DATE 21
DESIGN, AUTOMATION & TEST IN EUROPE
01 – 05 February 2021 · virtual conference
The European Event for Electronic System Design & Test

Virtual Gang Scheduling of Parallel Real-Time Tasks

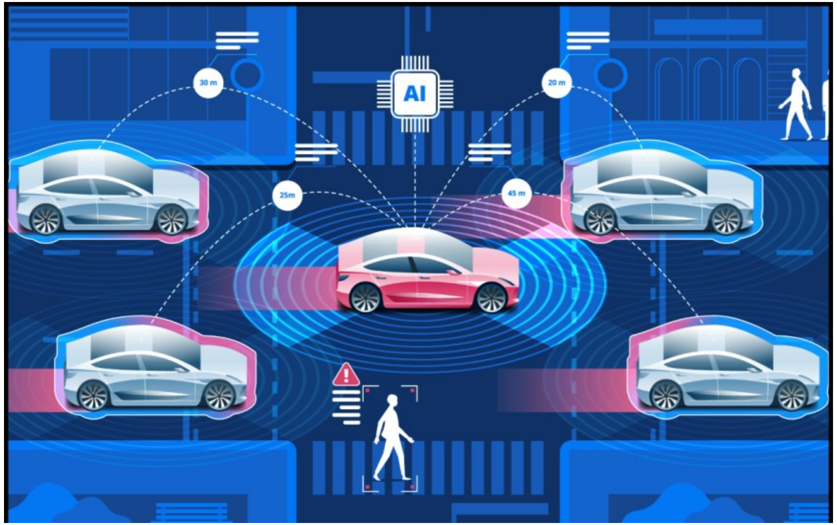
Waqar Ali[*], Rodolfo Pellizzoni[+] and Heechul Yun[*]

[*] THE UNIVERSITY OF KANSAS

[+] UNIVERSITY OF WATERLOO

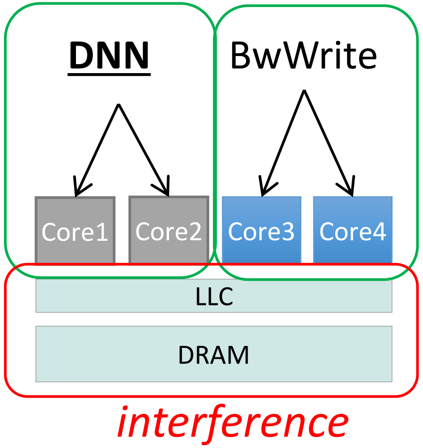
Parallel Real-Time Applications

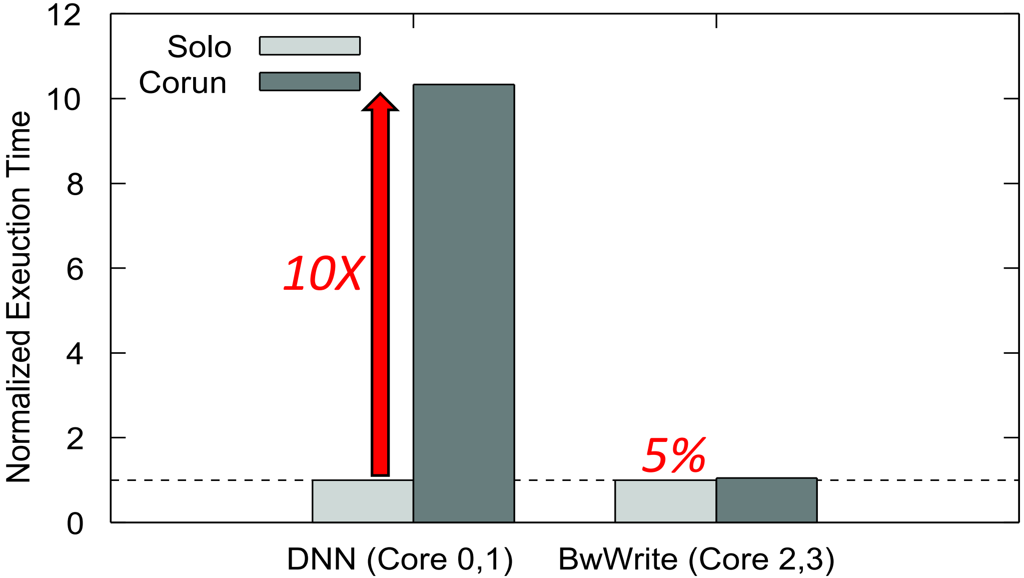
- Many emerging workloads are parallel real-time tasks
- Can benefit from parallelization on multicore platforms



