The 39th IEEE Real-Time Systems Symposium (RTSS'18)

Schedulability Analysis of Adaptive Variable-Rate Tasks with Dynamic Switching Speeds

Chao Peng[1], Yecheng Zhao[2], **Haibo Zeng[2]**

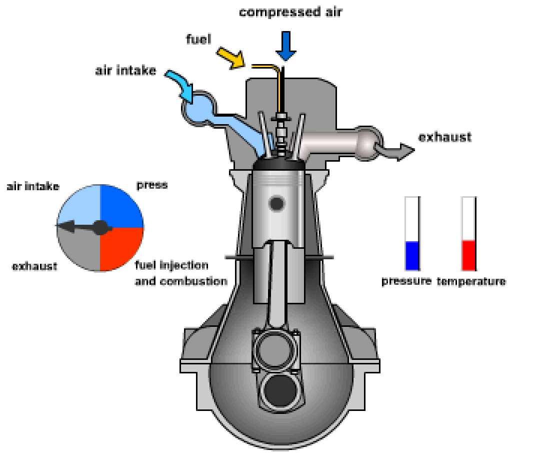
[1]National University of Defense Technology, China
[2]Virginia Tech, USA

Outline

- System Model: dAVR tasks
  - Motivated by the opportunity to optimize engine operation at runtime
- Schedulability Analysis
  - Partitioning speed space (safe but pessimistic)
  - Partitioning release time space (exact)
- Experimental Evaluation
  - Typically 10%-26% better accuracy than transforming to sAVR tasks

AVR Tasks

- Adaptive Variable-Rate (AVR) tasks
  - Different control strategies at different engine rotation speed intervals
  - Tradeoff between control performance and CPU workload

