Machine ballets don't need conductors

Towards scheduling-based service choreographies in a real-time SOA for industrial automation

Thomas Kothmayr, Alfons Kemper
Technische Universität München
{kothmayr, kemper}@in.tum.de

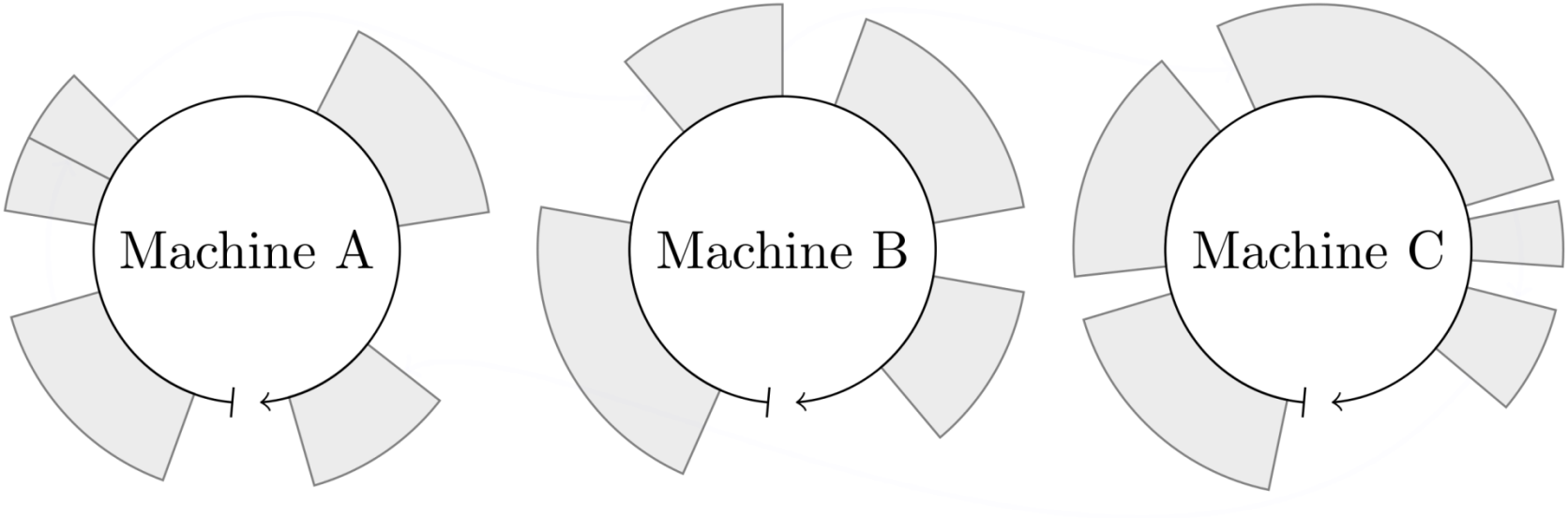
Andreas Scholz, Jörg Heuer
Corporate Technology, Siemens AG
{andreas.as.scholz, joerg.heuer}@siemens.com

Motivation

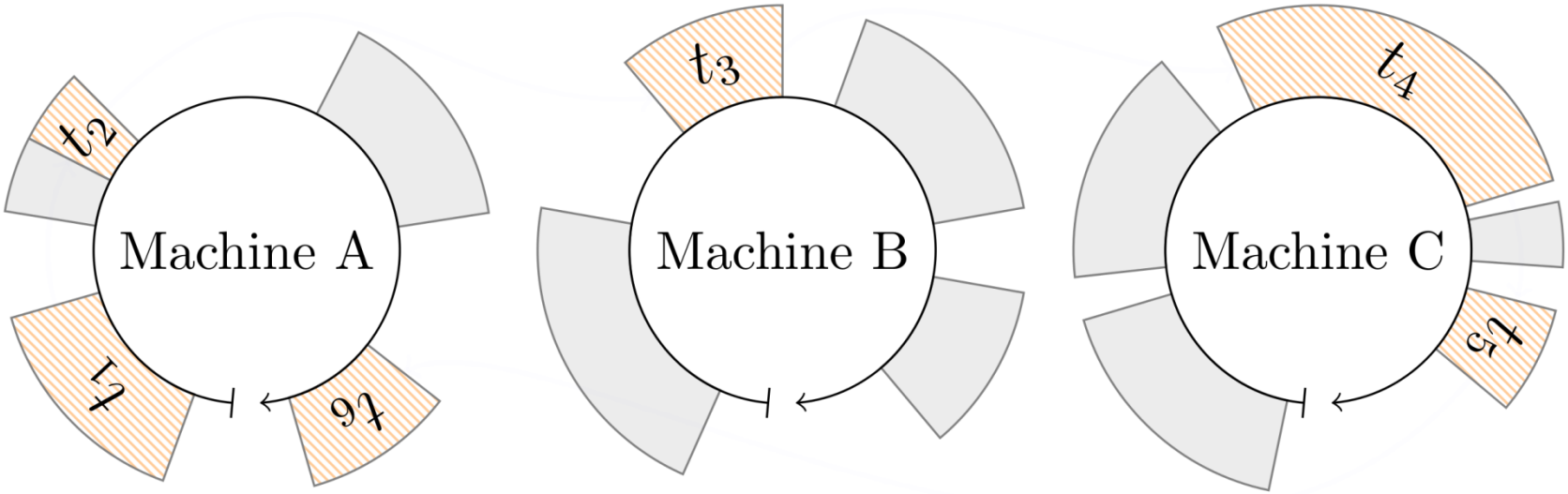
- Monolithic, scan-based control systems are inflexible
- Hardware side of automation is already changing
  - Industrial Ethernet is gaining traction
  - Smart embedded devices are performing more and more tasks
- SOA as the emerging design paradigm down to the hard real-time control cycle