Interference in Multicore

- Execution time of a task **depends** on co-running tasks
- Due to interference in shared hardware resources

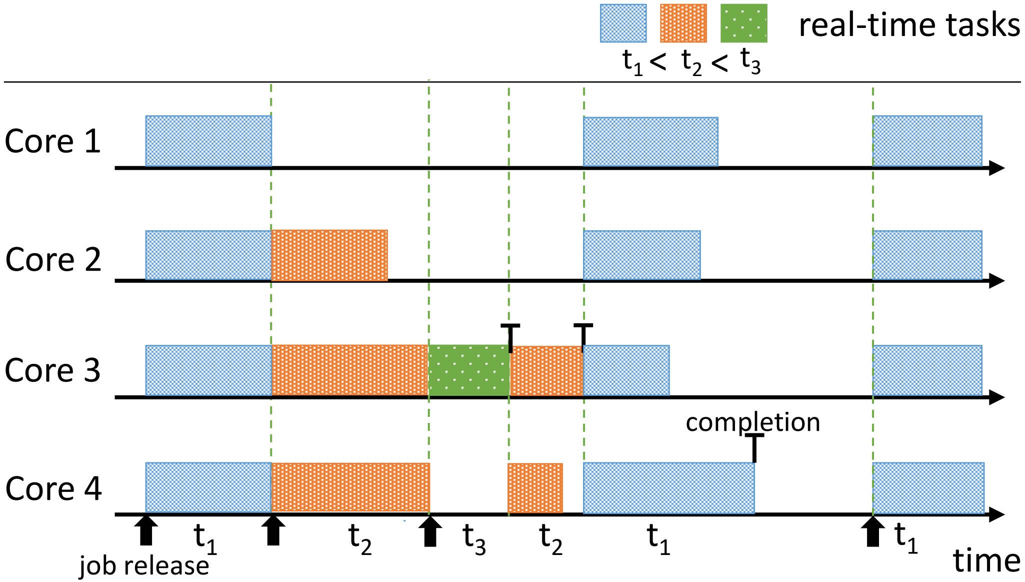




[1] W. Ali and H. Yun. "RT-Gang: Real-Time Gang Scheduling Framework for Safety-Critical Systems." In *RTAS* (2019)

RT-Gang

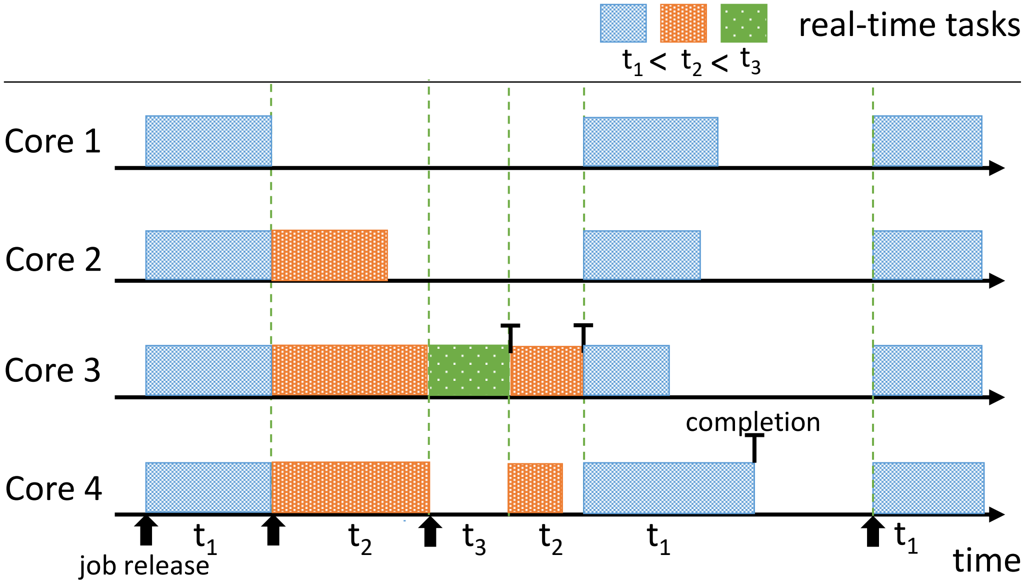
- A gang scheduling approach to avoid interference
- Schedule one gang task at a time