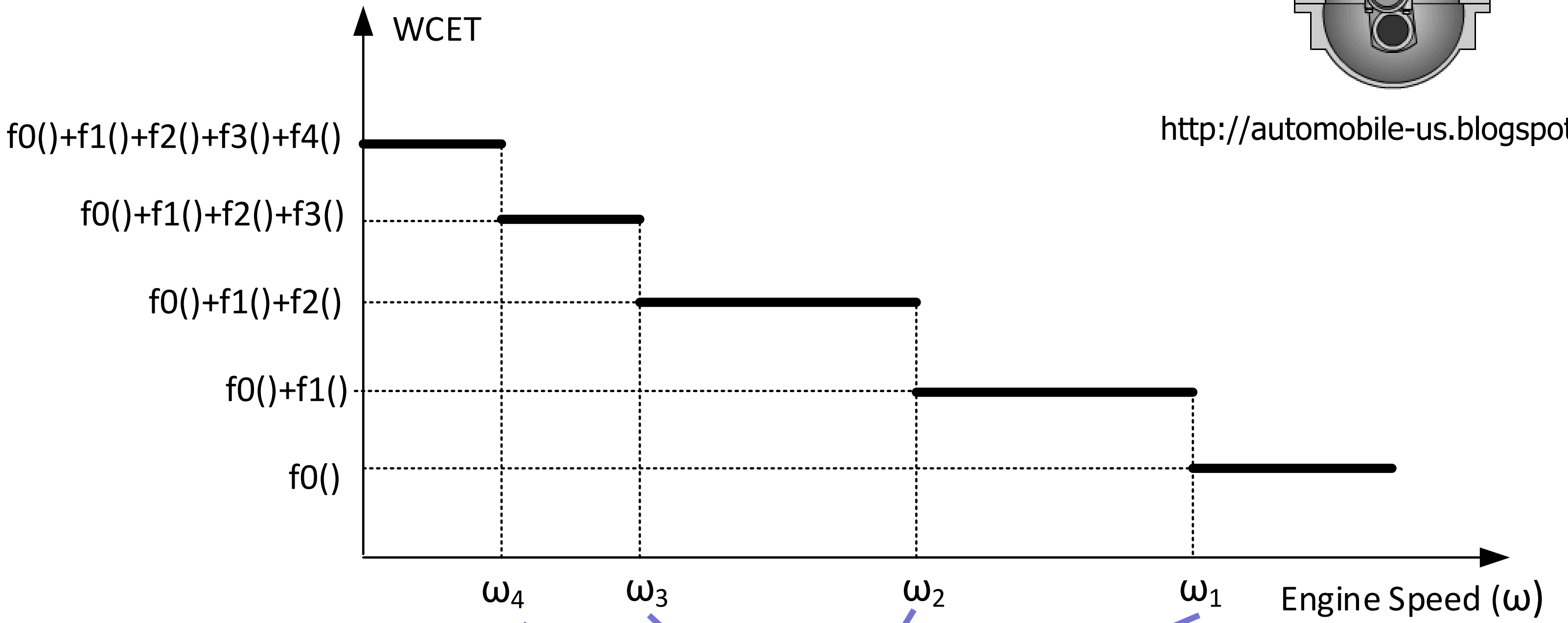


http://automobile-us.blogspot.com

```
#define ω4 1000
#define ω3 2000
#define ω2 4000
#define ω1 6000

task EC_task {
      ω = read_engine_speed();
      f0();
      if (ω ≤ ω1)     f1();
      if (ω ≤ ω2)     f2();
      if (ω ≤ ω3)     f3();
      if (ω ≤ ω4)     f4();
}
```

(a) [1]



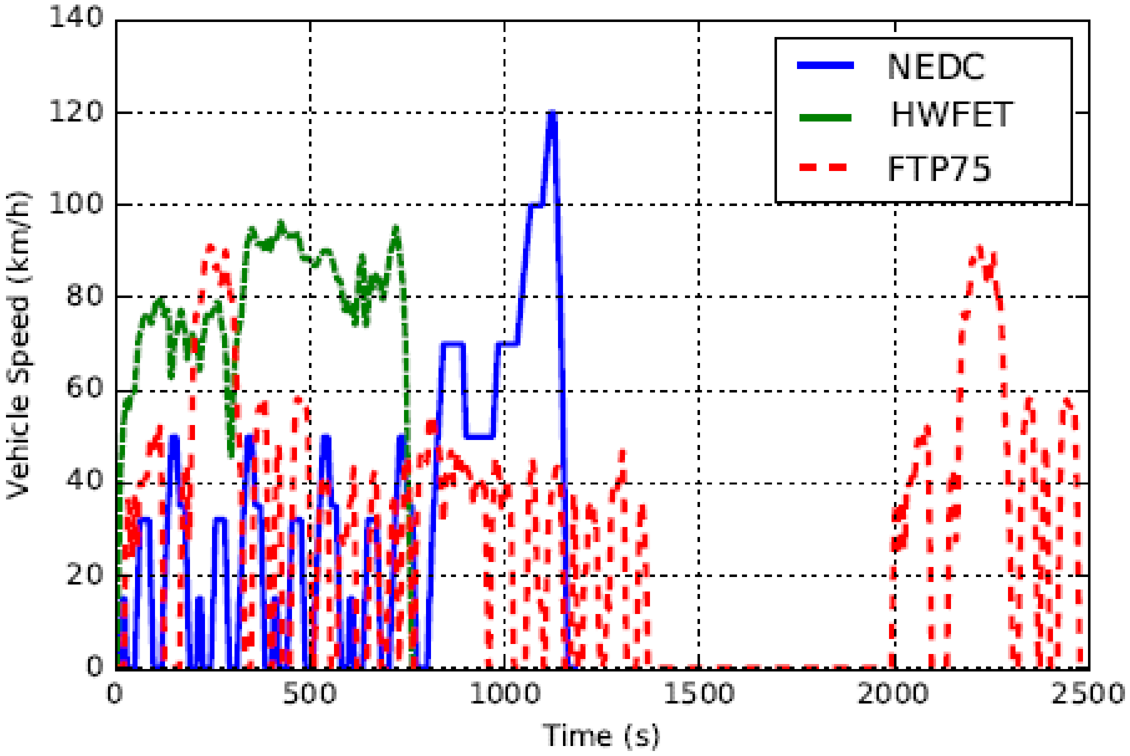
(b)

switching speeds **fixed at design time**

[1] D. Buttle, "Keynote speech: Real-time in the prime-time," in ECRTS 2012.

