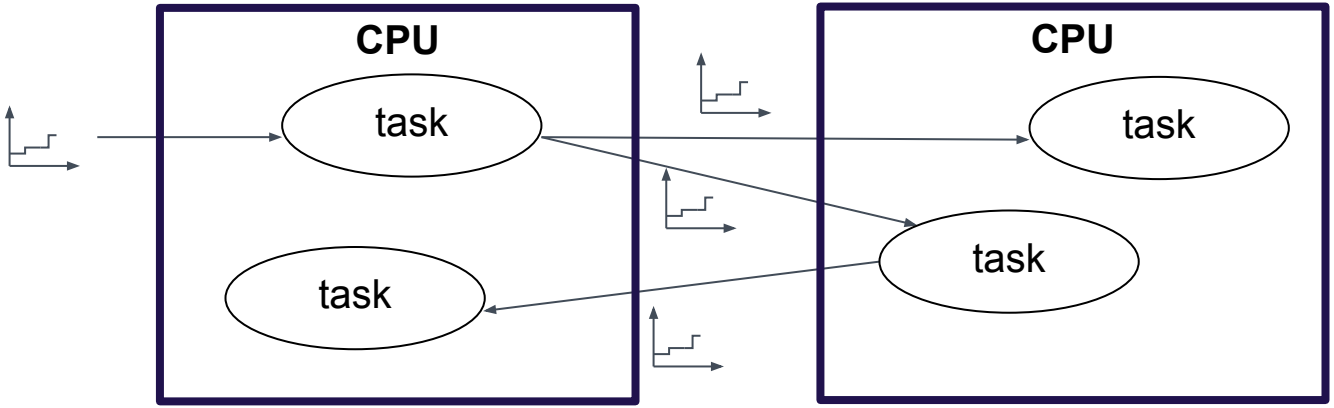
Per-CPU Analysis

Computes task response times given event arrival curves

Fixed-point search

Arrival-Curve Propagation

Computes event arrival curves given task response times



Background: Compositional Performance Analysis

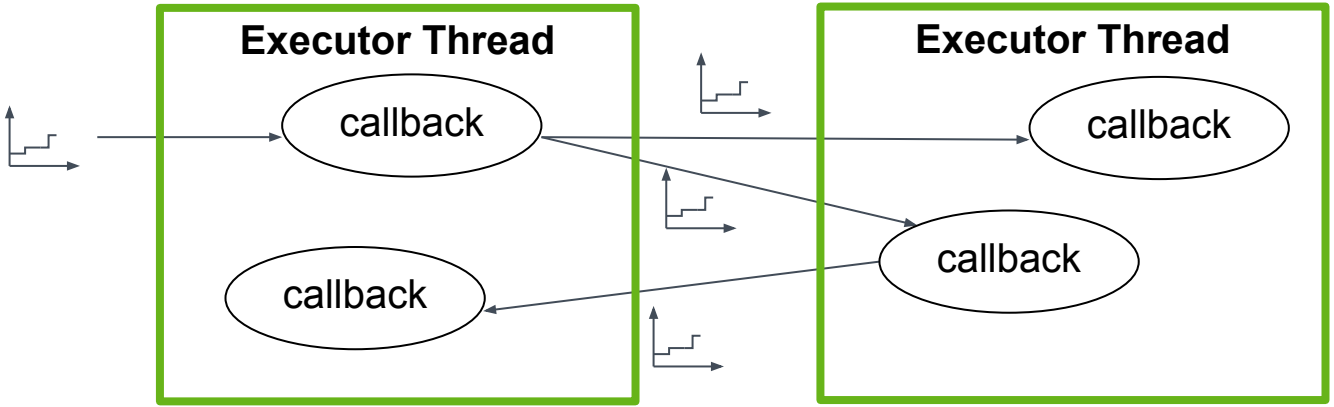