→ Goal: Global cooperation without central coordination

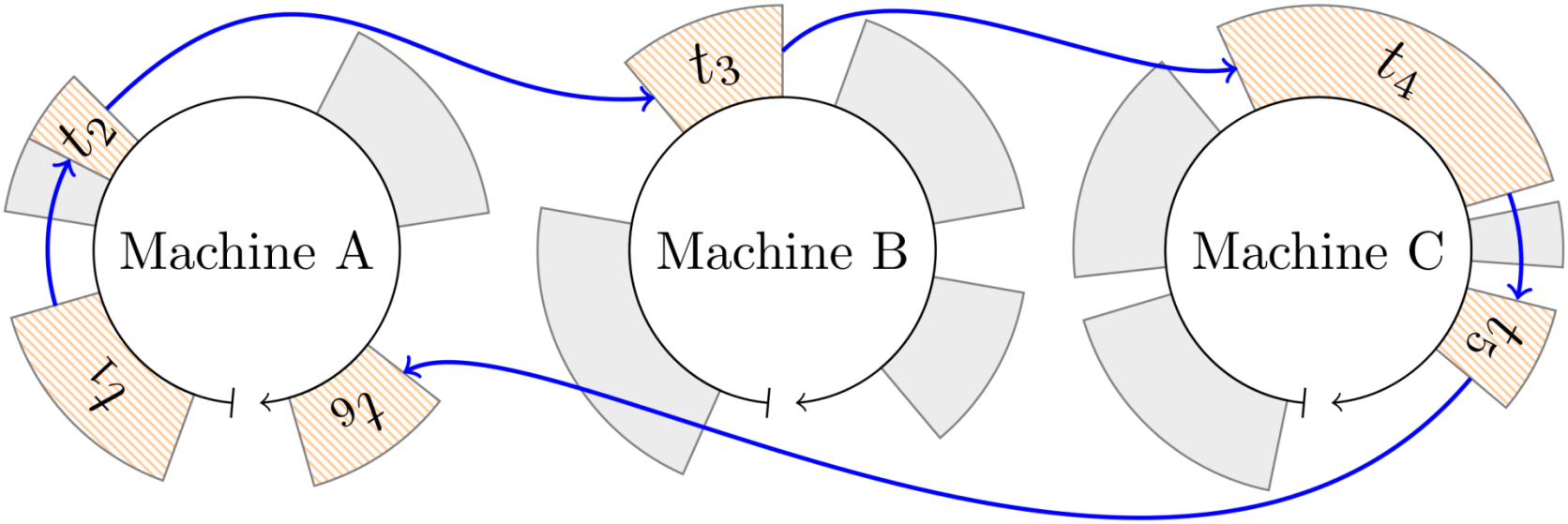
A distributed workflow example



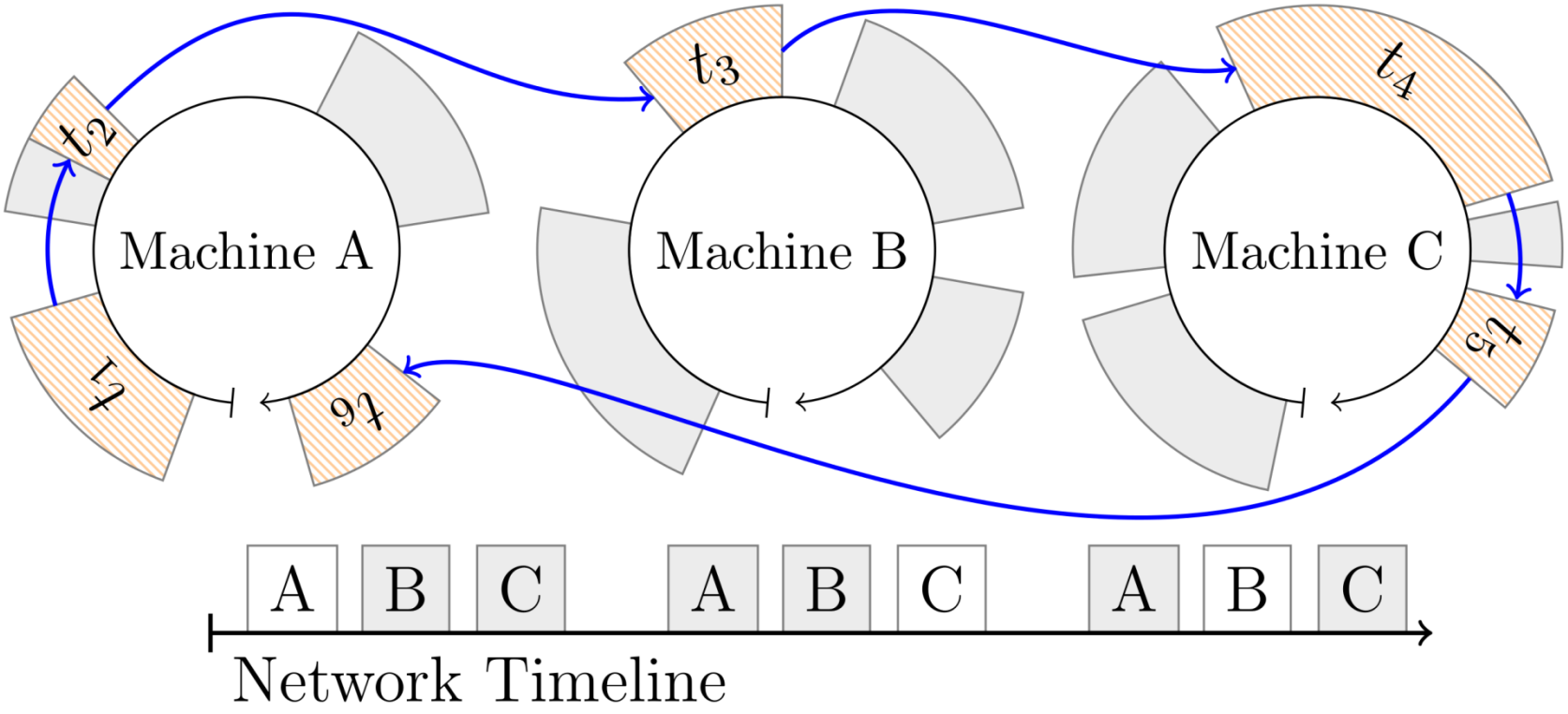
A distributed workflow example



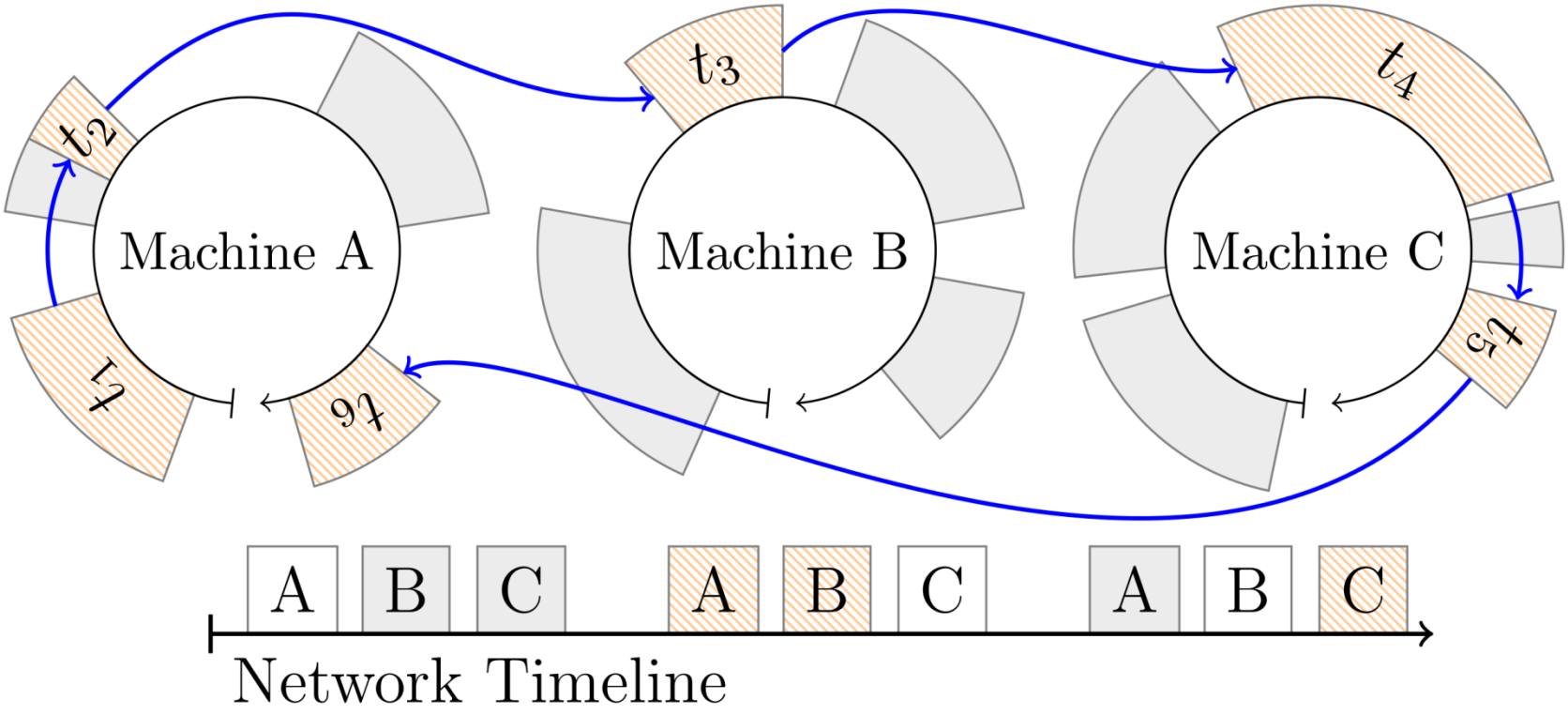
A distributed workflow example



A distributed workflow example

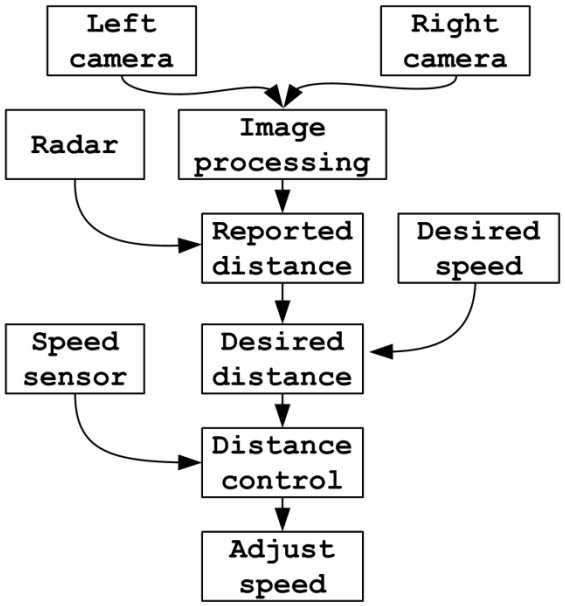


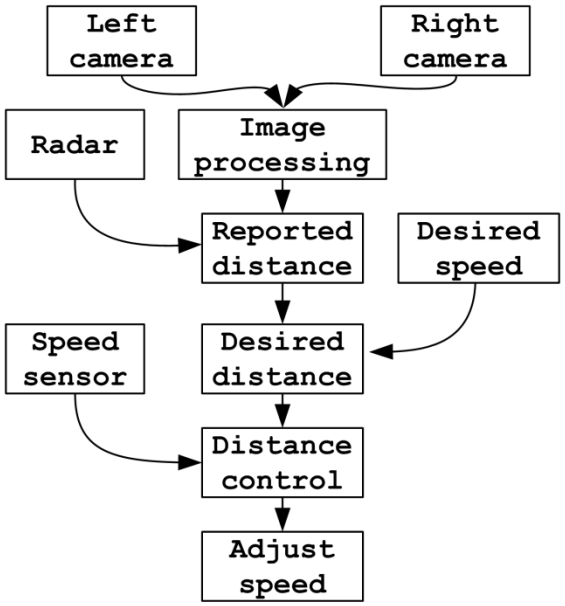
A distributed workflow example

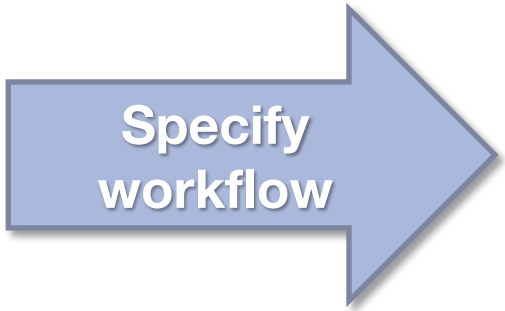