[1] W. Ali and H. Yun. "RT-Gang: Real-Time Gang Scheduling Framework for Safety-Critical Systems." In *RTAS* (2019)

Problems of RT-Gang

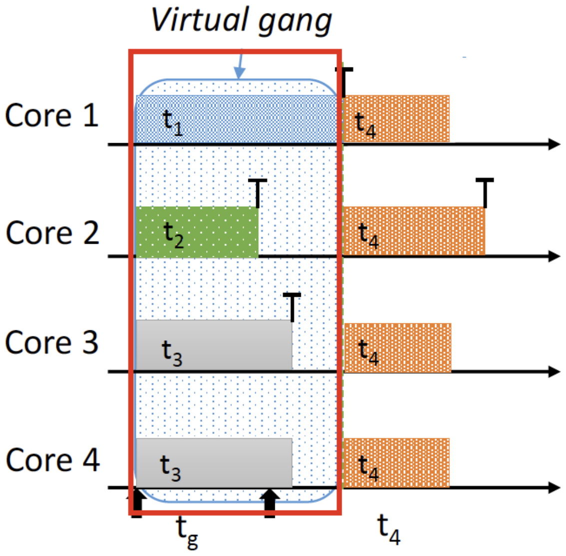
- **Not all tasks are fully parallelizable**
- **Low utilization**



[1] W. Ali and H. Yun. "RT-Gang: Real-Time Gang Scheduling Framework for Safety-Critical Systems." In *RTAS* (2019)

Our Approach: Virtual Gang Scheduling

- **Virtual Gang**
  - **A statically paired group of real-time tasks**
  - **Scheduled as a single gang task**
- **Benefits**
  - **Can achieve higher utilization**
  - **Co-running tasks never change**
  - **Can tightly bound interference**
- **Q: How to form the virtual gangs?**

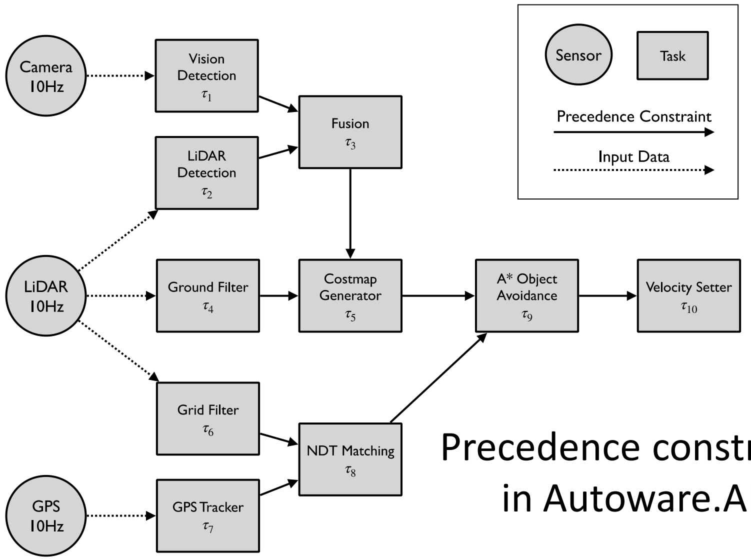


Real-Time Tasks: t_1, t_2, t_3, t_4
Virtual Gang (t_g): $t_1 + t_2 + t_3$
prio (t_g) > prio (t_4)

Considerations

- **Interference**
  - **Co-running tasks influence execution timing**

- **Precedence constraints**
  - **Common in real applications**
  - **Limit feasible task pairings**



Precedence constraints in Autoware.AI[2]

[2] S. Kato et al. "Autoware on board: Enabling autonomous vehicles with embedded systems." In *ICCPS* (2018)

System Model

- **A multicore platform with m unit-speed cores**
- **n periodic, rigid real-time gang tasks $\Gamma = \{\tau_1, \tau_2, ..., \tau_n\}$**

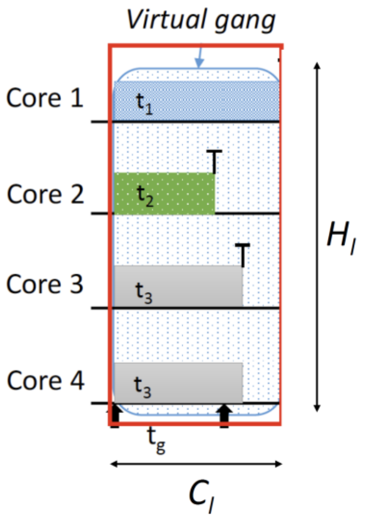
- **A real-time gang task: $\tau_i = (c_i, h_i, r_i, T_i)$**
  - c_i : WCET in isolation
  - h_i : #of cores required to run ($1 \leq h_i \leq m$)
  - r_i : Resource demand $r_i \in [0,1]$
  - T_i : Period of τ_i