Per-Executor Analysis

Computes callback response times given event arrival curves

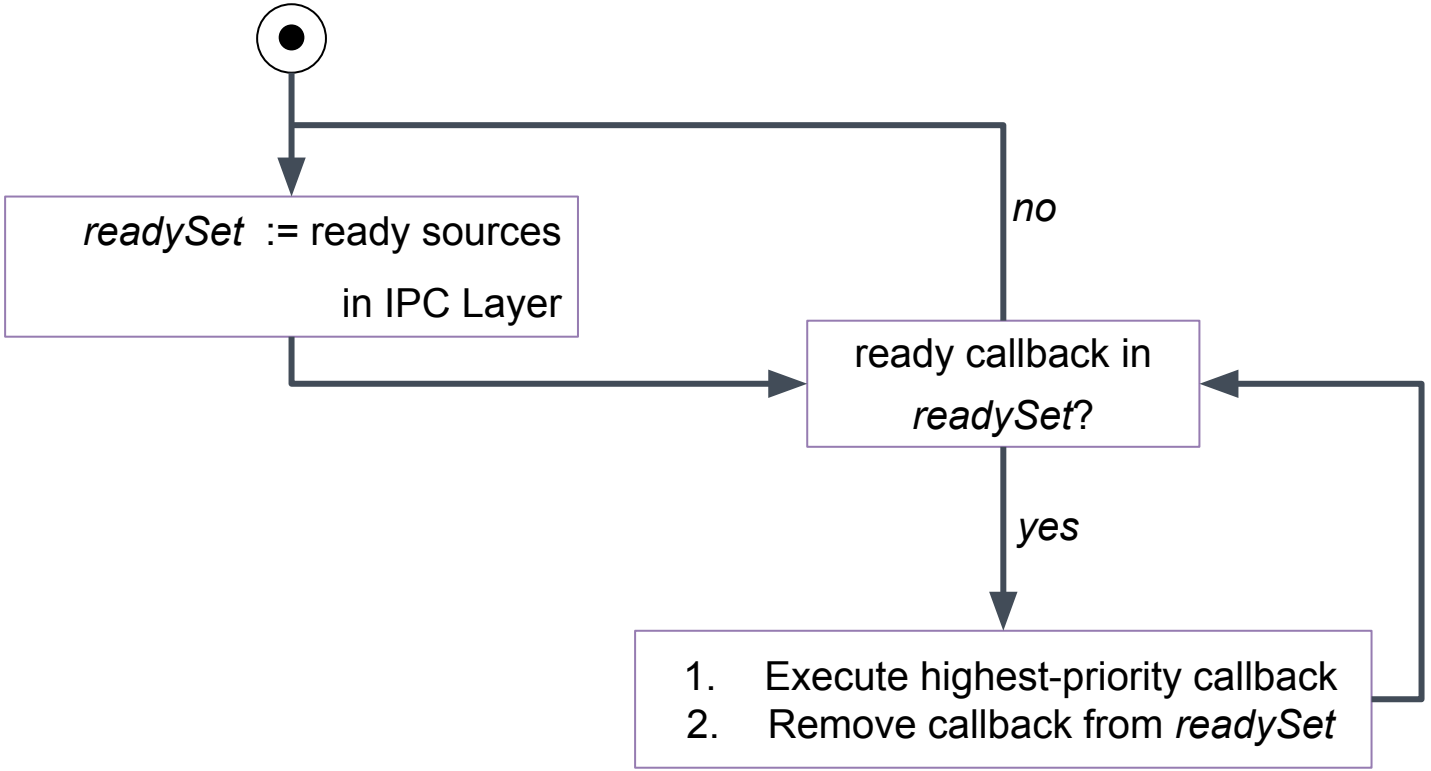
Fixed-point search

Arrival-Curve Propagation

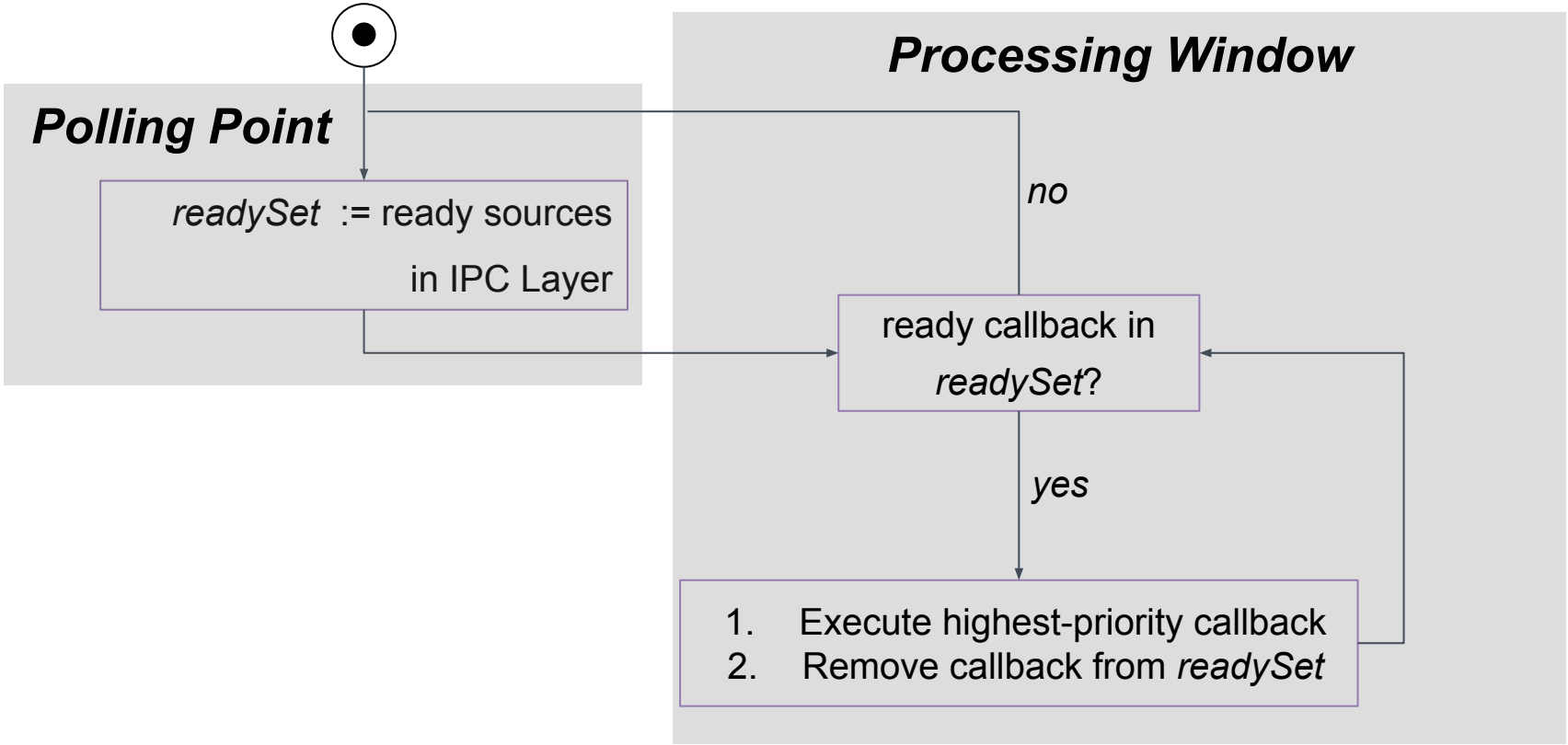
Computes event arrival curves given callback response times