Motivation

- Suboptimal with static switching speeds
  - Difference among driving cycles
  - Variations within the same driving cycle



- Data available through connected and Automated Vehicles
  - Access to valuable information by using
    - Various sensing: camera, radar, lidar, …
    - Communication: V2V, V2I, ...
  - Opportunities to improve vehicle operation
    - Engine control (including switching speeds)

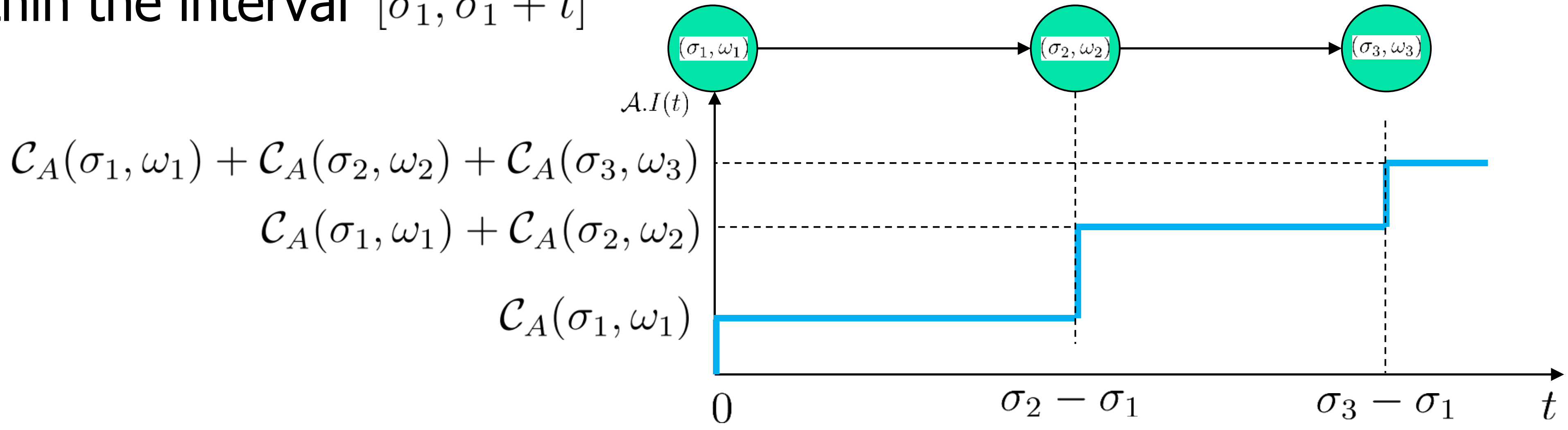
System Model

- Current approaches: static AVR (sAVR) tasks
  - the WCET of a job is a function of the engine speed
- Our proposal: dynamic AVR (dAVR) tasks
  - Reconfigured switching speeds at $\gamma_1, \gamma_2, \dots$
  - the WCET of a job is a function of the engine speed and release time

$$\mathcal{C}_i(t,\omega) = C_{i,k}^m \quad \text{if } t \in [\gamma_k, \gamma_{k+1}) \wedge \omega \in (\omega_{i,k}^{m-1}, \omega_{i,k}^m]$$

Schedulability Analysis of Periodic Tasks

- AVR job = (release time, engine speed at release time)
- AVR job sequence $\mathcal{A} = [(\sigma_1, \omega_1), \ldots, (\sigma_n, \omega_n)]$
- Interference function of AVR job sequence: the cumulative execution request within the interval $[\sigma_1, \sigma_1 + t]$