- **Precedence constraints described by a set of DAGs**

System Model

- **A candidate-set:** $\Delta_T = \{\forall \tau_i \in \Gamma \mid T_i = T\}$
  - **Tasks that share a common period** T
  - **A small number of candidate-sets exists in** Γ

- **A virtual gang:** $w_l = (C_l, H_l, R_l, T_l)$
  - **A subset of tasks within a candidate set** Δ_T
  - C_l : WCET of the virtual gang
  - H_l : collective #core demand ($1 \leq H_i \leq m$)
  - R_l : collective resource demand ($R_i \geq 0$)
  - T_l : common shared period



Interference Model

- **A virtual gang w_l's WCET:** $C_l = \max_{\forall \tau_k \in w_l}\{c_k\} \times \max(R_l, 1)$
  - $R_l < 1$: **suffers no interference until the resource is over-utilized**
  - $R_l > 1$: **applies a *linear scaling***

Virtual Gang Formation Problem

- **For a given candidate set of N real-time tasks with the period T**
- **Form a set of virtual gang tasks that minimize completion time**
- **Subject to a given set of precedence constraints**

Optimal SMT Algorithm

- **Based on Satisfiability Modulo Theories (SMT)**
  - **Step-1:** **Identify the parameters of the optimization problem and the *optimization variable***
  - **Step-2:** **Write the constraints that must be satisfied by the parameters in a *feasible* solution**

- **See the paper for the details.**

Greedy Heuristic Algorithm

- **High-level idea**
  - **Group tasks with similar WCET values while minimizing combined resource utilization**

  →higher core utilization, less slowdown

- **Algorithm**
  - **Sort tasks by WCET**
  - **Pick the longest**
  - **Find feasible co-runners**
  - **Rank the feasible co-runners considering resource utilization**
  - **Form a virtual gang and repeat**

- **Not optimal but fast and effective**

Algorithm 1: Virtual Gang Formation Heuristic

Input: Candidate Set (Δ_T), Number of Cores (m)
Output: Taskset comprising virtual gangs
function `gang_formation` (Δ_T, m)
  $pq = sort\_tasks\_by\_wcet(\Delta_T)$
  $virtualGangs = ()$
  while $not\_empty(pq)$ **do**
    $\tau_i = pq.pop()$
    $f_i = family(\tau_i)$
    $partners = ()$
    for $\tau_j \in pq$ **do**
      if $\tau_i.h + \tau_j.h \leq m \ \wedge \ \tau_j \notin f_i$ **then**
        $partners \leftarrow partners \cup \{\tau_j\}$
    $pq_i = score\_partners(partners)$
    while $not\_empty(pq_i)$ **do**
      $\tau_p = pq_i.pop()$
      $\tau_i = merge(\tau_i, \tau_p)$
      $pq.remove(\tau_p)$
      $update\_partners(\tau_i, pq_i)$
    $virtualGangs \leftarrow virtualGangs \cup \{\tau_i\}$
return $virtualGangs$

Response Time Analysis

- **For a taskset of virtual gangs $\{w_1, w_2, \ldots, wl\}$ and under the rate-monotonic priority assignment, the response time $\mathbb{R}_i$ of gang w_i can be iteratively computed using:**

$$\mathbb{R}_i^{k+1} = \overline{C_i} + \sum_{\forall w_j \in hp(w_i)} \left\lceil \frac{\mathbb{R}_i^k}{T_j} \right\rceil C_j$$

  - **$\overline{C_i}$ is the WCET of w_i + the WCET of all w_j (with the same period T as w_i) that come before w_i in the linear execution order**

First work which enables schedulability analysis of real-time gang tasks with precedence constraints!