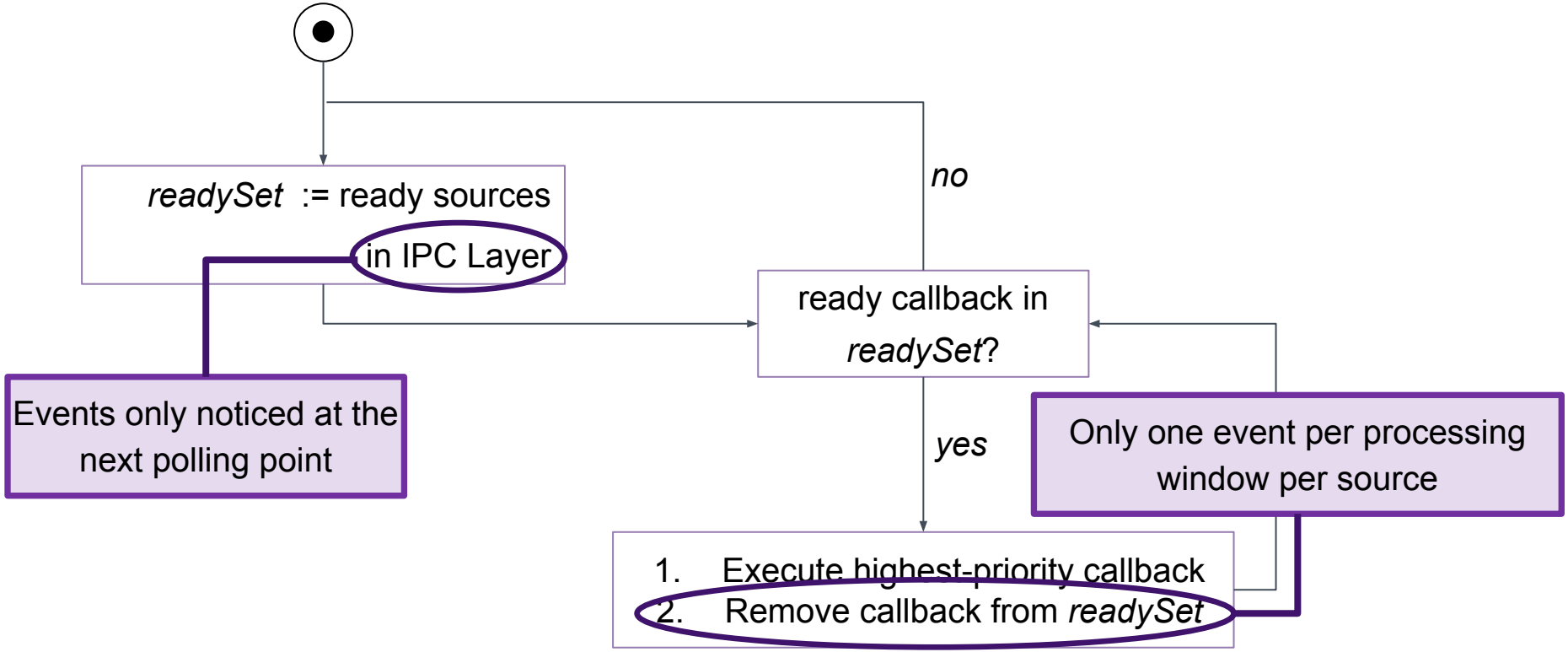
Background: The ROS 2 Executor (simplified)



Background: The ROS 2 Executor (simplified)



Background: Peculiar Properties of the Executor



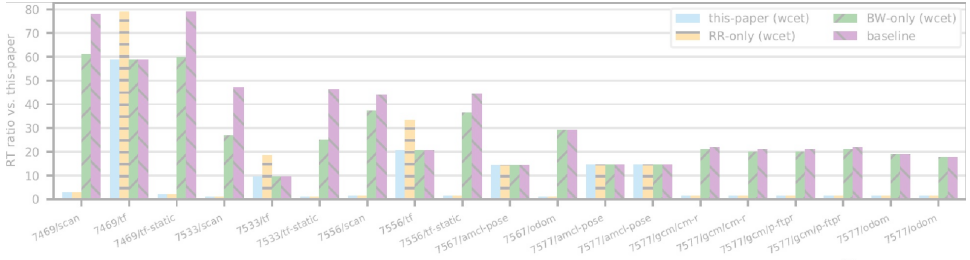
This Paper in a Nutshell

ROS 2 is one of the most popular robotics frameworks, with peculiar timing properties.

We **improve** upon existing response-time analyses with three techniques.

1. Address large **execution-time variance** over time
2. Exploit **starvation-freedom** in the callback scheduler
3. Improve activation-curve **propagation within executors**

Experiments show significant improvements (10-80x) in **real-world ROS packages**.