- Response time of periodic task τ_i interfered by AVR task τ_A^*

$$R(\tau_i, \tau_A^*) = \max_{\mathcal{A} \in \tau_A^*} R(\tau_i, \mathcal{A}), \quad \text{where}$$

$$R(\tau_i, \mathcal{A}) = \min_{t>0} \left\{ t \mid C_i + \sum_{\tau_j \in hp(i)} \left\lceil \frac{t}{T_j} \right\rceil C_j + \mathcal{A}.I(t) \le t \right\}$$

Infinitely many $\mathcal{A}$ due to continuous spaces of both release time and engine speed

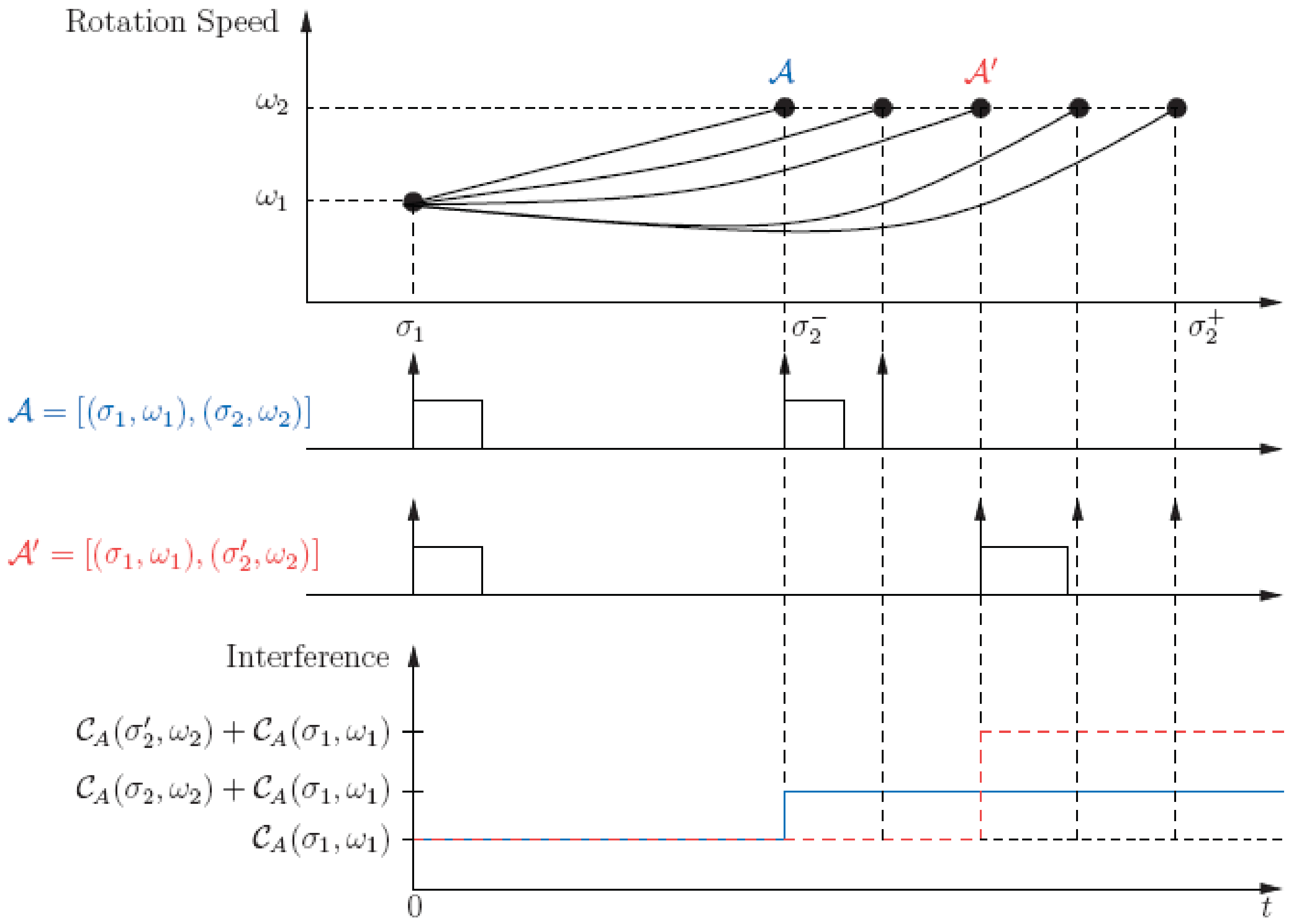
Difference between sAVR task and dAVR task

