Modeling and Scheduling of Fusion Patterns in Autonomous Driving Systems

Hoora Sobhani and Hyoseung Kim

University of California, Riverside

hsobh002@ucr.edu, hyoseung@ucr.edu

33rd International Conference on Real-Time Networks and Systems

Nov 5-7, 2025. Pisa, Italy

RTNS 2025

The Rise of Autonomous Driving Systems (ADS)

- Reduce traffic accidents,
- Ease congestion,
- Lower emissions,
- etc.

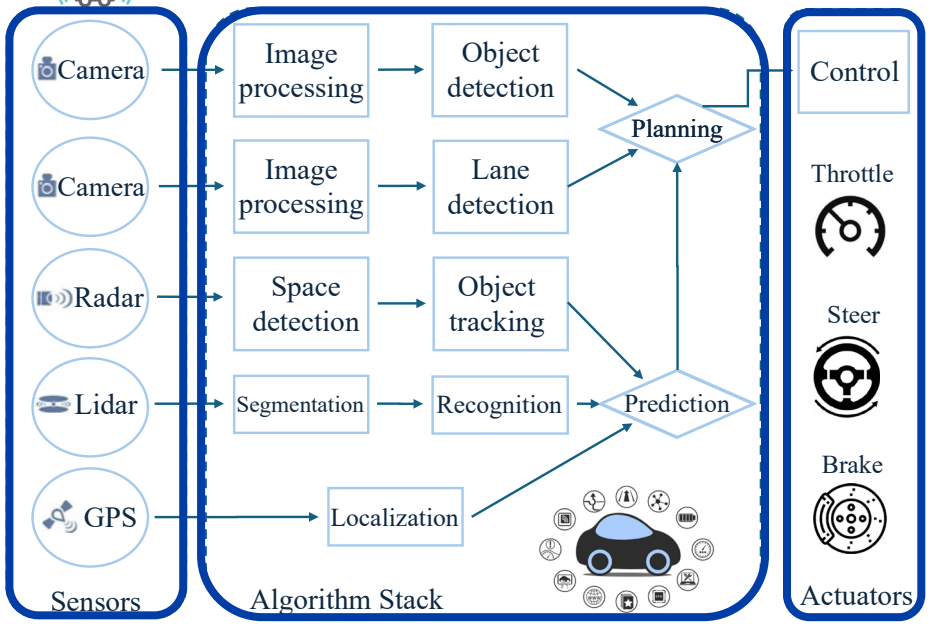
Ensure safe, intelligent, and reliable transportation systems

- ADS Software Stack:
  - Integration of advanced technologies (computer vision, machine learning, sensor fusion)
  - Significant challenges in modeling and task scheduling:
    - Highly complex real-time system
    - Safety-critical nature
    - Resource-efficiency requirements

A Closer Look at the Challenges

of Modeling and Scheduling of ADS Software Stack

Holistic overview of the ADS software stack

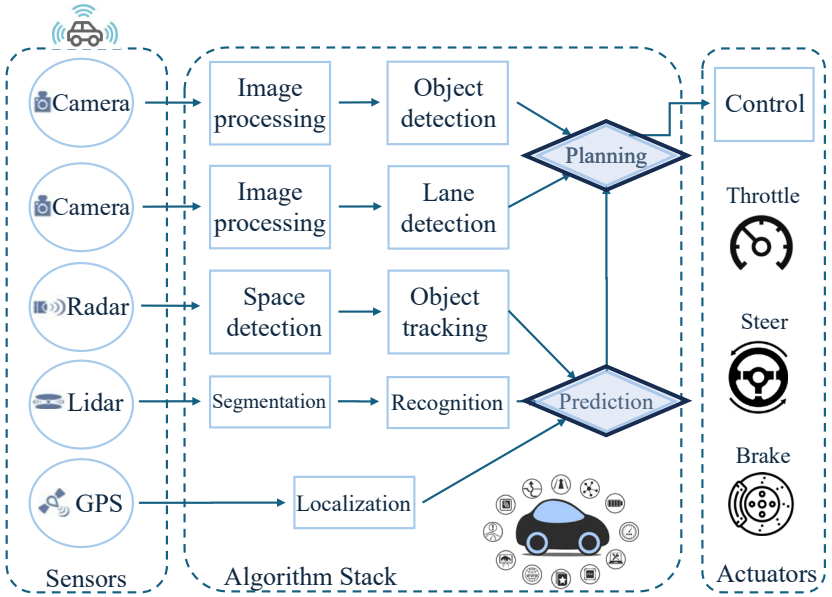


The picture is inspired by [Xu22, kuhse24]

- Numerous sensors with different data sampling rates
- Complex task dependencies and inter-tasks communication
- Multiple triggering options for tasks--which can be triggered by events or timers
- Diverse data paths from a sensor to an actuator—including fork-then-join paths (i.e. branch-then-fusion)
- Difficult to measure proper metrics for several actuator tasks
  (sensors time disparity, system reaction time, responsiveness, etc.)

Modeling and scheduling of ADS Software Stack is **definitely not easy**!!!

Core Challenge: Diverse Fusion Patterns



- Multiple inputs:
  - originating from **different sensors**,
  - being sampled at **different rates**,
  - being processed in **different paths**.
- Diverse triggering options:
  - own timer
  - external event
    - triggered when **all inputs** arrive
    - triggered when **any input** arrives
- Co-existence of diverse fusion nodes

Directed Acyclic Graphs (DAGs):

- Effectively capture the system's workflow.
- Accurately model task dependencies and inter-task communication.
- Enable measurement of key metrics such as end-to-end latency, data freshness, and system responsiveness.

