Data Propagation Delay Constraints in Multi-Rate Systems
Deadlines vs. Job-Level Dependencies

26th International Conference on Real-Time Networks and Systems
October 10, 2018

Tobias Klaus, Florian Franzmann, Matthias Becker (KTH), **Peter Ulbrich**

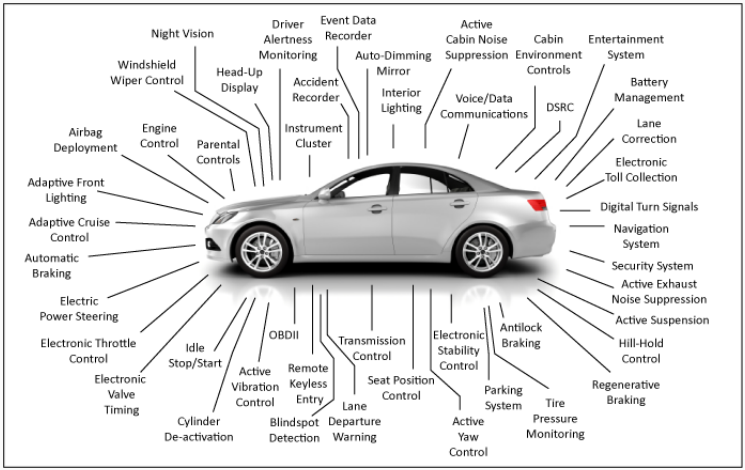
Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU)
KTH Royal Institute of Technology

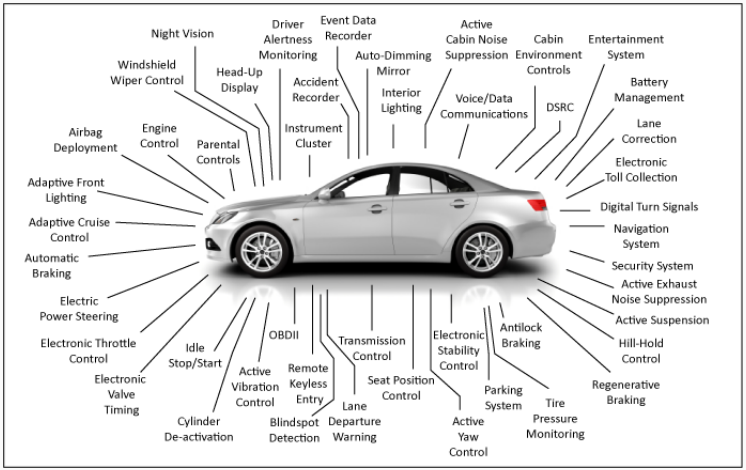
DFG SCHR 603/9-2
SFB/TR 89 C1

EU EFRE funds
0704/883 25

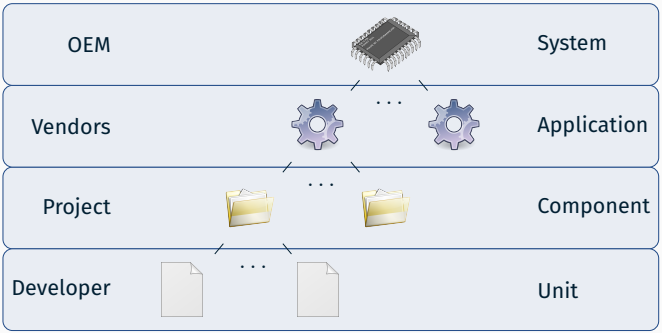
KTH VETENSKAP OCH KONST

FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG
FACULTY OF ENGINEERING


Night Vision
Driver Alertness Monitoring
Event Data Recorder
Active Cabin Noise Suppression
Cabin Environment Controls
Entertainment System
Windshield Wiper Control
Head-Up Display
Auto-Dimming Mirror
Accident Recorder
Interior Lighting
Voice/Data Communications
DSRC
Battery Management
Airbag Deployment
Engine Control
Parental Controls
Instrument Cluster
Lane Correction
Electronic Toll Collection
Adaptive Front Lighting
Digital Turn Signals
Adaptive Cruise Control
Navigation System
Automatic Braking
Security System
Active Exhaust Noise Suppression
Electric Power Steering
Active Suspension
Electronic Throttle Control
Idle Stop/Start
OBDII
Transmission Control
Electronic Stability Control
Antilock Braking
Hill-Hold Control
Electronic Valve Timing
Active Vibration Control
Remote Keyless Entry
Seat Position Control
Parking System
Regenerative Braking
Cylinder De-activation
Blindspot Detection
Lane Departure Warning
Active Yaw Control
Tire Pressure Monitoring