CR/AEE1 | 2021-11-29

BOSCH

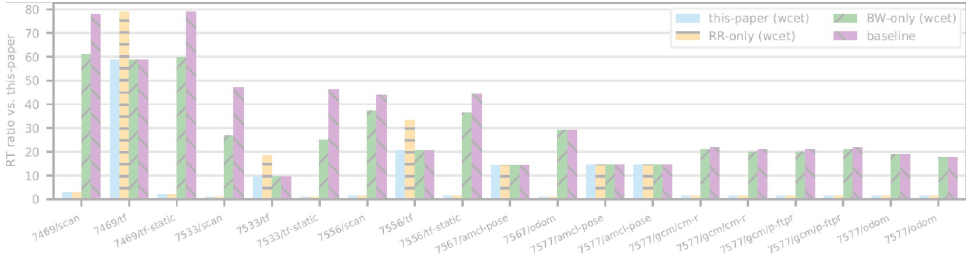
This Paper in a Nutshell

ROS 2 is one of the most popular robotics frameworks, with peculiar timing properties.

We **improve** upon existing response-time analyses with three techniques.

1. Address large **execution-time variance** over time
2. Exploit **starvation-freedom** in the callback scheduler
3. Improve activation-curve **propagation within executors**

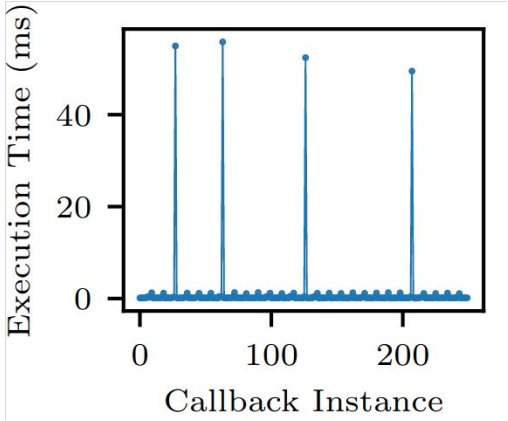
Experiments show significant improvements (10-80x) in **real-world ROS packages**.



BOSCH

Address Large Execution-Time Variance

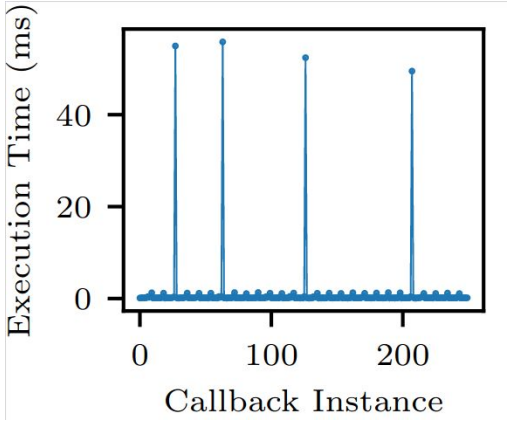
Problem: scalar WCET is too pessimistic



AMCL /tf callback in the navigation 2 package

Address Large Execution-Time Variance

Problem: scalar WCET is too pessimistic



AMCL /tf callback in the navigation 2 package

Solution: Execution-Time Curves
(Quinton et al., 2012)

Bound execution time of multi-instance **sequences**

\+ More precise execution-time model

\- More complex analysis

Details in the paper

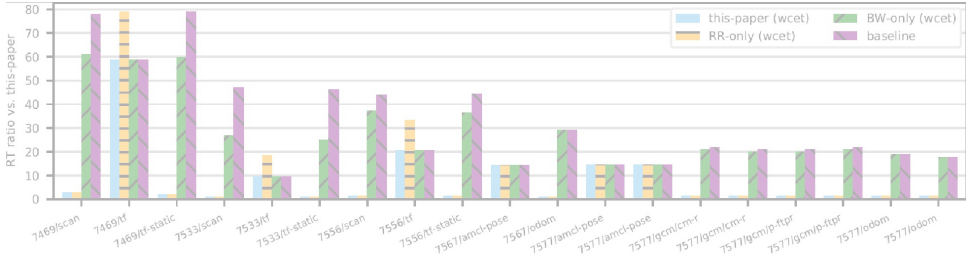
This Paper in a Nutshell

ROS 2 is one of the most popular robotics frameworks, with peculiar timing properties.

We **improve** upon existing response-time analyses with three techniques.

1. Address large **execution-time variance** over time
2. Exploit **starvation-freedom** in the callback scheduler
3. Improve activation-curve **propagation within executors**

Experiments show significant improvements (10-80x) in **real-world ROS packages**.