- Many analysis techniques for sAVR tasks are not applicable to dAVR

It is unsafe to only consider the minimum inter-release time between two consecutive jobs for dAVR tasks

⇓

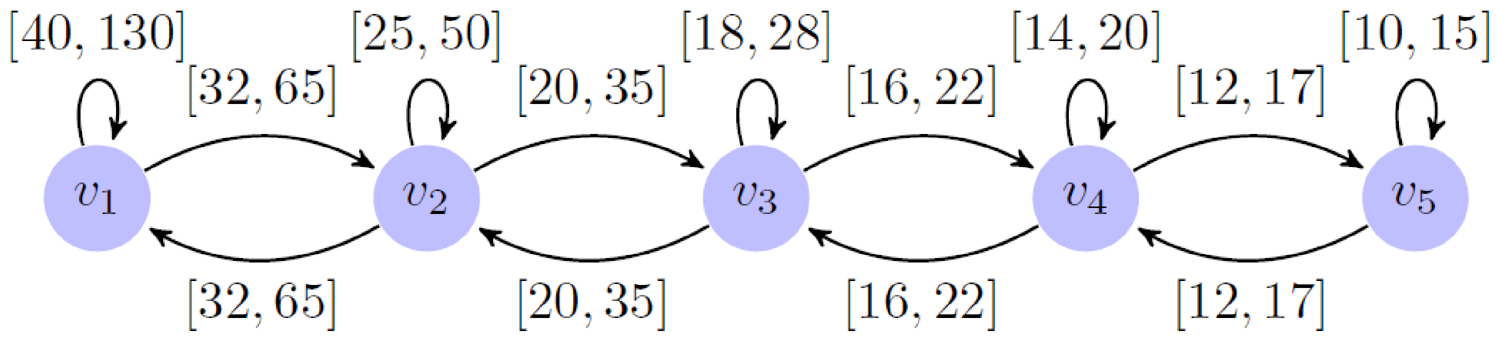
Necessary to keep the information on the minimum and maximum inter-release times between two consecutive jobs



Dynamic Digraph Real-Time (dDRT) Task Model

- Vertices represent a speed partition
  - Each vertex represents a separate speed interval
  - Also desirable to have WCET of each vertex only depends on the release time (but not the speed)

- An edge (v_i, v_j) is added if possible to release two consecutive jobs of type v_i and v_j respectively
  - Labeling it with the minimum and maximum inter-release times