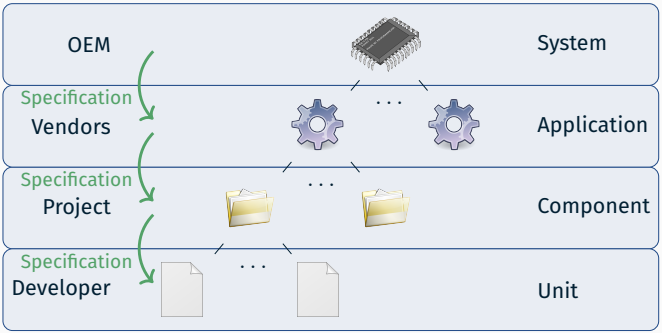


→ **Temporal verification along the workflow?**



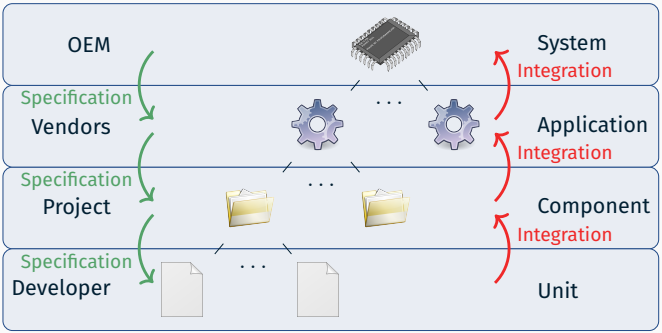
Distributed Concurrent Engineering Paradigm

- Temporal aspects: Focus on (isolated) tasks



Distributed Concurrent Engineering Paradigm

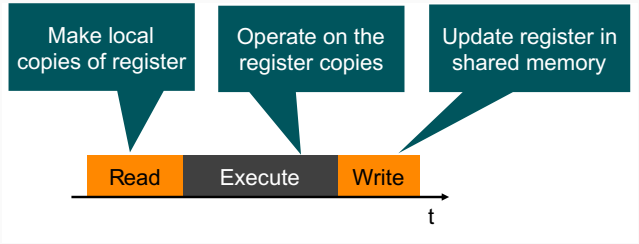
- Temporal aspects: Focus on (isolated) tasks
- Top-down specification (periods, budgets, **deadlines**)



Distributed Concurrent Engineering Paradigm

- Temporal aspects: Focus on (isolated) tasks
- Top-down specification (periods, budgets, **deadlines**)
- Bottom-up integration and verification

Implicit Communication

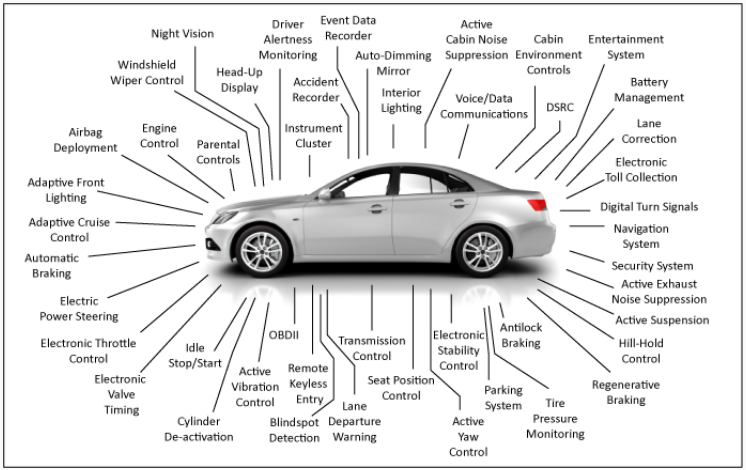


Implicit communication between tasks

- Tasks are independently triggered
- Three phases of execution (read, execute, write)
- Predominant in automotive applications

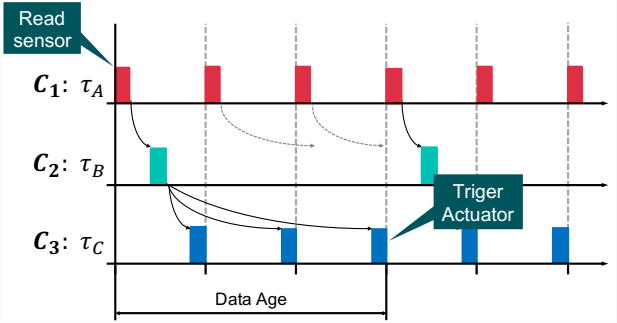
→ **Last is best semantics** (no synchronization)

Automotive Control Systems



→ **Are deadlines the right abstraction?**

Data Propagation Delays



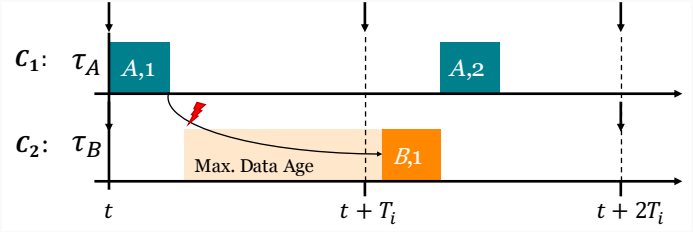
Propagation of data in control-systems

- Independent tasks with different periods and implicit communication?