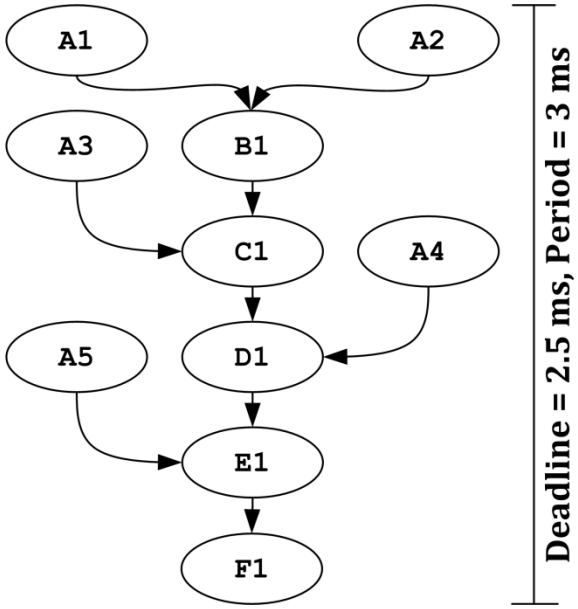
Goals and target environment

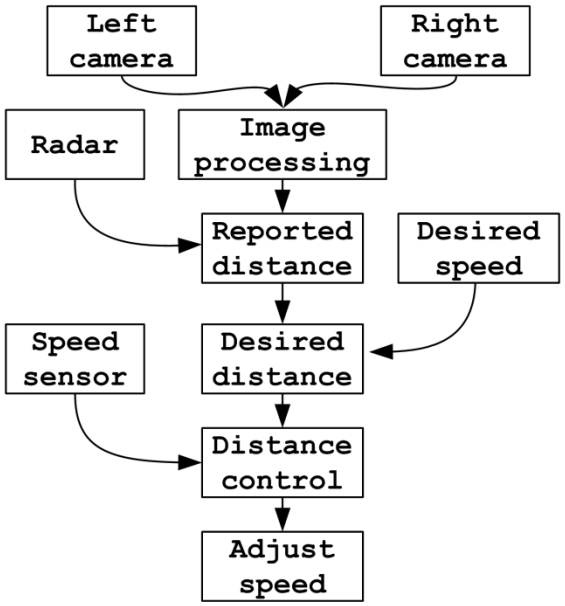
- Our target environment
  - Distributed hard real-time systems
  - Automation task modelled as workflow of communicating jobs
    - Workflow constraints: Global deadline, workflow period
  - Devices connected through TDMA network
  - Physical and network properties are inputs to the planner, i.e. network configuration is fixed an not an output parameter
- Our goal