BOSCH

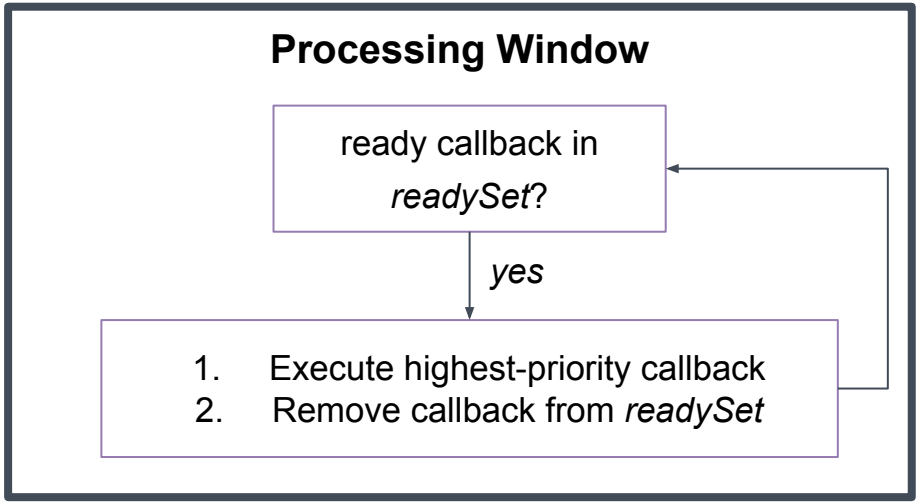
Problem: Bursty Callbacks

Scenario:

- Callback c_1 is triggered periodically
- Callback c_2 triggers bursts of 20 instances
- Assume c_1's response time < c_1's period

Prior Work:

20 instances of c_2 in c_1's busy window.



Pessimistic! Only one instance of c_2 runs in each processing window.

Solution: Round-Robin Analysis

Idea: count polling points

one polling point in this interval
(per self-interfering instance of c_1)

Scenario:

- Callback c_1 is triggered periodically
- Callback c_2 triggers bursts of 20 instances
- Assume c_1's response time < period

\+ Independent of c_2's activation frequency

\- Incompatible with busy-window principle

Combined Analysis

Round-Robin Analysis

- \+ Effective in executors with bursty callbacks
- \- Lacks busy-window principle

Busy-Window Analysis

- \+ Benefits of busy-window principle
- \- Pessimistic in executors with bursty callbacks

Just try both!

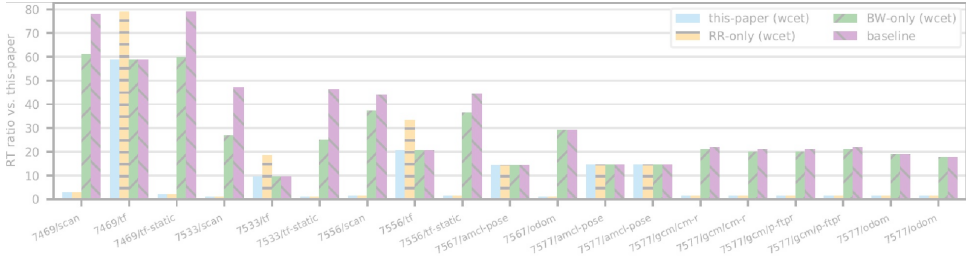
This Paper in a Nutshell

ROS 2 is one of the most popular robotics frameworks, with peculiar timing properties.

We **improve** upon existing response-time analyses with three techniques.

1. Address large **execution-time variance** over time
2. Exploit **starvation-freedom** in the callback scheduler
3. Improve activation-curve **propagation within executors**

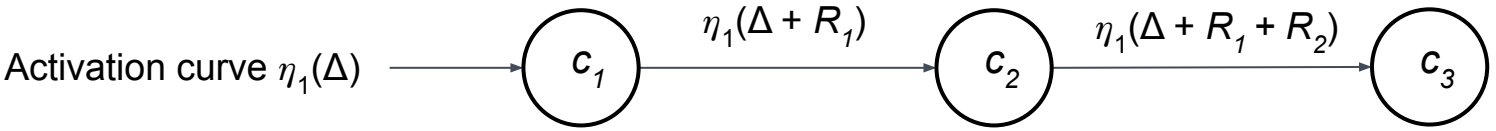
Experiments show significant improvements (10-80x) in **real-world ROS packages**.

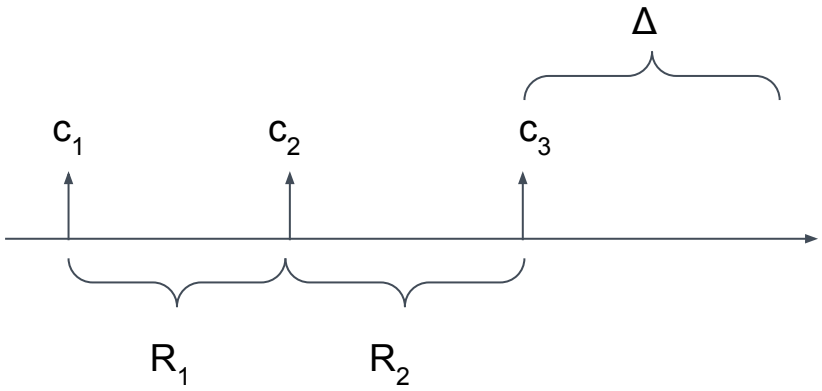


BOSCH

Activation Curve Propagation within Executors

Activation curves are propagated with response-time jitter (Henia et. al., 2005)