- Sensors: source nodes in DAG
- Algorithm Stack: middle nodes in DAG
- Actuators: sink nodes in DAG

Limitations of Existing Models for Fusion Patterns

ADS software stack modeling and scheduling have been widely studied;

however, most existing DAG models for ADS:

- Assume **a single triggering pattern** for fusion nodes, incapable to capture some real-world ADS behaviors:
  - Timer-triggered only (e.g., multi-rate DAG [Li24, Sun23])
  - Event-triggered (one dominant edge) – only one input activates the fusion node (e.g., *trigger/update* [Toba24], *passive/trigger* [Teper22])
  - Event-triggered (any input) – any input edge activates the fusion node [Sun25]

- Consider **various real-time metrics, but not all simultaneously**:
  - Maximum Reaction Time (MRT) [Durr19]
  - Maximum Data Age (MDA) [Durr19]
  - Maximum Time Disparity (MTD) [Jiang23]
  - Age of Information (AoI) [Xu22]

- Rely on **special assumptions**, making them difficult to use together and broadly

- Focus on **induvial path instead of the DAG** (ignoring branch nodes, multiple sinks)

Related Work

- Environments and platforms
  - Traditional real-time OS [Xu22] vs. Middleware (e.g. ROS 2) [Sun23] vs. Real-time software architecture (e.g. AUTOSAR) [Gunzel23]
  - Single-core [Teper22] vs. multi-core[Teper24]

No prior work addresses a comprehensive framework to handle various fusion behaviors found in real-world ADS as a DAG

- Chain types:

	Multi-rate Cause-Effect Chains	Enhanced Cause-Effect Chains
Task triggering	A sequence of multi-rate tasks, each triggered by timers with different rates.	A cause-effect chain enabling both event- and timer-based triggers.
ROS / Non-ROS	Non-ROS: [Becker16], [Becker17], [Saidi17], [verucchi20], [Tang22], [Maia 24], [Jiang23] ROS2: [Saito18], [Li23], [Sun23J], [Sun23S]	Non-ROS: [Tang23R], [Tang23O],[Sun25] ROS2: [Teper22],[Gunzel23], [Teper24], [Yano24]
Metrics	Maximum reaction time (MRT) Maximum data age (MDA) Maximum time disparity (MTD) Age of Information (AoI)	Maximum reaction time (MRT) Maximum data age (MDA) Maximum time disparity (MTD)

Proposed Work

Modeling and Scheduling Fusion Patterns in ADS

Unique Contributions of Our Framework:

- **Model various fusion patterns in ADS, supporting diverse chains and triggering options unmet by existing models**
- **Enable quantitative comparison to analyze the impact of different fusion patterns on real-time performance metrics**
- **Optimize multiple real-time metrics on multi-core platforms with a DAG scheduler using Integer Linear Programming (ILP)**

→ ILP: to explore the best achievable performance (under different fusion patterns)

→ With this framework, it is possible to faithfully represent real-world ADS behaviors in DAG and co-optimize various real-time metrics

- No prior work can do this

System Model

DAG:

$$G = (V, E)$$

Vertices (Nodes, Tasks) — Edges

Chain:

A path within the DAG from a source node to a sink node

A sequence of dependent tasks in a set order from a sensor to an actuator

Task:

WCET — Period — Deadline — Type — Set of adjacent predecessor tasks

$$\tau_i = \{e_i, T_i, D_i, \theta_i, pred_i\}$$

Task Instance:

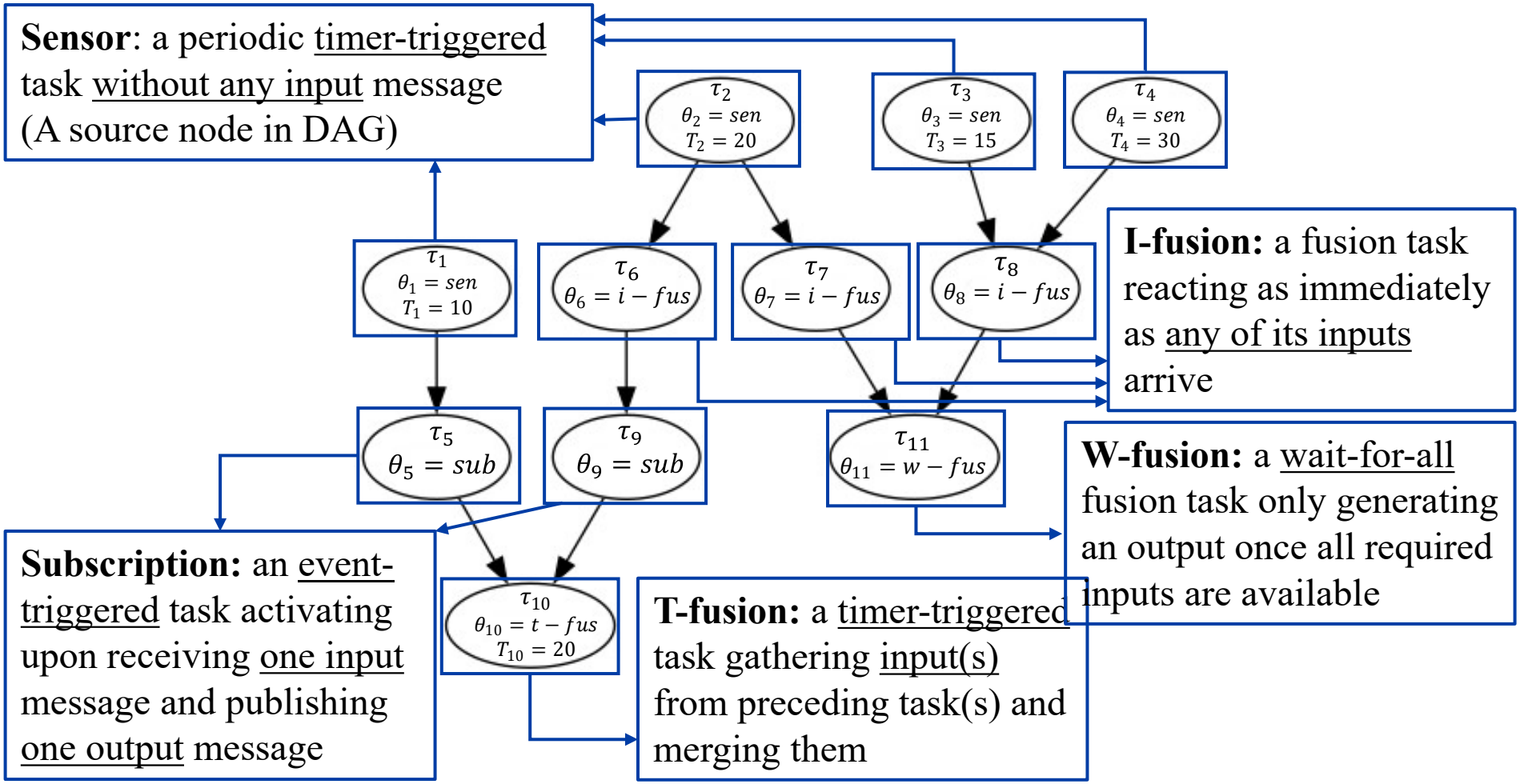
- $\tau_{i,j}$: j-th instance of task τ_i