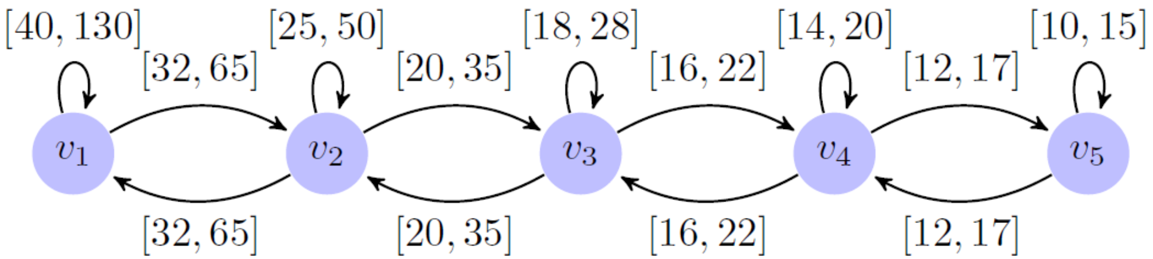
dAVR to dDRT Transformation

dAVR

Time Interval (ms)	Execution Modes					
$[\gamma_1, \gamma_2) = [0, 100)$	$\mathcal{M}_{i,1}$	m-th mode	1	2	3	
		$\omega_{i,1}^m$ (rpm)	2500	4500	6500	
		$C_{i,1}^m$ (μs)	600	400	200	
$[\gamma_2, \gamma_3) = [100, 200)$	$\mathcal{M}_{i,2}$	m-th mode	1	2	3	4
		$\omega_{i,2}^m$ (rpm)	1500	2500	4500	6500
		$C_{i,2}^m$ (μs)	600	400	300	200
$[\gamma_3, \gamma_4) = [200, +\infty)$	$\mathcal{M}_{i,3}$	m-th mode	1	2		
		$\omega_{i,3}^m$ (rpm)	3500	6500		
		$C_{i,3}^m$ (μs)	600	300		

Speed partition
{ [500,1500], [1500,2500], [2500,3500], [3500,4500], [4500,6500] }

dDRT



v_1 represents speed interval [500, 1500]
v_2 represents speed interval [1500,2500]
v_3 represents speed interval [2500,3500]
v_4 represents speed interval [3500,4500]
v_5 represents speed interval [4500,6500]

WCET function

Time Interval (ms)	WCET (μs)				
	v_1	v_2	v_3	v_4	v_5
$[\gamma_1, \gamma_2) = [0, 100)$	600	600	400	400	200
$[\gamma_2, \gamma_3) = [100, 200)$	600	400	300	300	200
$[\gamma_3, \gamma_4) = [200, +\infty)$	600	600	600	300	300

Schedulability Analysis based on Transformed dDRT

- Response time analysis based on the transformed dDRT τ_D^*
  - dDRT job = (release time, vertex)
  - dDRT job sequence $\mathcal{D} = [(\pi_1, \nu_1), \dots, (\pi_n, \nu_n)]$
  - Interference function of dDRT job sequence $\mathcal{D}.I(t)$
  - Response time of periodic task τ_i interfered by dDRT task τ_D^*

$$R(\tau_i, \tau_D^*) = \max_{\mathcal{D} \in \tau_D^*} R(\tau_i, \mathcal{D}), \quad \text{where}$$

$$R(\tau_i, \mathcal{D}) = \min_{t>0} \left\{ t \mid C_i + \sum_{\tau_j \in hp(i)} \left\lceil \frac{t}{T_j} \right\rceil C_j + \mathcal{D}.I(t) \leq t \right\}$$

Properties of the Transformation

- Safety
  - Intuition: Each vertex presents an interval of engine speeds. Thus, for any dAVR job sequence $\mathcal{A}$, we can always find a dDRT job sequence $\mathcal{D}$ such that they have the same interference function.
- Pessimism
  - For engines with maximum speed $\omega^{\mathrm{max}} > \omega^{\mathrm{t}}$ where ω^{t} depends on the maximum acceleration/deceleration, the analysis is pessimistic
  - Typical engines have $\omega^{\mathrm{max}} > 5.79\ \omega^{\mathrm{t}}$

Remaining Difficulty for Analyzing dDRT Systems

- Impractical to directly adopt the following computing equation

$$R(\tau_i, \tau_D^*) = \max_{\mathcal{D}\in\tau_D^*} R(\tau_i, \mathcal{D}), \quad \text{where}$$

$$R(\tau_i, \mathcal{D}) = \min_{t>0} \left\{ t \mid C_i + \sum_{\tau_j \in hp(i)} \left\lceil \frac{t}{T_j} \right\rceil C_j + \mathcal{D}.I(t) \le t \right\}$$