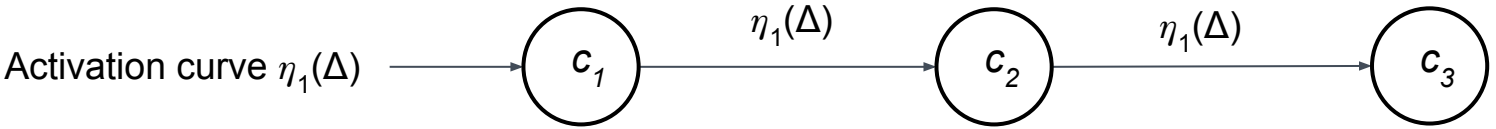


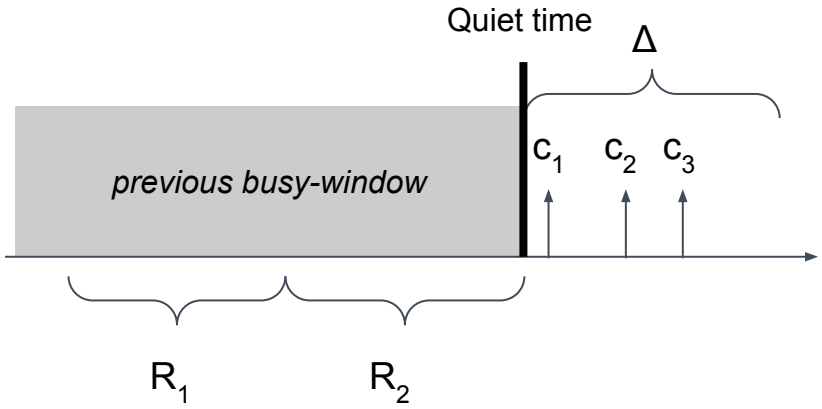


Rationale:
Activations of c_3 ultimately result from activations of c_1 outside the window

Activation Curve Propagation within Executors

If c_1, c_2, c_3 belong to the **same executor** and the Δ-interval starts at a **quiet time**:





Rationale:
Activations of c_1,c_2 cannot cross the quiet time.

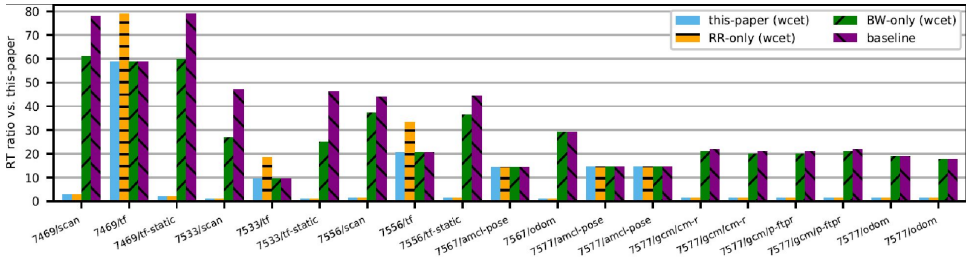
This Paper in a Nutshell

ROS 2 is one of the most popular robotics frameworks, with peculiar timing properties.

We **improve** upon existing response-time analyses with three techniques.

1. Address large **execution-time variance** over time
2. Exploit **starvation-freedom** in the callback scheduler
3. Improve activation-curve **propagation within executors**

Experiments show significant improvements (10-80x) in **real-world ROS packages**.

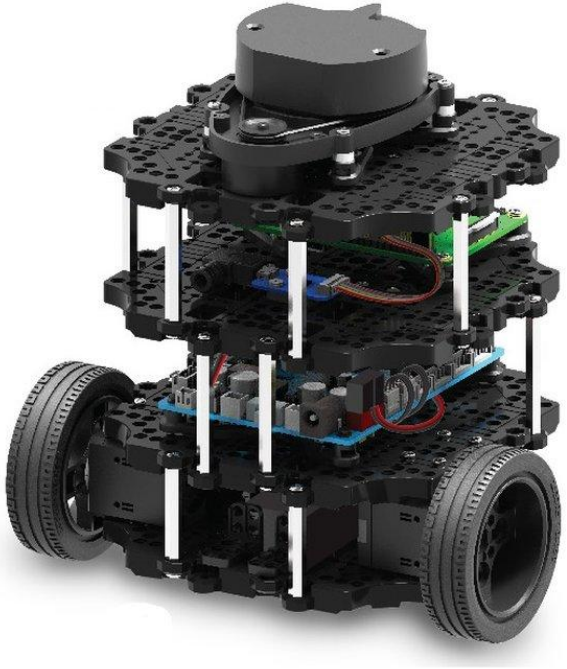


BOSCH

Evaluation

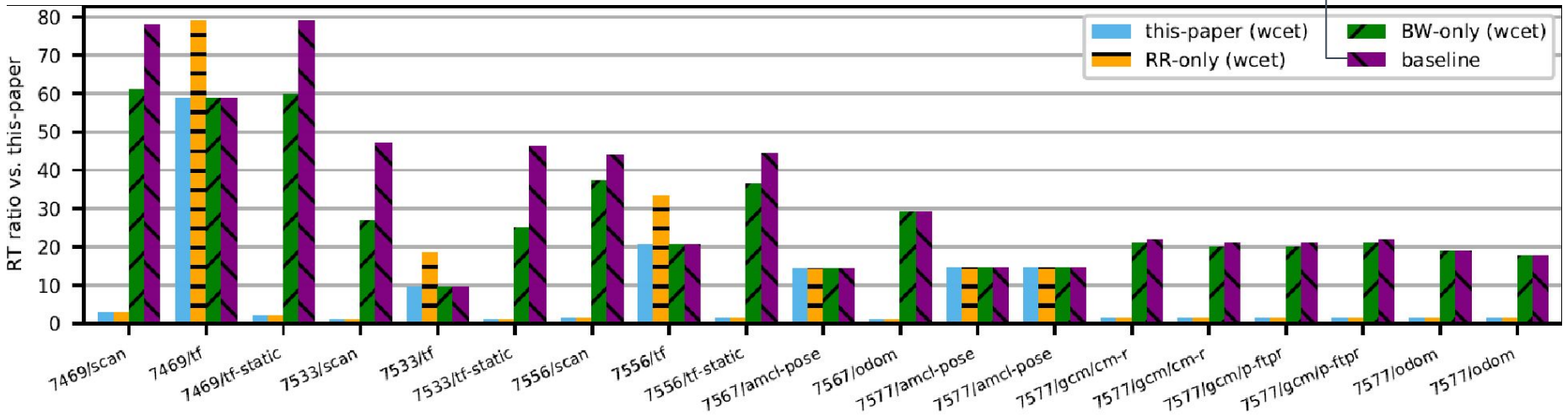
- **Turtlebot 3** “Burger” controlled by a **Raspberry Pi 4B**
- Running various **ROS packages**
  - Navigation 2 packages
  - Turtlebot 3 drivers
- Callback graph extracted from measurements