Multi-core:

- Identical cores
- Instances of a task may run on different cores

* See Appendix for general real-time requirement formulations

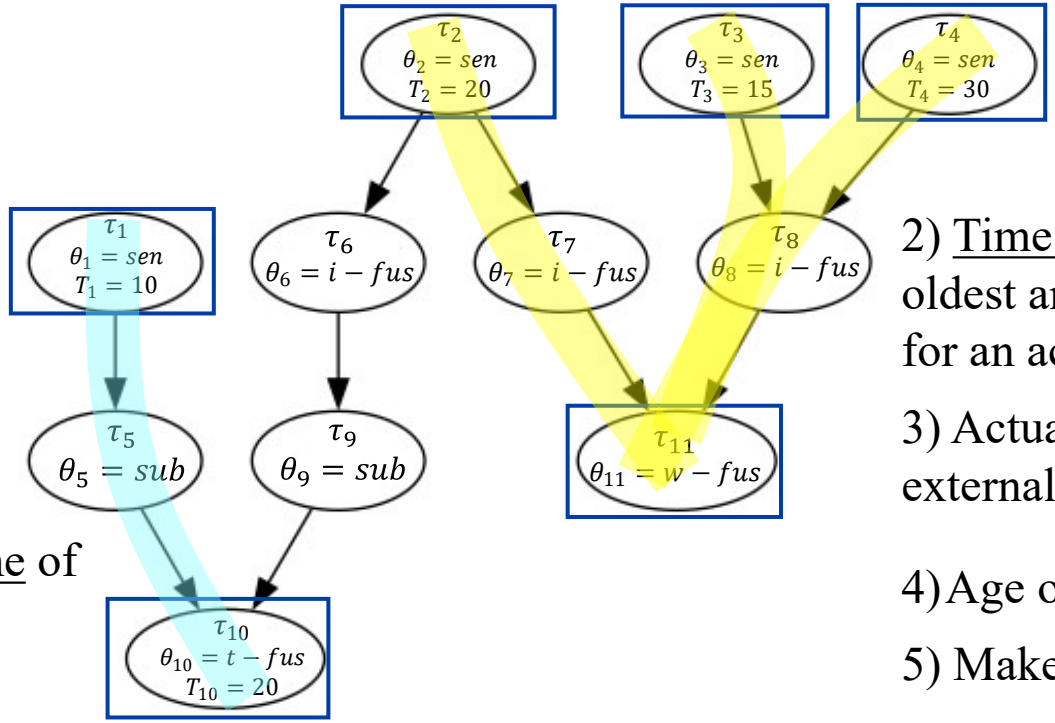
Task Type Model



** Sensor & subscription node details in the paper

** A branch node can be any type; its successor is modeled as **I-fusion** if it is a subscription node

Real-time Performance Metrics



1) Response time of particular paths

2) Time disparity between the oldest and newest sensor data for an actuator

3) Actuator's reaction time to an external event

4) Age of Information

5) Makespan

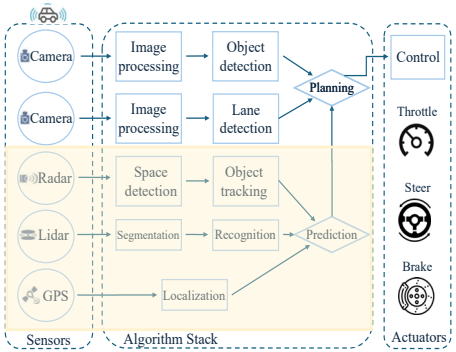
6) etc.

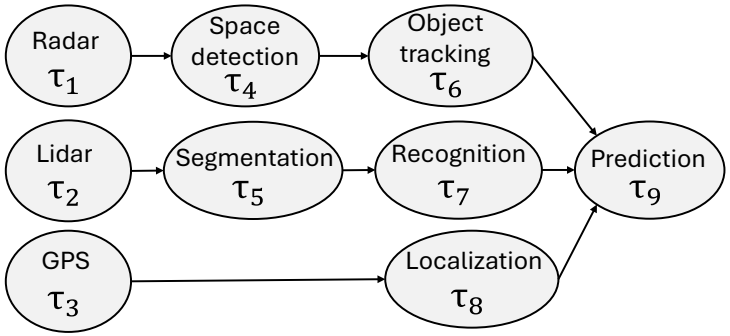
- How to co-optimize all these, while considering:
  - Different periods of time-triggered nodes
  - Dependency of subscription nodes
  - Activation conditions of T/W/I-fusion nodes

Formulation of Fusion Tasks

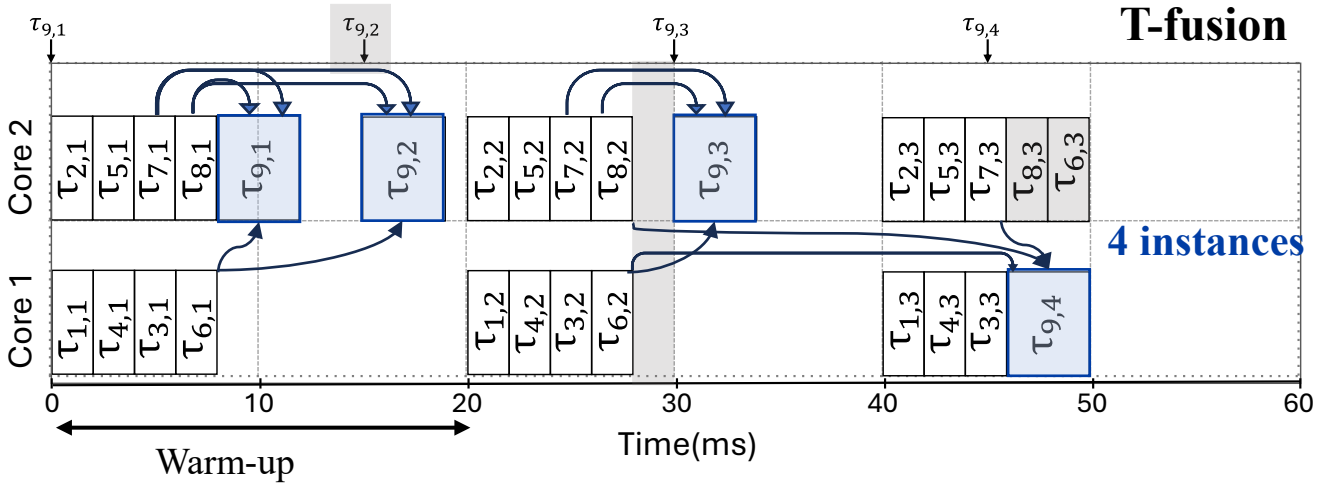
Major Challenges

- **Hard to determine which predecessor instance contributes to a fusion instance**
- **Even harder to determine which sensor instance contributes to an actuator instance (through intermediate fusion nodes in the DAG)**





