A Formal Approach towards Safe and Stable Schedule Synthesis in Weakly Hard Control Systems

Debarpita Banerjee, **Parasara Sridhar Duggirala**, **Bineet Ghosh**, **Sumana Ghosh**

1. Introduction and Motivation

Speed sensed

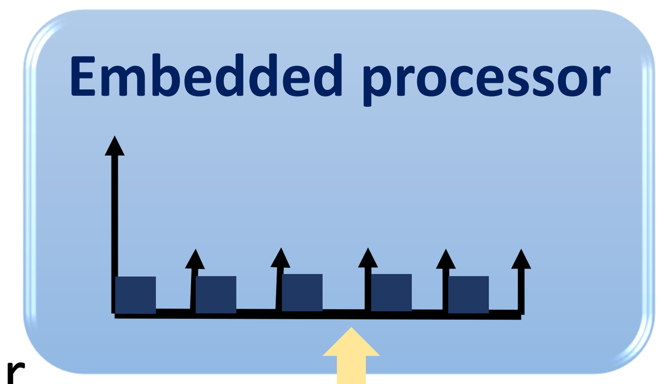
Adjusts speed

Feedback controller

Plant

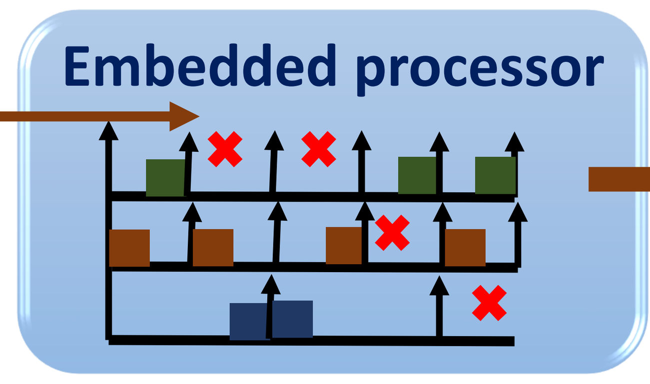
Execution of discrete-time controller

Plant-Control Closed-Loop System



Software Control Task

Deadline missed: weakly hard setting



Scheduling Multiple Weakly Hard Control Tasks

1. Is the schedule **safe**?

2. Is the schedule **stable**?

2. Limitations of State-of-the-Art

- Existing methods explore either stable schedule [1] or safe schedule[2, 3], but **not both aspects simultaneously**
- [2],[3]: Ensure safety **only over bounded time horizon**

---

1. A structured methodology for pattern based adaptive scheduling in embedded control, S. Ghosh et al., ACM-TECS 2017.
2. Statistical approach to efficient and deterministic schedule synthesis for cyber-physical systems, Shengjie Xu et al., ATVA 2023.
3. Safety-aware flexible schedule synthesis for cyber-physical systems using weakly-hard constraints, Shengjie Xu et al., ASP-DAC 2023.

3. Contributions of this Work

- **First time :** Addressing **(stability, safety, schedulability)**
- Synthesizing **schedule** that ensures **stability and safety** over **unbounded time horizon** with **minimized WCRT**

1. • Ensuring exponential **stability**
2. • Ensuring **safety** over infinite time horizon
3. • Synthesizing optimal safe and stable **schedule**

4. The Proposed Method

Step 1: Ensuring Stability

Settling time + Reference Values → **(l, ∊)- exponential stability criterion** → $\frac{\|x[k+l]\|}{\|x[k]\|} < \epsilon$

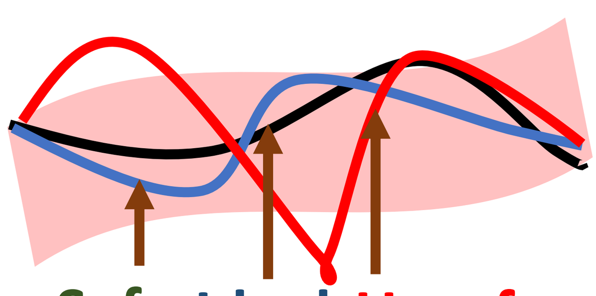
- Deducing one stable **(M, K)-firm constraint**
- Meet at least M out of K deadlines: satisfies (l, ∊)- criterion

Step 2: Ensuring Safety

All hits: 111….11…. → **ideal** behavior

Hit-miss (CES): 100….01..→ behavior with **missed deadlines**

❖ **Safe behavior:** **Bounded deviation** from ideal behavior



- Deducing a **Safe CES**: Satisfies stable (M, K), exhibits safe behavior
- Establishing **safe behavior** over **infinite time horizon**

Step 3: Synthesizing SMT-based Schedule

Real-time constraints + **Stability-safety** constraints + **Minimizing worst-case response time (WCRT)** → **SMT-optimizer** → SAT → Feasible uniprocessor **schedule**

5. Experimental Evaluation

Scalability

- **Time-** and **compute-efficient scheduling** with increased tasks/jobs
- **Multiple tasks** utilizing **high processor bandwidth:** Reports schedule within **reasonable time**

# Tasks	# Jobs	Util.	Time Taken
9	14,490	0.7 – 0.97	Less than 4s
11	18,270	0.7 – 0.98	Less than 3 min
13	20,790	0.7 – 0.97	Less than 7 min
15	46,620	0.7 – 0.85	Less than 38 min

Comparison with State-of-the-Art

State-of-the-Art: PGS, DSHT, SCS

Proposed	PGS [1]	SCS [2], DSHT [3]
Stable, **safe**, handles **multiple tasks**, **time-efficient**, **improved schedulability**	Stable, not safe, high runtime overhead, fails to schedule mostly	Safe but only for bounded time, mostly unstable, high runtime overhead, multiple task-handling is a challenge

Proposed method outperforms all state-of-the-arts!

6. Conclusion

- We first time develop an SMT-based schedule with minimized WCRT, ensuring safety and stability for infinite time.
- We design a time-efficient, scalable scheduling approach that outperforms existing methods in a significant margin.

Author's email: debarpita2023_r@isical.ac.in