  - Synthesize a cyclic, non-preemptive schedule for each processor in the system
  - Employ heuristics to enable interactive planning and development of large systems


Left camera
Right camera
Radar
Image processing
Reported distance
Desired speed
Speed sensor
Desired distance
Distance control
Adjust speed

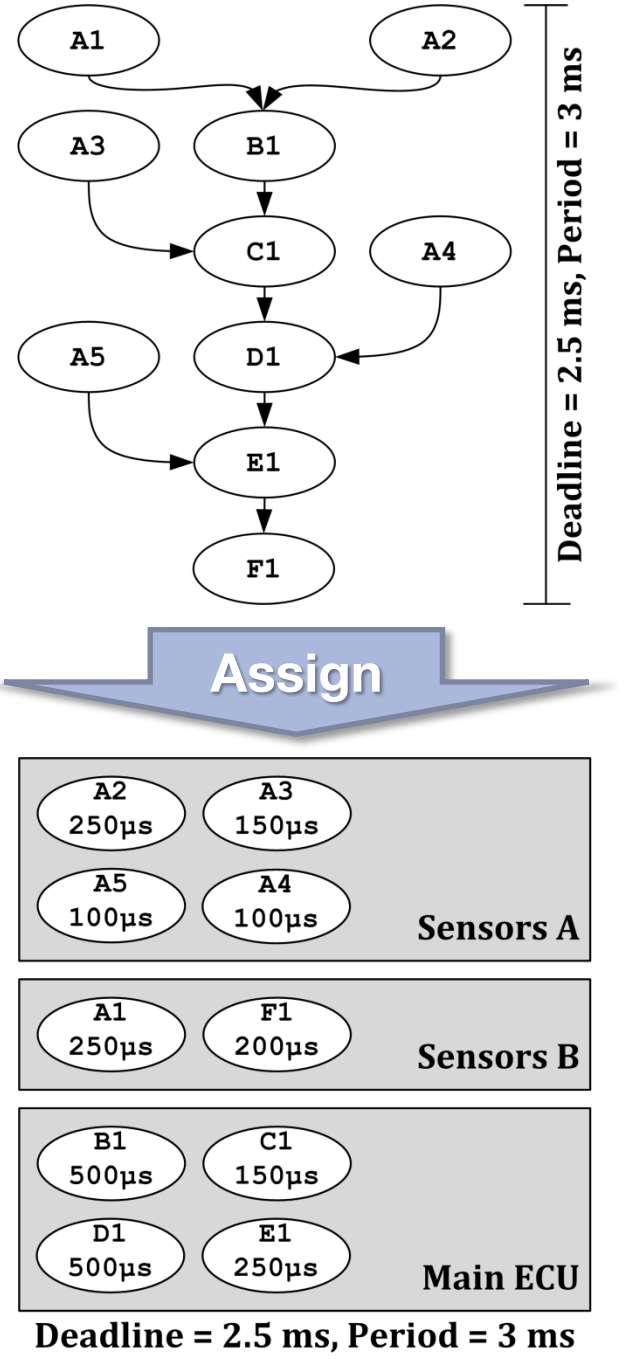

Left camera
Right camera
Radar
Image processing
Reported distance
Desired speed
Speed sensor
Desired distance
Distance control
Adjust speed