$\rightarrow$ **Complex under- and oversampling situations**

Cause effect Chains $\zeta_i \rightarrow$ Sensor to actuator paths (DAG)

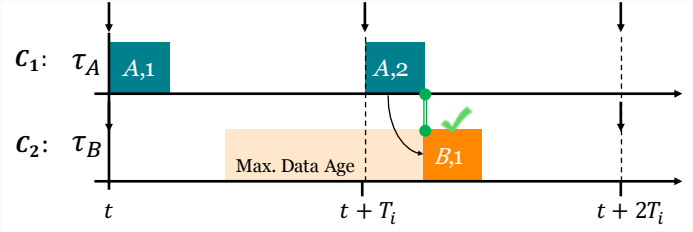
Data age $\rightarrow$ Time between sampling and actuation in ζ_i



Data propagation delay constraints

- Ensure a certain quality of control

→ **Maximum data age constraint**



Data propagation delay constraints

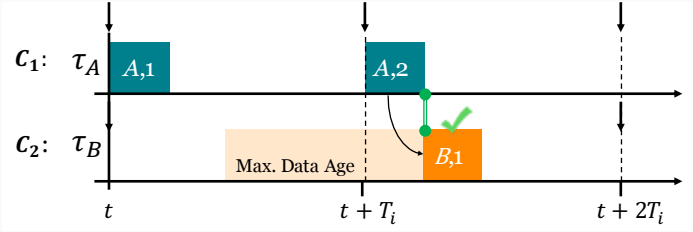
- Ensure a certain quality of control

→ **Maximum data age constraint**

What is a job-level dependency?

- Constraints execution order: $\tau_i \xrightarrow{(k,l)} \tau_j$ wrt. $lcm(\tau_i, \tau_j)$

→ Scheduling-agnostic; guarantees delay constraints (sound)



Data propagation delay constraints

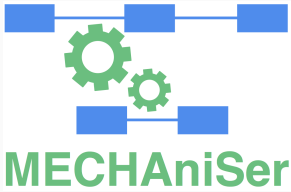
- Ensure a certain quality of control

→ **Maximum data age constraint**

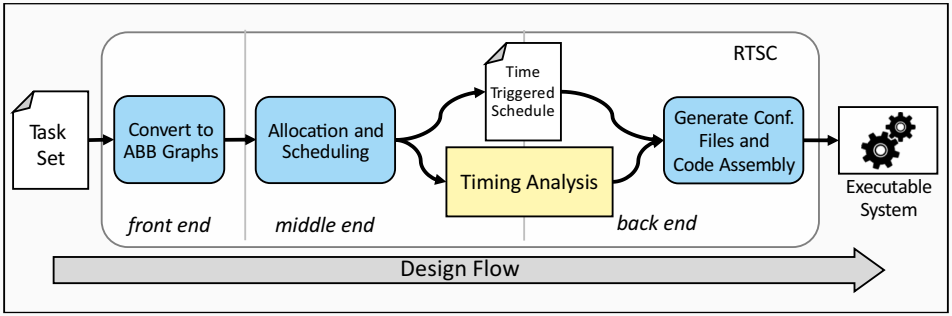
What is a job-level dependency?

- Constraints execution order: $\tau_i \xrightarrow{(k,l)} \tau_j$ wrt. $lcm(\tau_i, \tau_j)$

→ Scheduling-agnostic; guarantees delay constraints (sound)

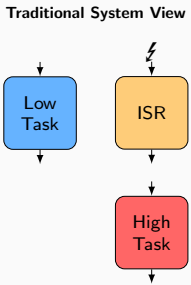


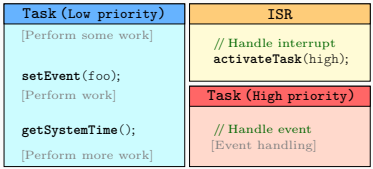
The Real-Time Systems Compiler (RTSC)