  See Blass et al., “Automatic Latency Management for ROS 2: Benefits, Challenges, and Open Problems”, RTAS *2021*

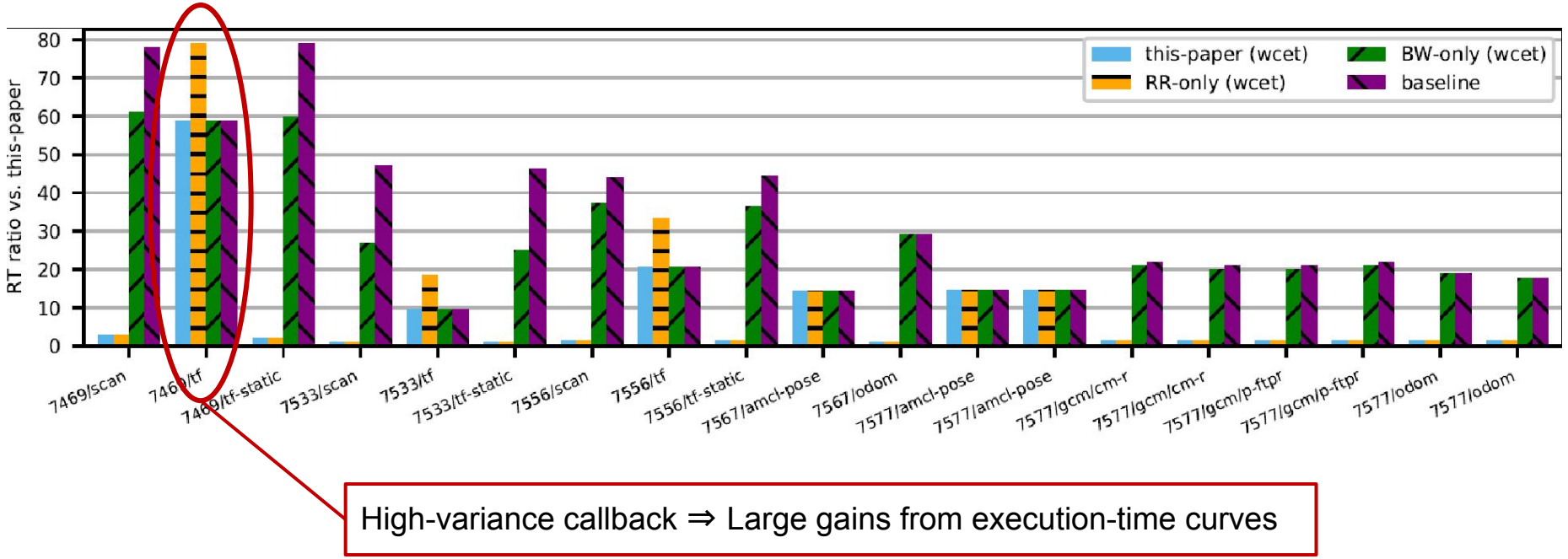


Evaluation

Casini et. al., “*Response-Time Analysis of ROS 2 Processing Chains under Reservation-Based Scheduling*”, ECRTS 2019