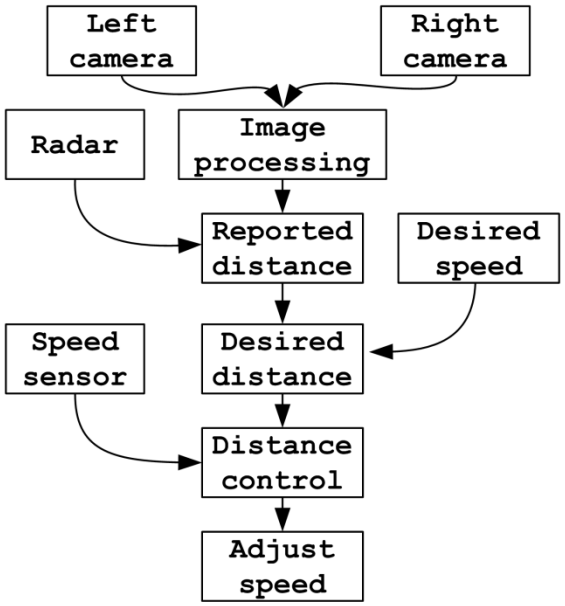

Specify workflow


A1
A2
A3
B1
C1
A4
A5
D1
E1
F1
Deadline = 2.5 ms, Period = 3 ms

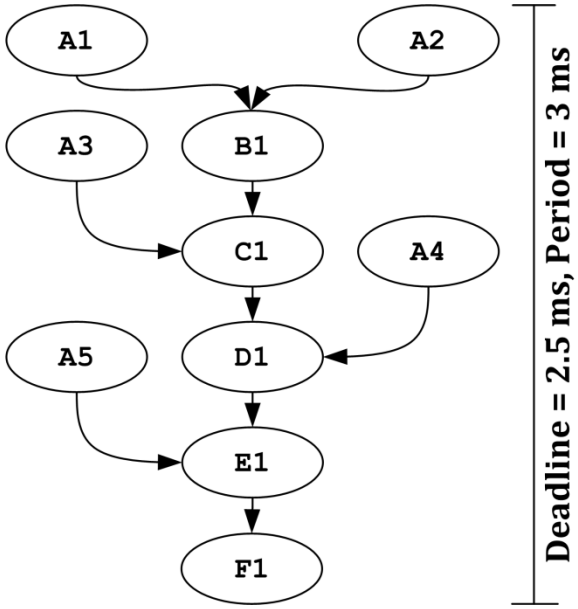


Specify workflow

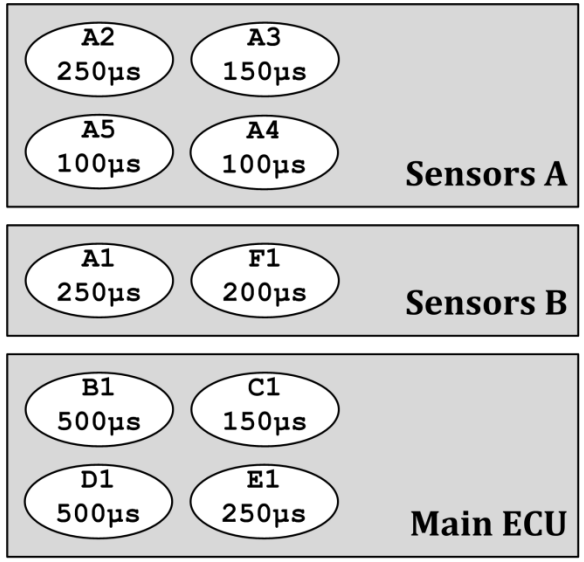




Specify workflow

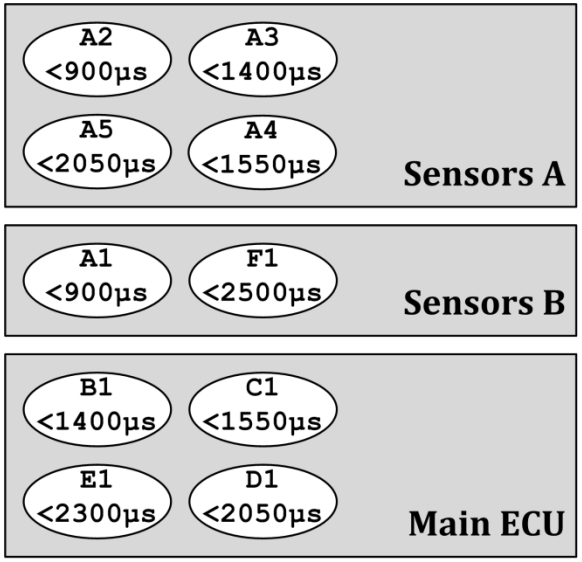


Assign

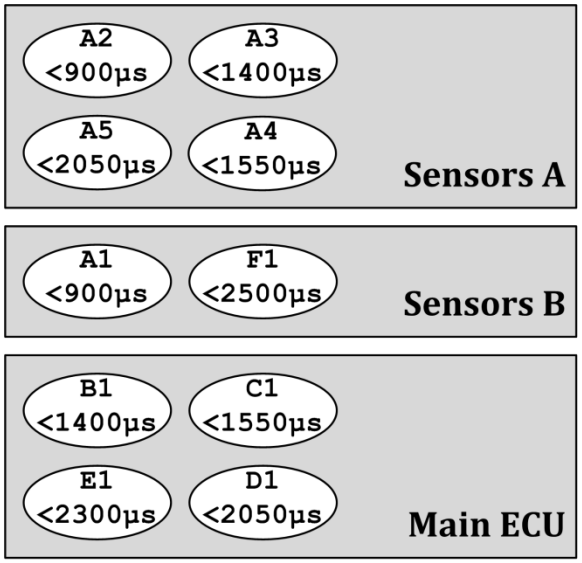


Deadline = 2.5 ms, Period = 3 ms

Assign local constraints

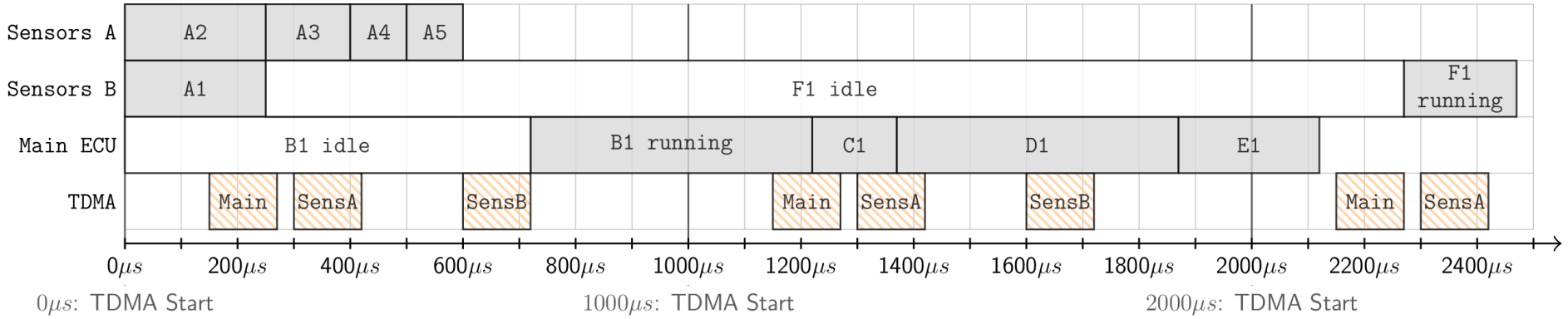


Deadline = 2.5 ms, Period = 3 ms



Deadline = 2.5 ms, Period = 3 ms

Schedule



$0\mu s$: TDMA Start $1000\mu s$: TDMA Start $2000\mu s$: TDMA Start

Heuristics for Scheduling

Scheduling Heuristics

- Earliest Release time First (ERF)
  - Map precedence constraints to release times:
    $\forall j_i, j_k \epsilon \mathcal{J} \cup \Omega : j_i \prec j_k \implies r'_k = max\{r_k, r_i + wcet_i\}$