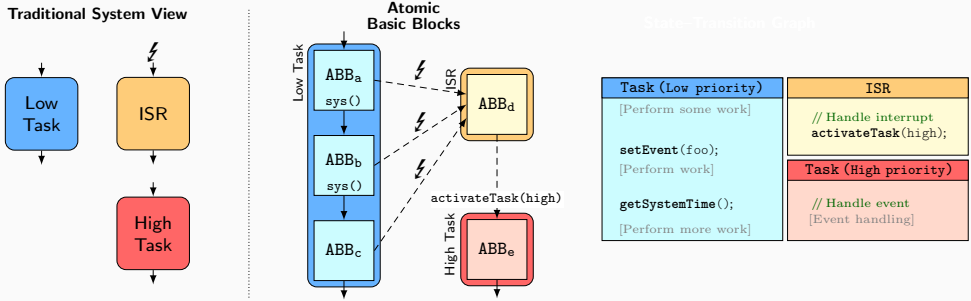
From Architectural to System Analysis

- LLVM-based real-time system analysis and transformation tool
- RTOS- and platform-agnostic intermediate representation
- Testbed for system transformation, scheduling and platforms





Fine-grained (job-level) decomposition of existing systems

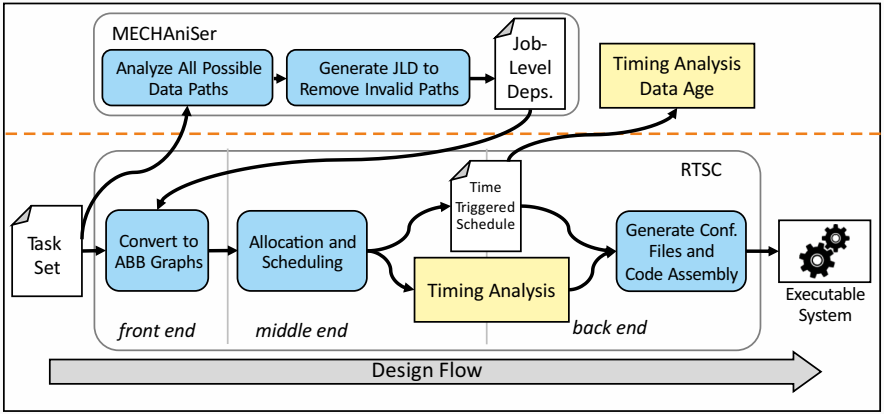


Fine-grained (job-level) decomposition of existing systems

Atomic Basic Blocks (ABB)

- Atomic from scheduler perspective
- Single-entry single-exit region

Extending the RTSC



Timing Analysis and JLD-Aware Scheduling

- ABB-graph transformation to incorporate job-level dependencies
- Data-age analysis to identify maximal data-age paths in static schedules

Random System Generator

- 433 randomly generated systems (task sets)
  - 59 - 1000 Jobs (average: 458)
  - 1 - 3 cause effect chains
  - 0.6 - 2.0 utilization (average: 1.218)
- Variants for allocation + scheduling (8 algorithms)
- Testbed: LitmusRT with 1-4 cores

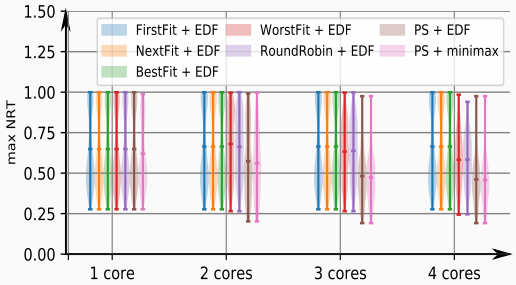
Random System Generator

- 433 randomly generated systems (task sets)
  - 59 - 1000 Jobs (average: 458)
  - 1 - 3 cause effect chains
  - 0.6 - 2.0 utilization (average: 1.218)
- Variants for allocation + scheduling (8 algorithms)
- Testbed: LitmusRT with 1-4 cores

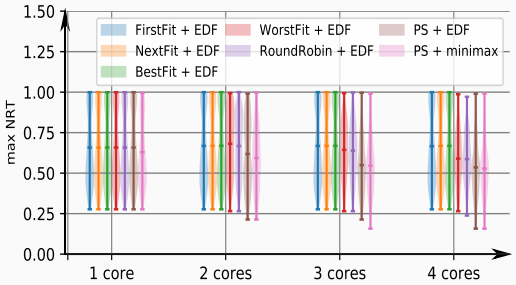
Subject of Evaluation

1. Impact of job-level dependencies on traditional schedulability parameters
2. Impact of allocation and scheduling on data age
3. Impact of job-level dependencies on data age distribution
4. Impact of number of CPU cores on data age