`ndt_scan_matcher`
(e.g. in Autoware)



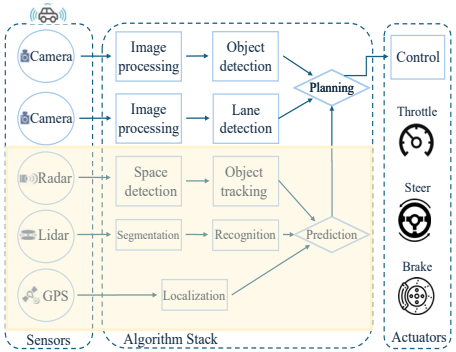
Illustrative example:

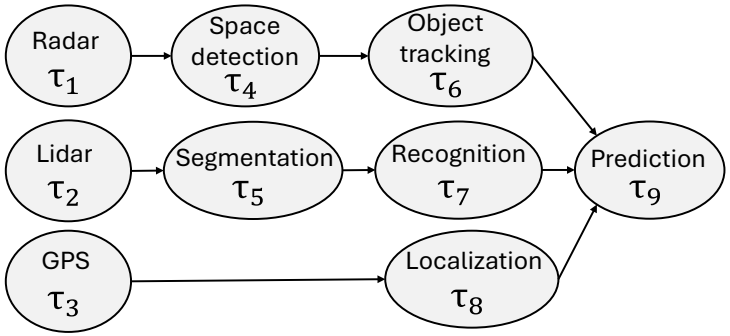
- Tasks 1-3 (sensor):
  - WCET=2ms, period = 20ms
- Tasks 4-8 (subscription):
  - WCET = 2ms
- Task 9 (T-fusion):
  - WCET=4ms, period = 15ms
- HP = 60ms

Formulation of Fusion Tasks

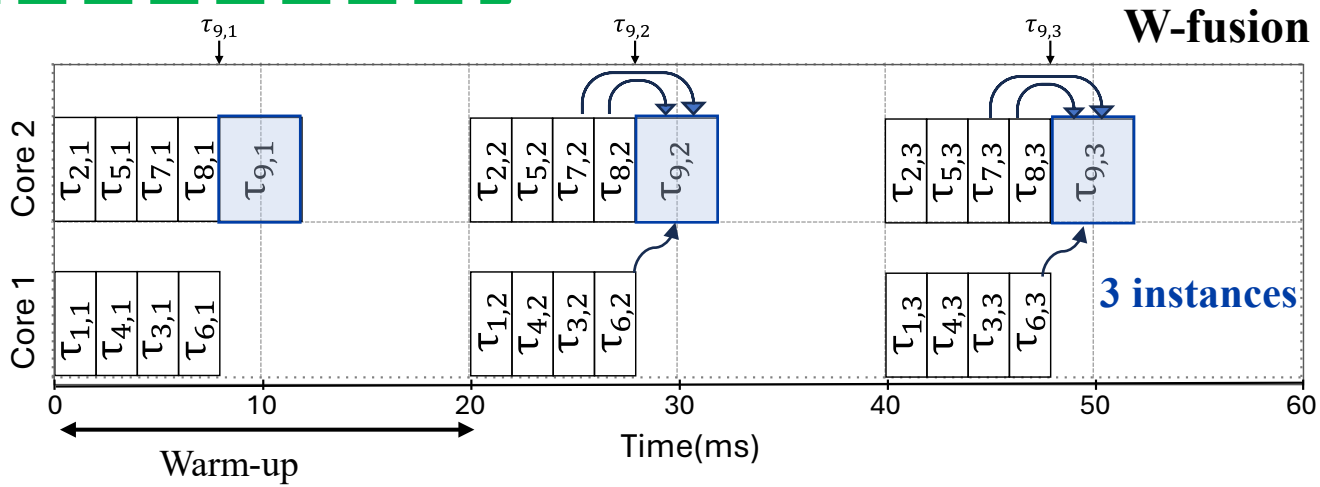
Major Challenges

- **Hard to determine which predecessor instance contributes to a fusion instance**
- **Even harder to determine which sensor instance contributes to an actuator instance (through intermediate fusion nodes in the DAG)**





`tracking_object_merger`
(e.g. in Autoware)



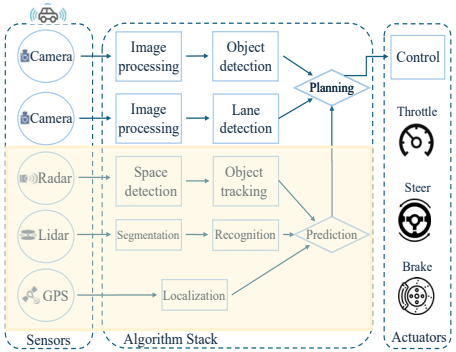
Illustrative example:

- Tasks 1-3 (sensor):
  - WCET=2ms, period = 20ms
- Tasks 4-8 (subscription):
  - WCET = 2ms
- Task 9 (T-fusion):
  - WCET=4ms, period = 15ms
- HP = 60ms