  - Sort tasks according to their release time

- Earliest Deadline First (EDF)
  - Map precedence constraints deadlines:
    $\forall j_i, j_k \epsilon \mathcal{J} \cup \Omega : j_i \prec j_k \implies d'_i = min\{d_i, d_k - wcet_k\}$
  - Schedule task with earliest release time if it does not violate earliest deadline

Scheduling Heuristics

- BLOCK heuristic
  - Setup schedule through ERF
  - Divide schedule into consecutive blocks of jobs
  - If invalid, move jobs with higher deadline towards end of block

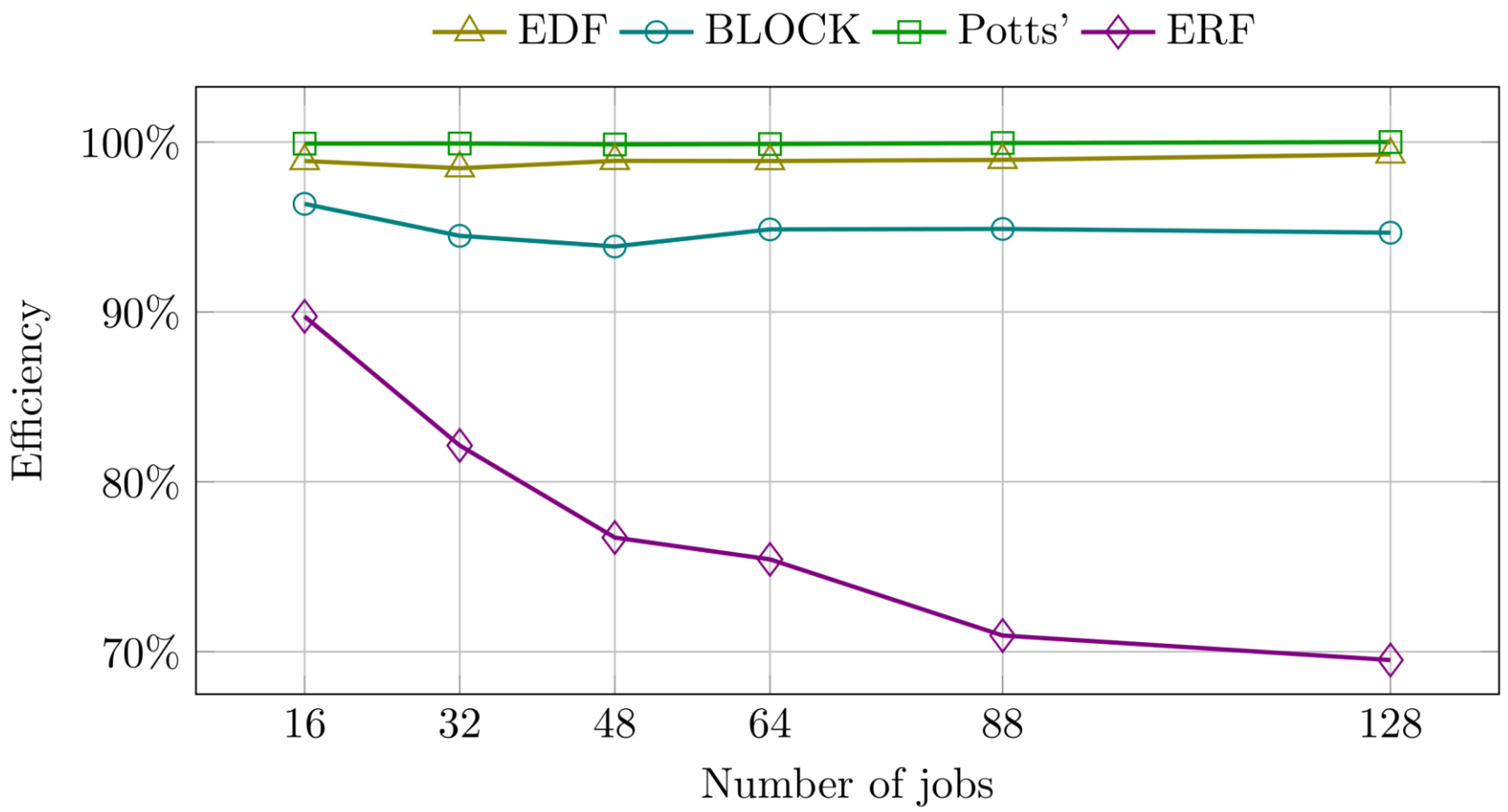
- Pott‘s algorithm
  - Setup by sorting jobs by deadline in topological order
  - If invalid, analyze critical sequence and identify critical job (missing its deadline) and interference job (before critical job)
  - Move interference job behind critical job