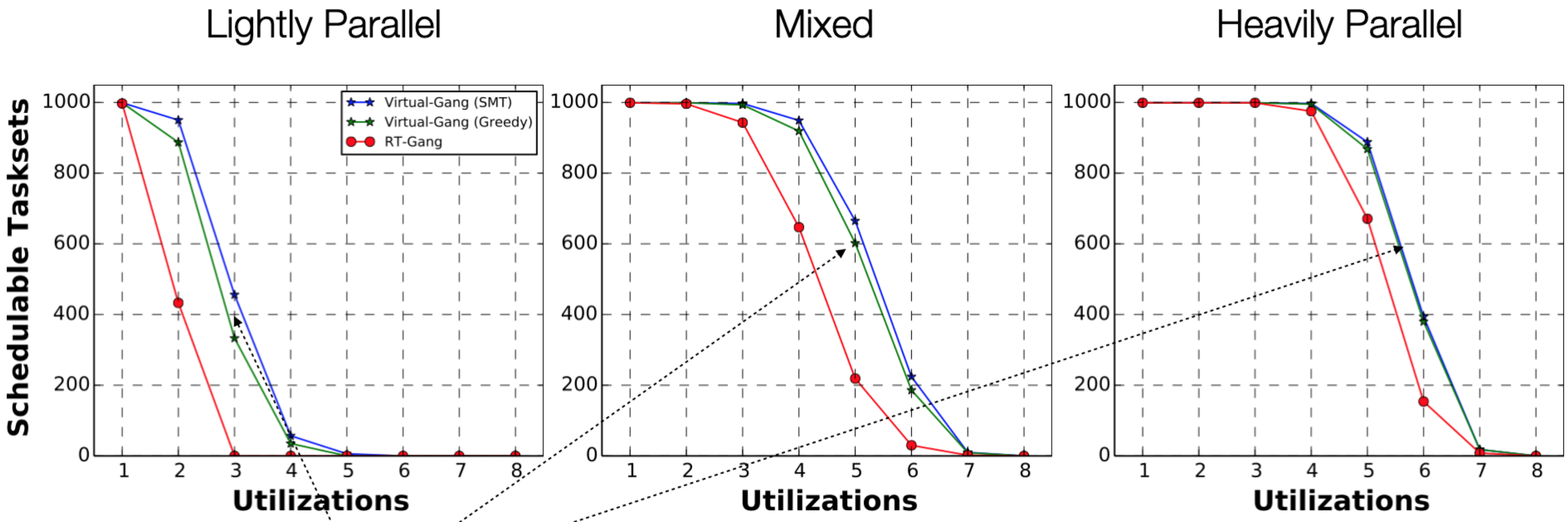
Evaluation

- **Simulation study with randomly generated synthetic tasksets**
  - **Lightly parallel (**h=[1, $\lceil 0.3 \times m \rceil$]**)**
  - **Mixed (**h=[1, m]**)**
  - **Heavily parallel (**h=[$\lceil 0.3 \times m \rceil$, m]**)**
  - **See the paper for other parameters (period, tasks, resource demand, etc.)**

- **Compared scheduling schemes**
  - **RT-Gang: one gang task at a time** [1]
  - **Virtual Gang (SMT/Greedy): our approach**
  - **Gang-FTP: fixed-priority gang scheduling** [3]
  - **Threaded: vanilla Linux (global FP scheduling)** [4]