Impact of JLDs on Traditional Real-Time Parameter



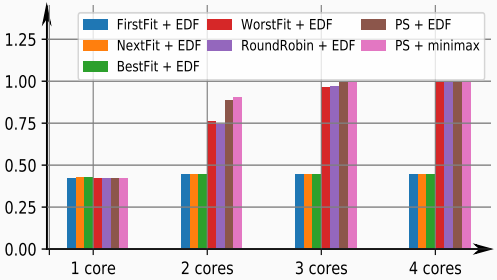
without JLDs



with JLDs

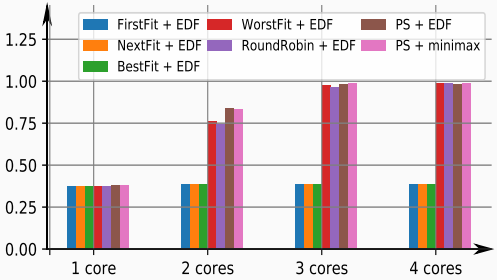
- Maximum Normalized Response Time
  - Light increase

Impact of JLDs on Traditional Real-Time Parameter



without JLDs

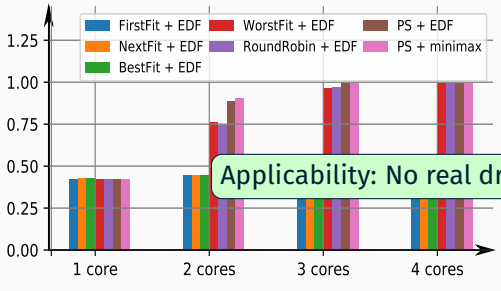
- Maximum Normalized Response Time
  - Light increase



with JLDs

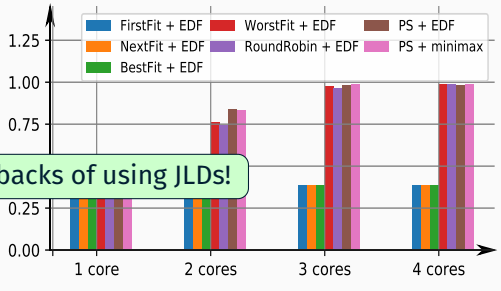
- Task Local Schedulability
  - Worst Reduction: 5%

Impact of JLDs on Traditional Real-Time Parameter



without JLDs

- Maximum Normalized Response Time
  - Light increase

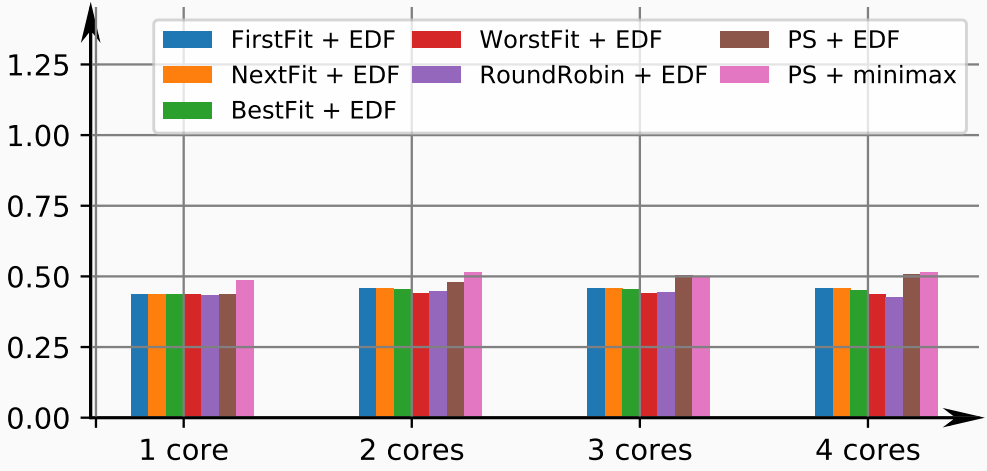


with JLDs

- Task Local Schedulability
  - Worst Reduction: 5%

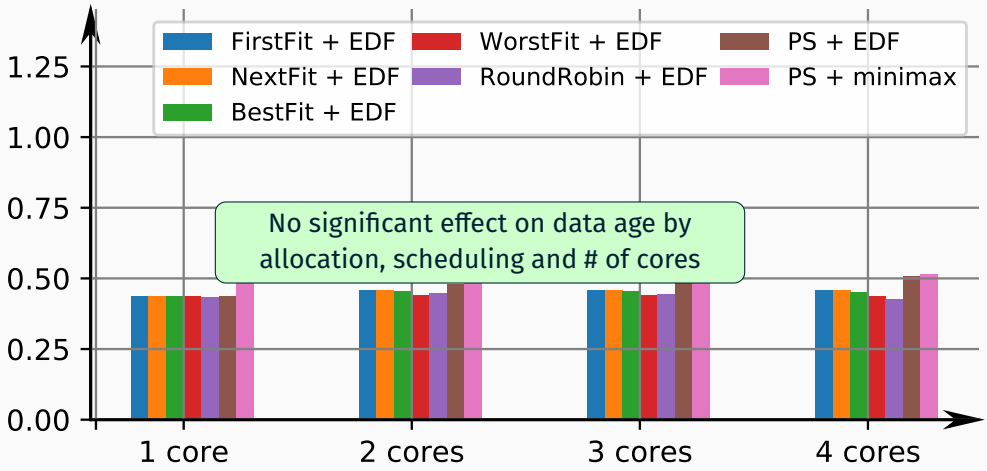
Applicability: No real drawbacks of using JLDs!