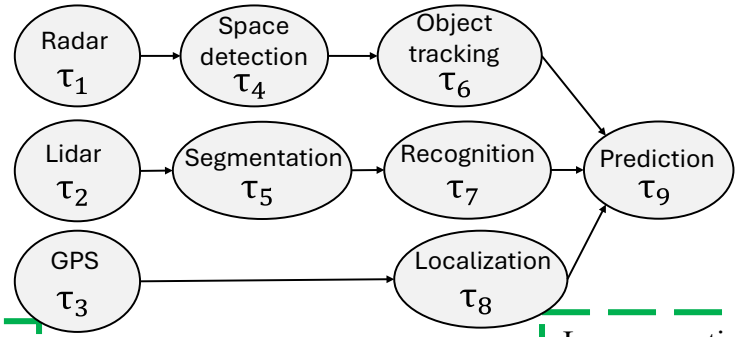
Formulation of Fusion Tasks

Major Challenges

- **Hard to determine which predecessor instance contributes to a fusion instance**
- **Even harder to determine which sensor instance contributes to an actuator instance (through intermediate fusion nodes in the DAG)**



`traffic_light_occlusion_predictor`
(e.g. in Autoware)

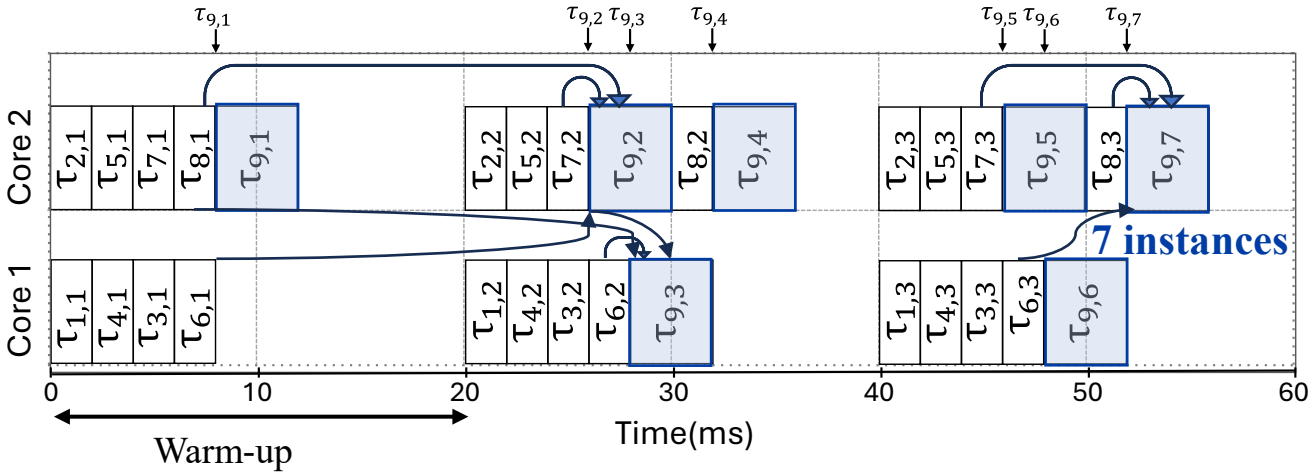


Lower reaction time

I-fusion

Illustrative example:

- Tasks 1-3 (sensor):
  - WCET=2ms, period = 20ms
- Tasks 4-8 (subscription):
  - WCET = 2ms
- Task 9 (T-fusion):
  - WCET=4ms, period = 15ms
- HP = 60ms



Formulation of Fusion Tasks

- **Uncertainty in identifying instance contributions**
- **Producer nodes** (sensors + fusion nodes): root of a new task instance with a new *#instance* or *j*
- Subscription tasks inherit the instance numbering of their producer
- ***pred_producers(τ_i)***: the set of producers of τ_i's predecessors

Where, τ_i: a fusion task
$\tau_p \in pred\_producers(\tau_i)$

$$u_{(i,j),(p,j_p)} = \begin{cases} 1, & if\ \tau_{i,j}\ uses\ the\ j_p\text{-}th\ instance\ of\ \tau_p \\ 0, & otherwise \end{cases}$$

Start-time of fusion nodes:

- Each fusion node instance begins only after the appropriate predecessor instance completes

$$s_{i,j} \geq f_{i',j'} \cdot u_{(i,j),(producer(\tau_{i'}),j')}, \quad \forall \tau_{i'} \in pred_i$$

✓ $\tau_{i,j}$'s start-time depends on the finish-time of its predecessor
✓ $\tau_{i',j'}$ only if j'-th instance of $\tau_{i'}$ or its producer is used

Formulation of Fusion Tasks

- Each fusion node instance uses only one (the most recent) instance from each producer

$$\sum_{j'=1}^{n\text{-}ins(\tau_{i'},\Delta)} u_{(i,j),(producer(\tau_{i'}),j')} = 1, \ \forall \tau_{i'} \in pred_i$$

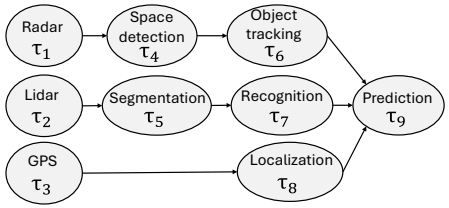
$$\sum_{j'=1}^{n\text{-}ins(\tau_{i'},\Delta)} j' \cdot u_{(i,j),(producer(\tau_{i'}),j')} \leq \sum_{j'=1}^{n\text{-}ins(\tau_{i'},\Delta)} j' \cdot u_{(i,j+1),(producer(\tau_{i'}),j')}, \ \forall \tau_{i'} \in pred_i$$

T-fusion	W-fusion	I-fusion
Sensor formulation + Fusion formulation	A fusion task can use each predecessor's producer instance only once. $\sum_{j=1}^{n\text{-}ins(\tau_i,\Delta)} u_{(i,j),(producer(\tau_{i'}),j')} \leq 1, \ \forall \tau_{i'} \in pred_i$ ** T-fusion or I-fusion: the same producer instance can be used multiple times	Triggers when at least one predecessor provides a new instance $\sum_{\tau_{i'} \in pred_i} \left(\sum_{j'=1}^{n\text{-}ins(\tau_{i'},\Delta)} j' \cdot u_{(i,j+1),(producer(\tau_{i'}),j')} - \sum_{j'=1}^{n\text{-}ins(\tau_{i'},\Delta)} j' \cdot u_{(i,j),(producer(\tau_{i'}),j')} \right) \geq 1$

Formulation of Real-time Performance Metrics

Metrics optimized in our work:

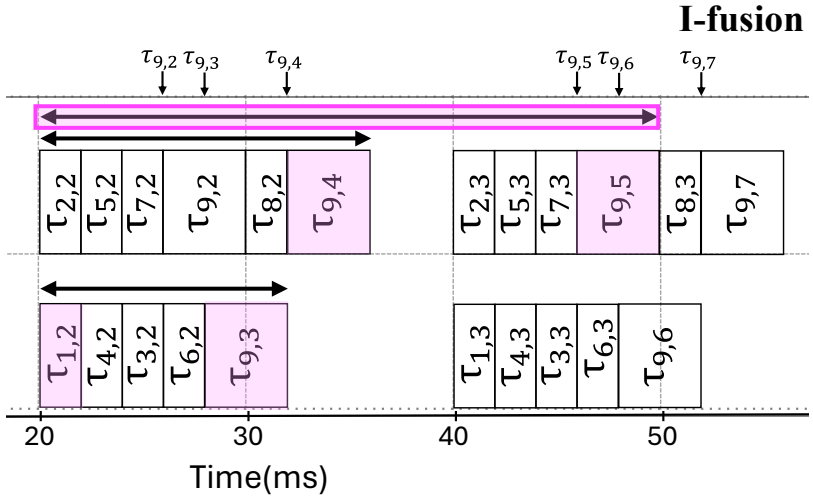
- **Maximum Reaction Time (MRT)**
- **Maximum Time Disparity (MTD)**
- **Peak Age of Information (PAoI)**
- **Worst-case response-time (WCRT)**
- **Makespan (MS)**



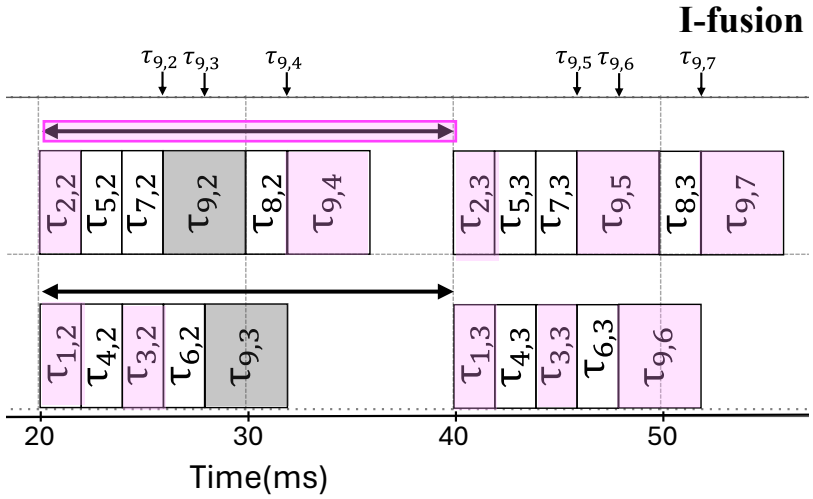
End-to-end metrics: Timing relationships between sensors and actuators (source and sink nodes in the DAG)

Requires identifying **which sensor instances contributed to each actuator instance.**

MRT



MTD



Formulation of Real-time Performance Metrics

Major Challenges

- Hard to determine: Multiple fusion nodes may exist between a sensor and an actuator
- **Even harder to determine which sensor instance contributes to an actuator instance**

$$U_{(i,j),(s,j_s)} = \begin{cases} 1, & \text{if instance } \tau_{i,j} \text{ uses the } j_s\text{-th instance of sensor } \tau_s \\ 0, & \text{otherwise} \end{cases}$$

U(i,j),(s,j_s) indicates whether a sink/fusion task instance $\tau_{i,j}$ uses the j_s-th instance of a sensor task τ_s.

Computation of U:

- depends on the type of $\tau_{i,j}$ and its distance from the sensor τ_{s,j_s}
- Derived iteratively using variable u and the follow
- Refer to our paper for more details.

$$\sum_{\tau_p \in pred\_producers(\tau_i)} \sum_{j_p=1}^{n\text{-}ins(\tau_p,\Delta)} u_{(i,j),(p,j_p)} \cdot U_{(p,j_p),(s,j_s)} \geq 1$$

Computation of real-time metrics:

$$PAoI(\tau_\otimes) = max\{(s_{s,j_s} - s_{s,j_s-1}) \cdot U_{(\otimes,j),(s,j_s)} | \\ \forall j_s \in [1, n\text{-}ins(\tau_s, \Delta)], \quad \forall \tau_s \in V, \quad \forall j \in [1, n\text{-}ins(\tau_\otimes, \Delta)]\}$$

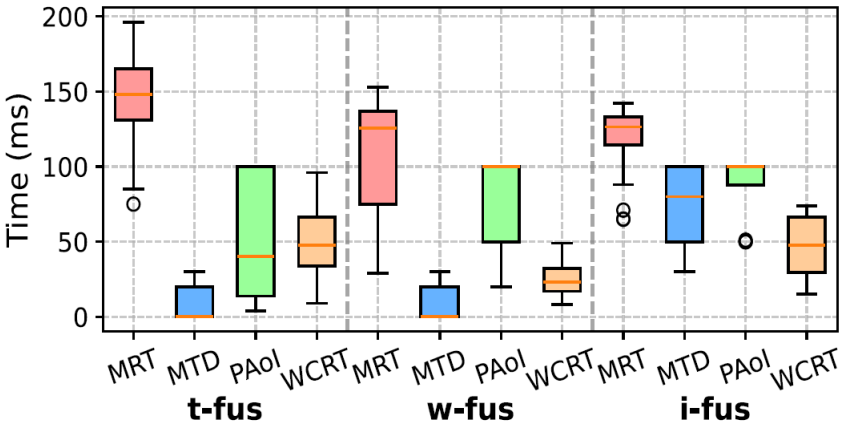
Evaluation

Impact of fusion patterns

on real-time metrics

Fusion Type Impact

(6 nodes including 3 sensors, 7 edges)



- **Randomly-generated DAGs**
- Each experiment: 100 DAGs
- **Timer-triggered tasks:** utilization ∈ [0.1, 0.4]; period ∈ {20, 40, 50, 100} ms; WCET = utilization × period
- **Event-triggered tasks:** WCET ∈ [1, 5] ms
- Π = 2

Fusion types and tradeoff among metrics:

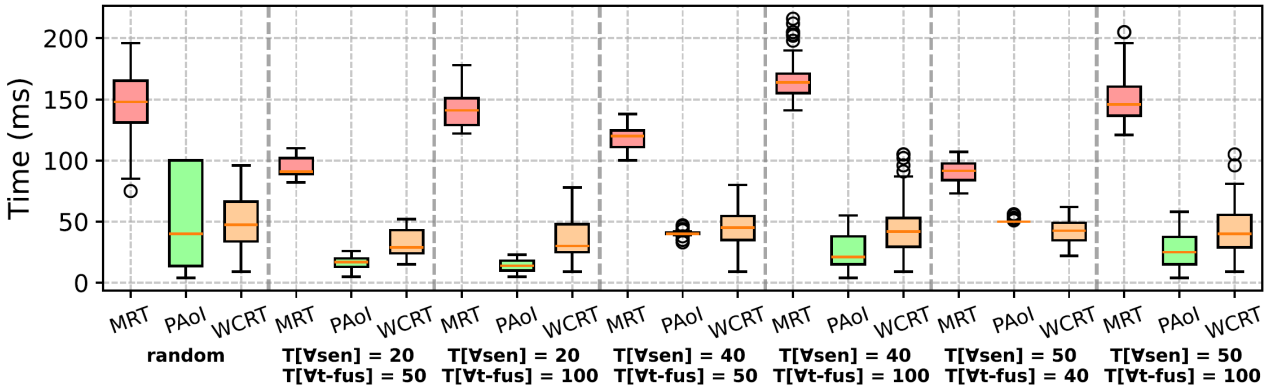
- I-fusion ↓ MRT, ↑ MTD
- T-fusion may ↑ MRT if periods poorly chosen

Period Tuning Insights:

- Shorter sensor periods → lower PAoI
- Aligned sensor & fusion periods → better MRT

Task Period Impact

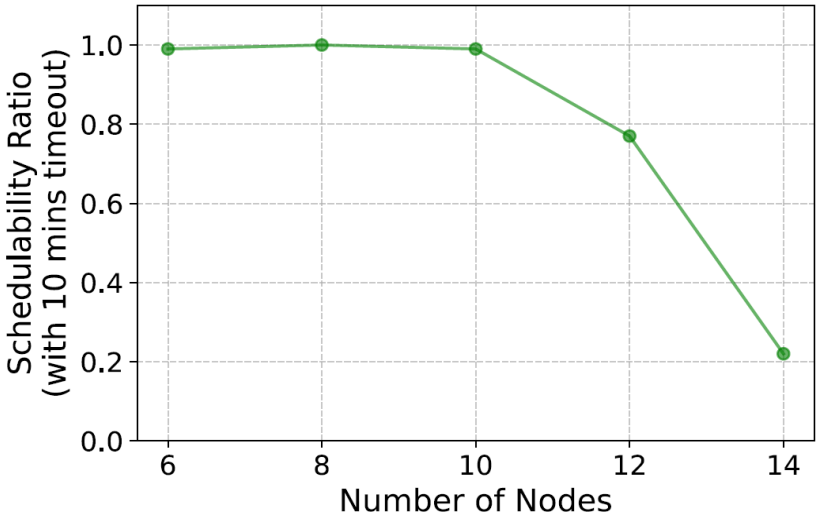
(Only T-fusion allowed)

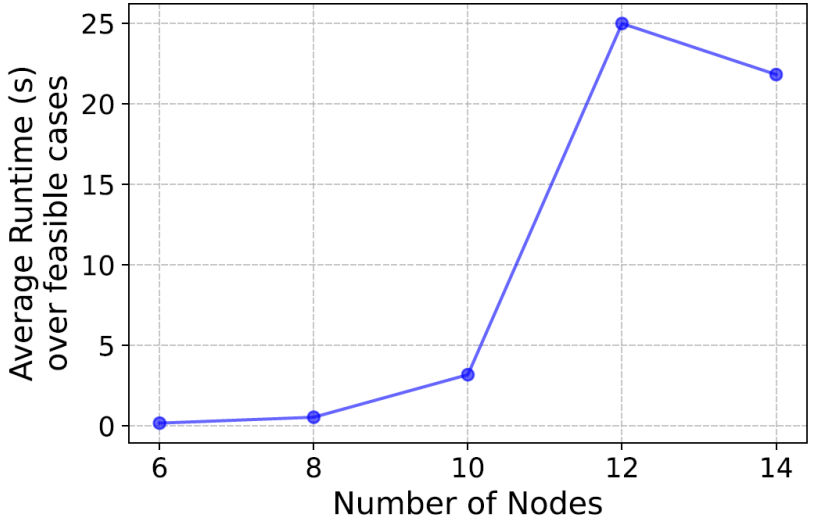


Evaluation

Impact of DAG Complexity on Computation Cost

- **Randomly-generated DAGs**
- Each experiment: 100 DAGs
- **Timer-triggered tasks:** utilization ∈ [0.1, 0.4]; period ∈ {20, 40, 50, 100} ms; WCET = utilization × period
- **Event-triggered tasks:** WCET ∈ [1, 5] ms
- #Sensors = ½ × #nodes
- #Edges = 2 × #nodes
- only W-fusion allowed
- Π = 2





- ↑ DAG complexity, ↑ MRT and WCRT
- ↓ feasible cases on Π=2
- ↑ Average runtime → ILP Scalability

* See the extended version of our paper (Appendix) for additional experiments

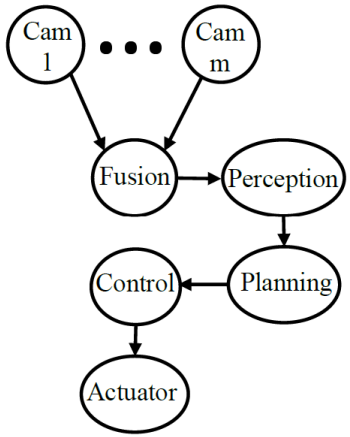
Evaluation

Case Study:Navigation System on Raspberry Pi 4

#camera	UB*	LB*	IDS: MRT	Observed MRT on Raspberry Pi**
1	480	155	155	155.06
2	515	160	160	159.85
3	840	165	165	164.94
4	890	170	170	170.37
5	940	175	175	175.38
6	990	180	180	180.31
7	1040	185	185	184.94
8	1090	190	190	190.48
9	1140	195	195	194.93
10	1190	530	200	200.01