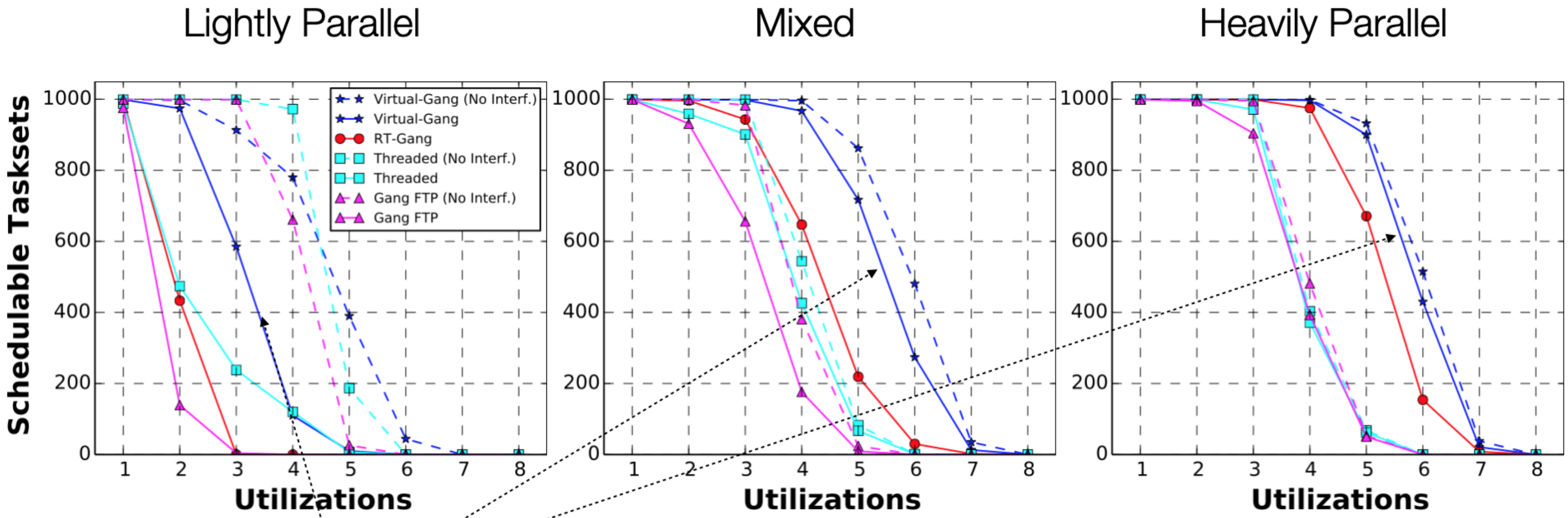
[1] W. Ali and H. Yun. "RT-Gang: Real-Time Gang Scheduling Framework for Safety-Critical Systems," In *RTAS* (2019)
[3] S. Wasly and R. Pellizzoni, "Bundled Scheduling of Parallel Real-Time Tasks," In *RTAS* (2019)
[4] J. Fonseca et al., "Improved Response Time Analysis of Sporadic DAG Tasks for Global FP Scheduling," In *RTNS* (2017)

Results (with Precedence Constraints)



- Heuristic performs very close to the optimal algorithm!
  - Schedulability is highest for virtual-gang (SMT)

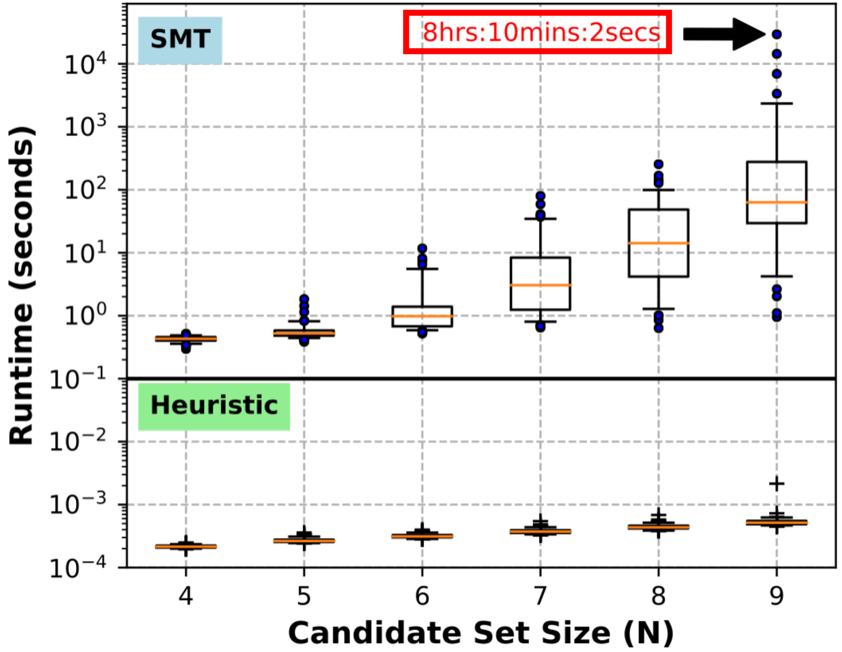
Results (w/o Precedence Constraints)



- When interference is considered:
  - ✓ Virtual Gang scheme outperforms every other analysis for all taskset types

Runtime of Gang Formation Algorithms

SMT can be very slow!



Even for $N = 50$, the heuristic gives a solution in less than a second!

Summary

- **Proposed a virtual-gang scheduling scheme**
  - **Achieves high utilization**
  - **Considers interference and precedence constraints**
  - **Enables effective and tight timing analysis**

- **Future work**
  - **Virtual gang formation on heterogeneous platforms (w/ accelerators)**
  - **Empirical evaluation with real-world workloads**

https://github.com/CSL-KU/VirtualGang-Simulator