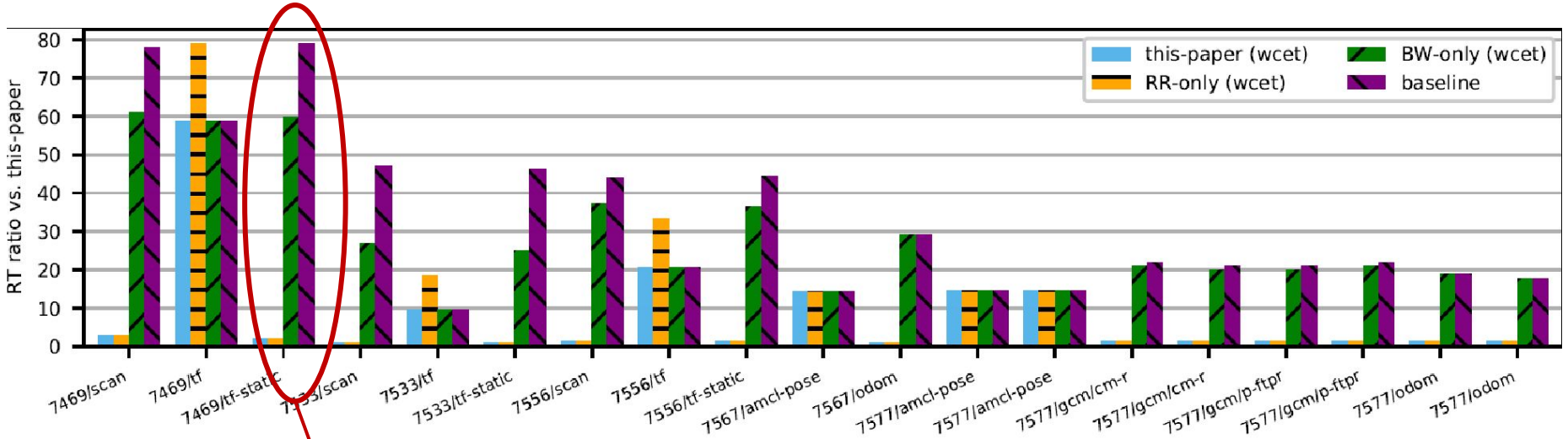


Evaluation



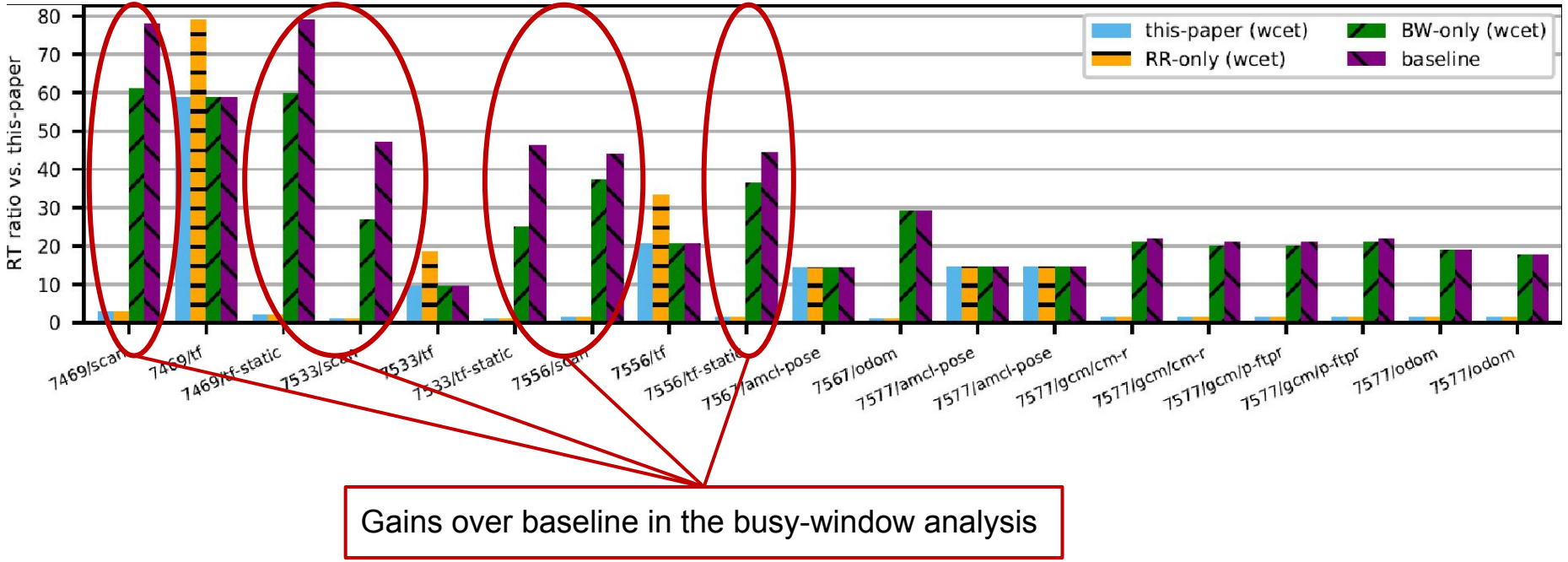
High-variance callback ⇒ Large gains from execution-time curves

Evaluation



Shares executor with bursty callback
⇒ Large gains from round-robin analysis

Evaluation



Gains over baseline in the busy-window analysis

Conclusion

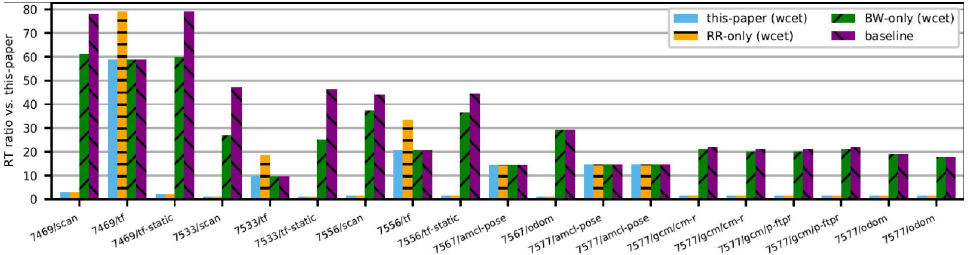
ROS 2 is one of the most popular robotics frameworks, with peculiar timing properties.

ROS

We **improve** upon existing response-time analyses with three techniques.

1. Address large **execution-time variance** over time
2. Exploit **starvation-freedom** in the callback scheduler
3. Improve activation-curve **propagation within executors**

Experiments show significant improvements (10-80x) in **real-world ROS packages**.



BOSCH