Impact of JLDs on Data-Age Schedulability



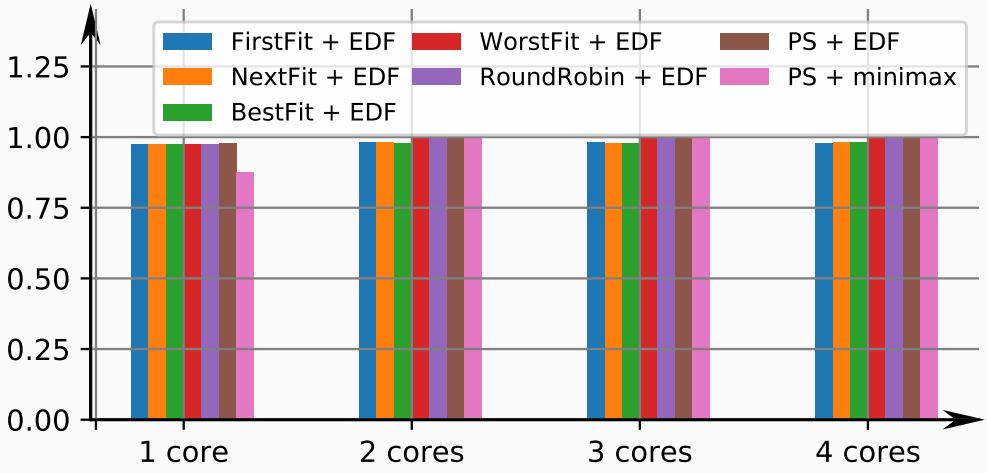
data-age schedulable systems without JLDs

Impact of JLDs on Data-Age Schedulability



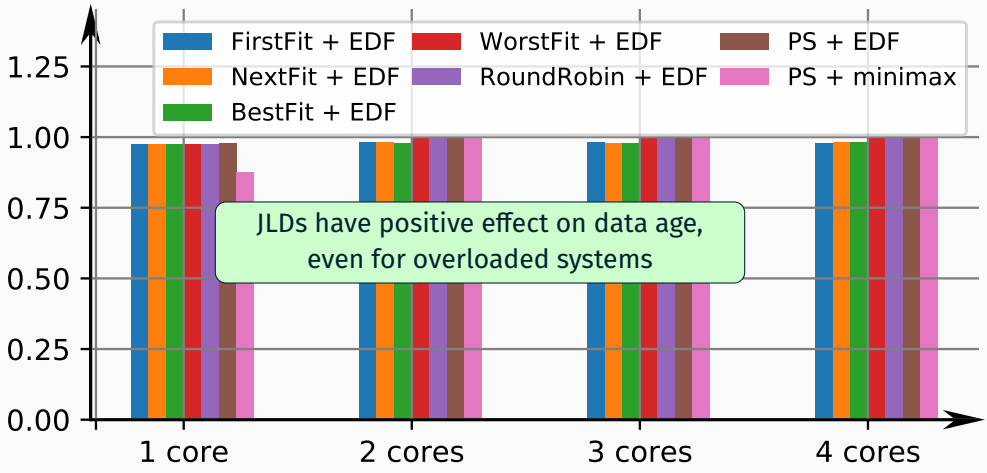
data-age schedulable systems without JLDs