Evaluation of scheduling heuristics

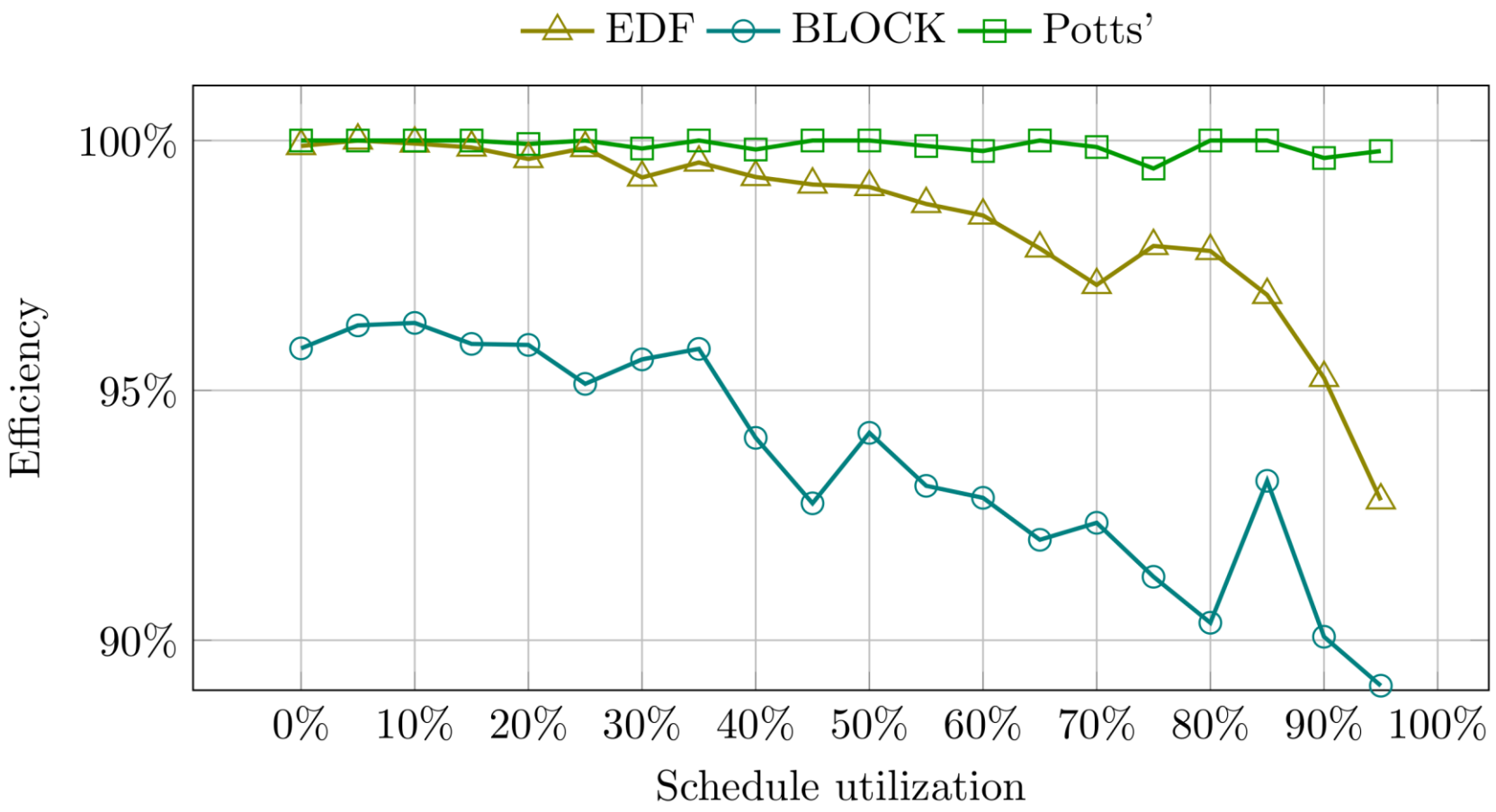
- 20 500 feasible random in-tree test cases
  - Verified through linear program solver
  - Test case size between 16 and 128 jobs
  - Jobs have to be scheduled before a global deadline

- Evaluation for efficiency
  - Which percentage of all feasible test cases are solved?
  - Makespan, max lateness, etc. not relevant
  - All or nothing scoring: one late job invalidates the schedule