* UB and LB: [Teper22]'s upper- and lower-bound on MRT

** Observed MRT: from static user-level scheduler on Linux, run for 100 hyperperiods on Raspberry Pi



MRT: for comparison with prior work

Other optimized metrics:

- MTD = 0, PAoI = 100 for all camera counts (fixed camera periods)
- MS ∈ [255, 300], WCRT ∈ [55, 100] (5 ms increments per additional camera)

- IDS:MRT and observed MRT closely match the MRT in prior work.
- Supports **diverse fusion patterns** while optimizing **multiple real-time metrics**

Conclusion

IDS Framework: a systematic framework for modeling and scheduling fusion patterns in ADS

- **Model diverse fusion patterns**, supporting multiple chain types and task triggering options, unmet by existing models.
- **Enable quantitative comparison** to evaluate the impact of different fusion patterns on real-time performance metrics
- **Optimize multiple real-time metrics** on multi-core platforms with an **ILP-based DAG scheduler**

Future Work:

- Improve IDS scalability for highly complex DAGs using heuristics such as simulated annealing and learning-based methods

Thank you

for your attention

References

[Xu22] Xu, C., Xu, Q., Wang, J., Wu, K., Lu, K., Qiao, C., “Aoi-centric task scheduling for autonomous driving systems”, in IEEE INFOCOM Conference on Computer Communications. pp. 1019–1028, 2022.

[Kuhse24] Kuhse, D., Holscher, N., Gunzel, M., Teper, H., Von Der Bruggen, G., Chen, J.J.,Lin, C.C., “Sync or sink? the robustness of sensor fusion against temporal misalignment”, in IEEE 30th Real-Time and Embedded Technology and Applications Symposium (RTAS),. pp. 122–134, 2024.

[Li24] Li, R., Jiang, X., Dong, Z., Wu, J.M., Xue, C.J., Guan, N., “Worst-case latency analysis of message synchronization in ros”, in IEEE Real-Time Systems Symposium (RTSS), pp. 185–197, 2023.

[Sun23] Sun, J., Wang, T., Li, Y., Guan, N., Guo, Z., Tan, G., “Seam: An optimal message synchronizer in ros with well-bounded time disparity”, in IEEE Real-Time Systems Symposium (RTSS), pp. 172–184, 2023.

[Teper22] Teper, H., Günzel, M., Ueter, N., von der Brüggen, G., Chen, J.J., “End-to-end timing analysis in ros2”, in IEEE Real-Time Systems Symposium (RTSS), pp. 53–65, 2022.

[Toba24] Toba, H., Azumi, T. “Deadline miss early detection method for dag tasks considering variable execution time,”, in 36th Euro-micro Conference on Real-Time Systems (ECRTS), 2024.

[Sun25] Sun, J., Li, X., Gong, M., Guan, N., Guo, Z., Chen, M., Zhao, J., Deng, Q., “Jointly ensuring timing disparity and end-to-end latency constraints in hybrid dag”, in IEEE 31st Real-Time and Embedded Technology and Applications Symposium (RTAS), pp. 190–201, 2025.

[Durr19] Dürr, M., Brüggen, G.V.D., Chen, K.H., Chen, J.J., “End-to-end timing analysis of sporadic cause-effect chains in distributed systems”, in ACM Transactions on Embedded Computing Systems (TECS), pp. 1–24, 2019.

[Jiang23] Jiang, X., Luo, X., Guan, N., Dong, Z., Liu, S., Yi, W., “Analysis and optimization of worst-case time disparity in cause-effect chains”, in Design, Automation &Test in Europe Conference & Exhibition (DATE), pp. 1–6, 2023.

References

[Gunzel23] Günzel, M., Teper, H., Chen, K.H., von der Brüggen, G., Chen, J.J., "On the equivalence of maximum reaction time and maximum data age for cause-effect chains", in 35th Euro-micro Conference on Real-Time Systems (ECRTS), 2023.

[Teper24] Teper, H., Betz, T., Günzel, M., Ebner, D., Von Der Brüggen, G., Betz, J., Chen,J.J., "End-to-end timing analysis and optimization of multi-executor ros 2 systems", in IEEE 30th Real-Time and Embedded Technology and Applications Symposium (RTAS), pp. 212–224, 2024.

[Becker16] Becker, M., Dasari, D., Mubeen, S., Behnam, M., Nolte, T., "Synthesizing job-level dependencies for automotive multi-rate effect chains", in IEEE 22nd International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA), pp. 159–169, 2016.

[Becker17] Becker, M., Mubeen, S., Dasari, D., Behnam, M., Nolte, T., "A generic framework facilitating early analysis of data propagation delays in multi-rate systems", in IEEE 23rd International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA), pp. 1–11, 2017.

[Saidi17] Saidi, S.E., Pernet, N., Sorel, Y., "Automatic parallelization of multi-rate fmi-based co-simulation on multi-core", in TMS/DEVS Symposium on Theory of Modeling and Simulation, 2017.

[Verucchi20] Verucchi, M., Theile, M., Caccamo, M., Bertogna, M., "Latency-aware generation of single-rate dags from multi-rate task sets", in IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS), pp. 226–238, 2020.

[Tang22] Tang, Y., Jiang, X., Guan, N., Ji, D., Luo, X., Yi, W., "Comparing communication paradigms in cause-effect chains", in IEEE Transactions on Computers, vo.72, vol.1, pp. 82–96, 2022..

[Maia24] Maia, L., Fohler, G., "Reducing end-to-end latencies of multi-rate cause-effect chains in safety critical embedded systems" in 12th European Congress on Embedded Real Time Software and Systems (ERTS), 2024.

[Tang23R] Tang, Y., Guan, N., Jiang, X., Dong, Z., Yi, W., "Reaction time analysis of event-triggered processing chains with data refreshing", in 60th ACM/IEEE Design Automation Conference (DAC), pp. 1–6, 2023.

References

[Tang23O] Tang, Y., Jiang, X., Guan, N., Liu, S., Luo, X., Yi, W., “Optimizing end-to-end latency of sporadic cause-effect chains using priority inheritance”, in IEEE Real-Time Systems Symposium (RTSS), pp. 411–422, 2023.

[Yano24] Yano, A., Azumi, T., “Work-in-progress: Multi-deadline dag scheduling model for autonomous driving systems”, in IEEE Real-Time Systems Symposium (RTSS), pp. 451–454, 2024.