Impact of JLDs on Data-Age Schedulability



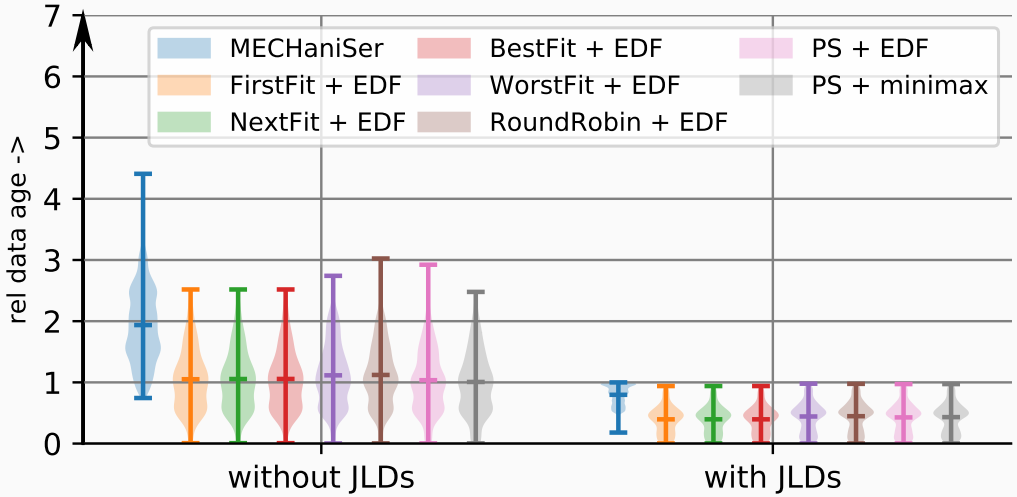
data-age schedulable systems with JLDs

Impact of JLDs on Data-Age Schedulability

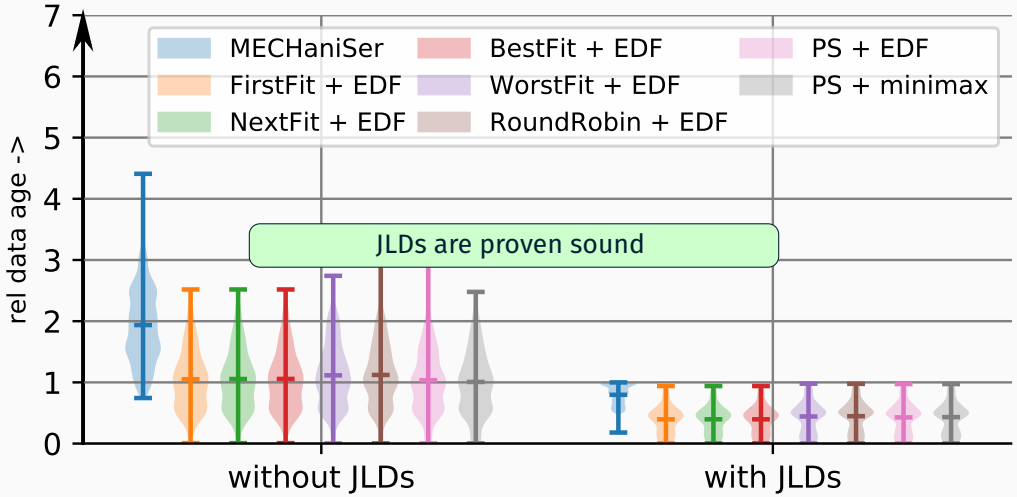


data-age schedulable systems with JLDs

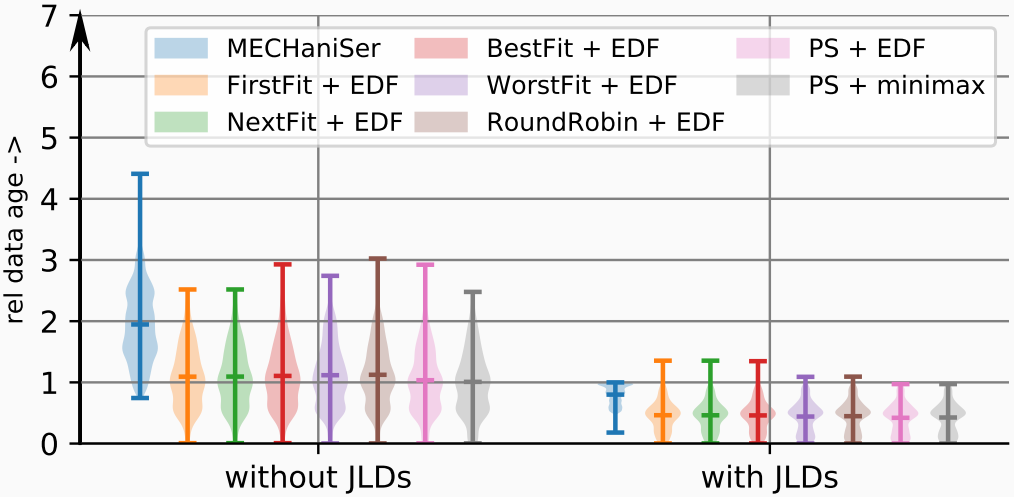
Impact of JLDs on Data-Age Distribution



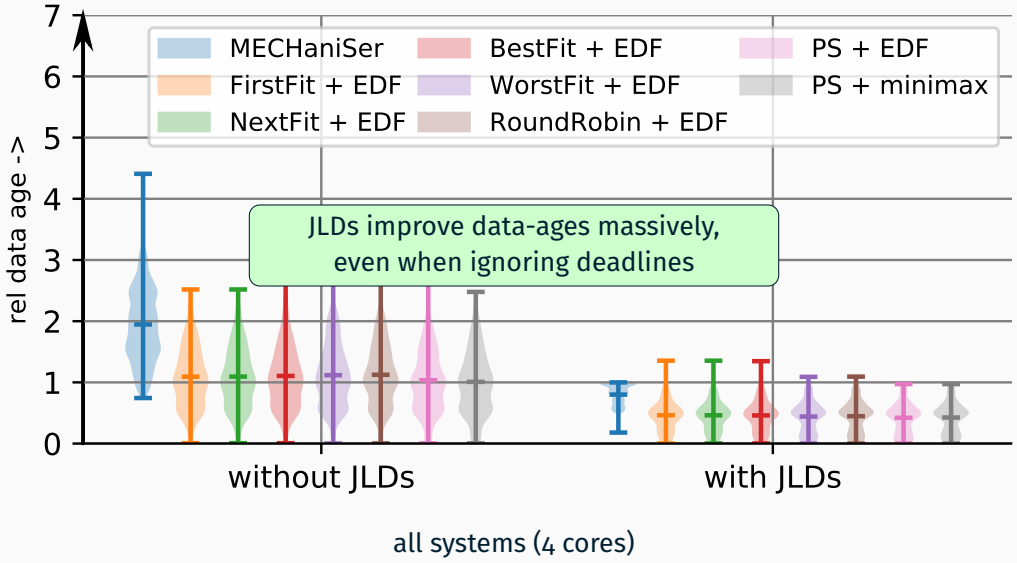
Impact of JLDs on Data-Age Distribution



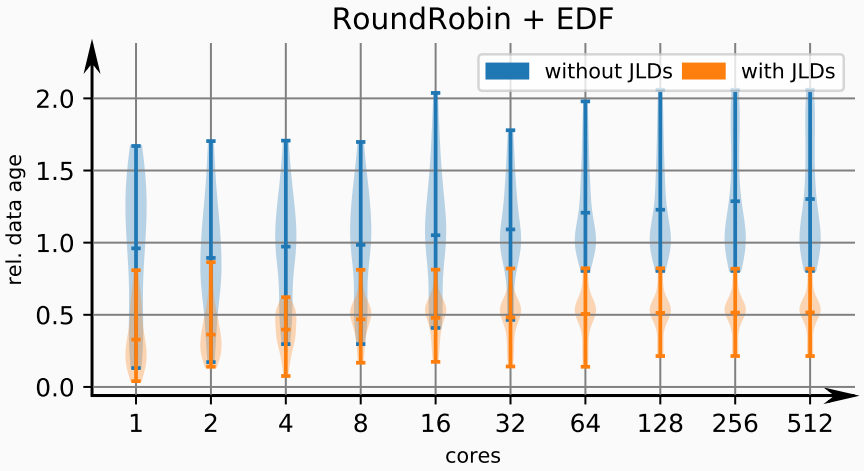
Impact of JLDs on Data-Age Distribution



Impact of JLDs on Data-Age Distribution



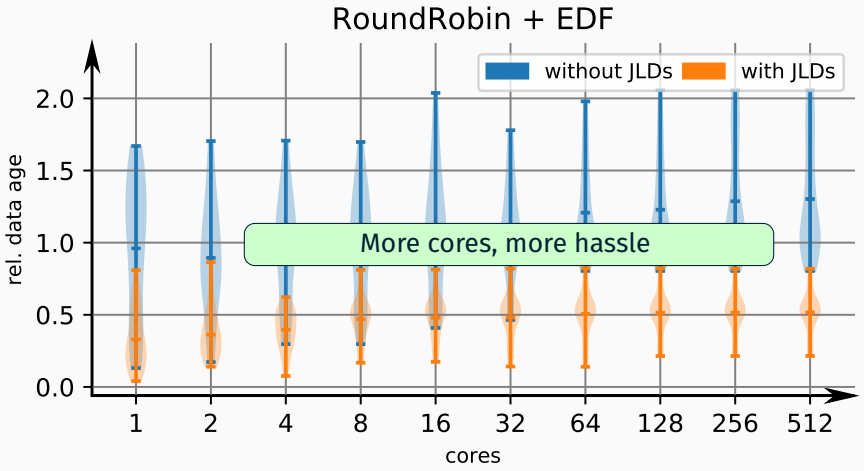
Impact of Additional Cores on Data Age



allocation + edf scheduling

round robin

Impact of Additional Cores on Data Age



allocation + edf scheduling

round robin

Conclusion

Take Aways

- Job-level dependencies are a practical approach
- Superior on meeting data-age constraints than:
  - Processing power (multicore)
  - Scheduling algorithms or
  - Task-local deadlines

Future Work

- Dynamically enforce dependencies on event-triggered systems
- Assess run-time overhead of dependencies
- Compare different load situations between time-triggered and event-triggered systems