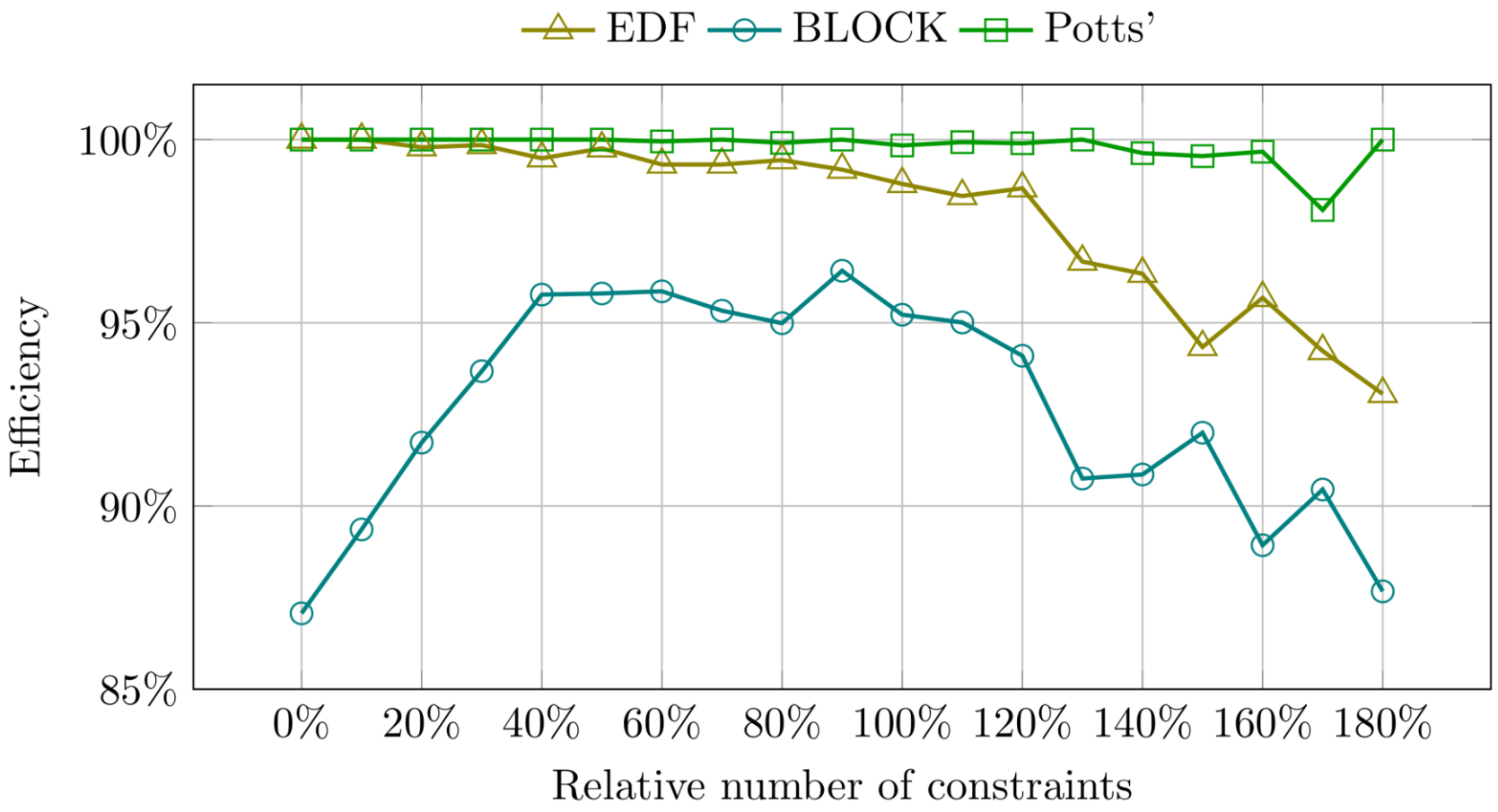
Heuristics: Efficiency



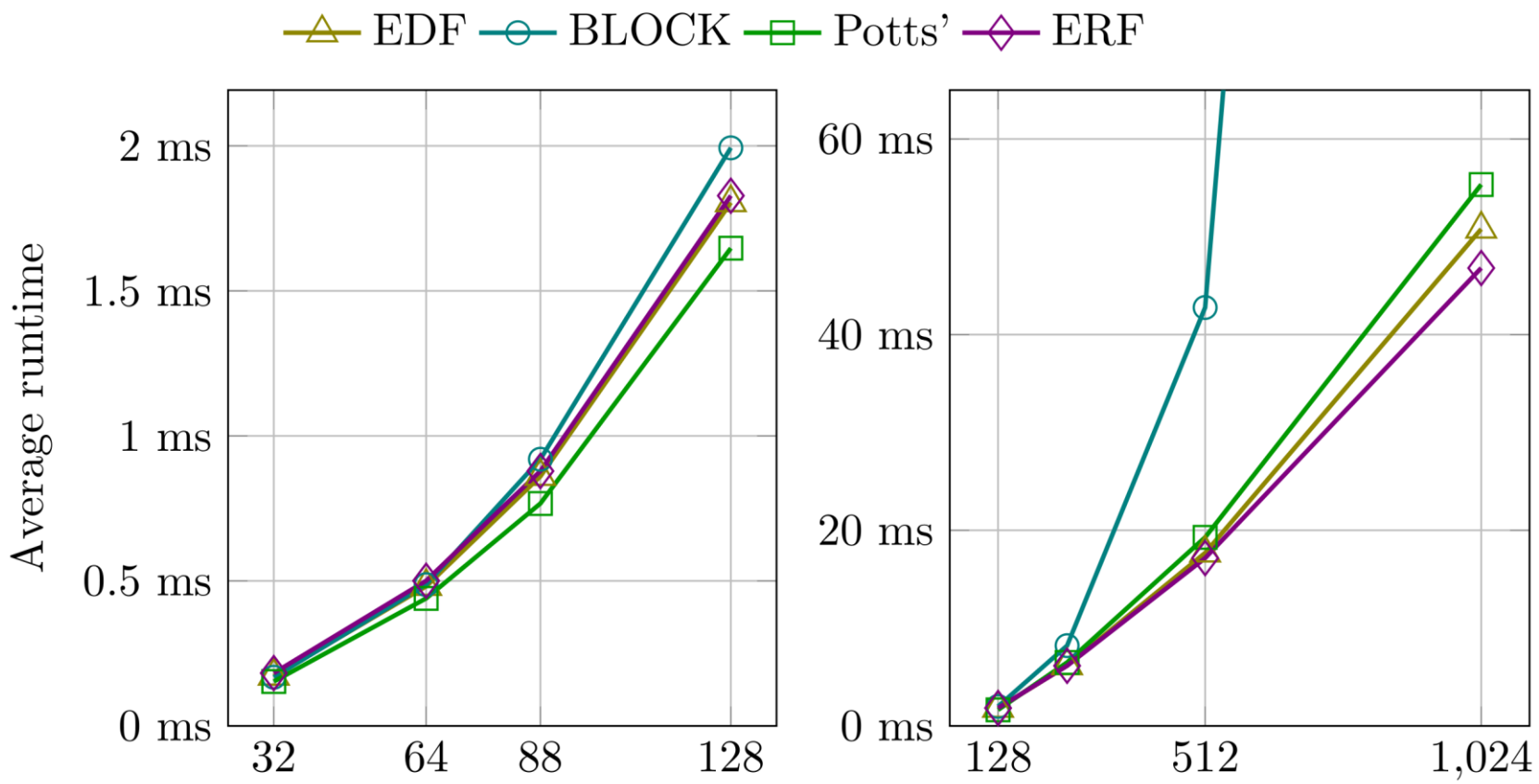
Heuristics: Efficiency vs. Utilization



Heuristics: Efficiency vs. Constraints



Heuristics: Runtime



Conclusions and future work

Three steps for schedule-based service choreographies:

1. Task assignment
2. Deriving local constraints
3. Scheduling

Conclusions and future work

Three steps for schedule-based service choreographies:

1. Task assignment → Performed by skilled engineer ✓
2. Deriving local constraints → Ongoing research ⊘
3. Scheduling → Existing heuristics are up to the task ✓

Thank you for your attention!