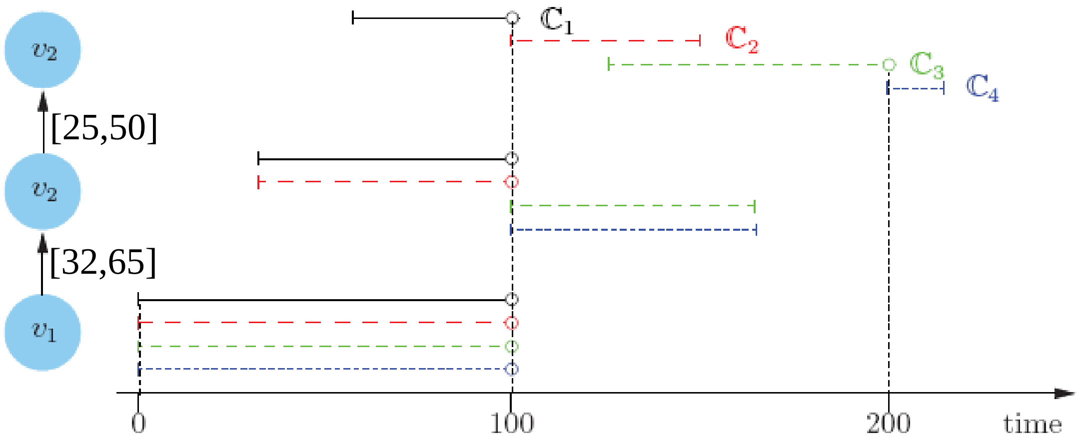
Still infinitely many $\mathcal{D}$ due to continuous space of release time

- Our Intuition:
  - If two dDRT job sequences share the same sequence of job WCETs, then the one always with a shorter inter-release time will dominate the other in terms of their interference functions

Digraph Path $(\nu_1, \ldots, \nu_n)$ → Construct → Common-WCET dDRT Job Sequence Set $\mathbb{C}$ → Find → Critical dDRT Job Sequence of $\mathbb{C}$

Construction of Critical dDRT Job Sequence

- Example of constructing common-WCET dDRT job sequence sets



(a) Partitioning release time ranges by the reconfiguration times $\gamma_1 = 0, \gamma_2 = 100, \gamma_3 = 200$.

$\mathbb{C}$	(c_1^-, c_1^+, ν_1)	(c_2^-, c_2^+, ν_2)	(c_3^-, c_3^+, ν_3)
1	$(0, 100, v_1)$	$(32, 100, v_2)$	$(57, 100, v_2)$
2			$(100, 150, v_2)$
3		$(100, 165, v_2)$	$(125, 200, v_2)$
4			$(200, 215, v_2)$

(b) Resultant four common-WCET dDRT job sequence sets.

contains invalid job sequence

- Finding critical dDRT job sequences
  - Idea: the inter-release times are valid but also as tight as possible

```
1:  procedure ConstructCriticalJobSequence(ℂ)
2:      ℂ̃ ← [(c̃1−, c̃1+, ν1), ..., (c̃n−, c̃n+, νn)];
3:      c̃n− ← cn−, c̃n+ ← cn+;
4:      for l = n to 2 do // Backward Pass
5:          c̃(l−1)− ← max(c̃l− − p^max(ν(l−1), νl), c(l−1)−);
6:          c̃(l−1)+ ← min(c̃l+ − p^min(ν(l−1), νl), c(l−1)+);
7:      D^c ← [(π1^c, ν1), ..., (πn^c, νn)];
8:      π1^c ← c̃1+;
9:      for l = 1 to n − 1 do // Forward Pass
10:         π(l+1)^c ← max(πl^c + p^min(νl, ν(l+1)), c̃(l+1)−);
11:     return D^c;
```

$\mathbb{C}$	Line 3		Lines 4–6				Line 8	Lines 9–10	
	$\tilde{c}_3^-$	$\tilde{c}_3^+$	$\tilde{c}_2^-$	$\tilde{c}_2^+$	$\tilde{c}_1^-$	$\tilde{c}_1^+$	π_1^c	π_2^c	π_3^c
1	57	100	32	75	0	43	43	75	100
2	100	150	50	100	0	68	68	100	125
3	125	200	100	165	35	100	100	132	157
4	200	215	150	165	85	100	100	150	200

can't be tighter due to validity

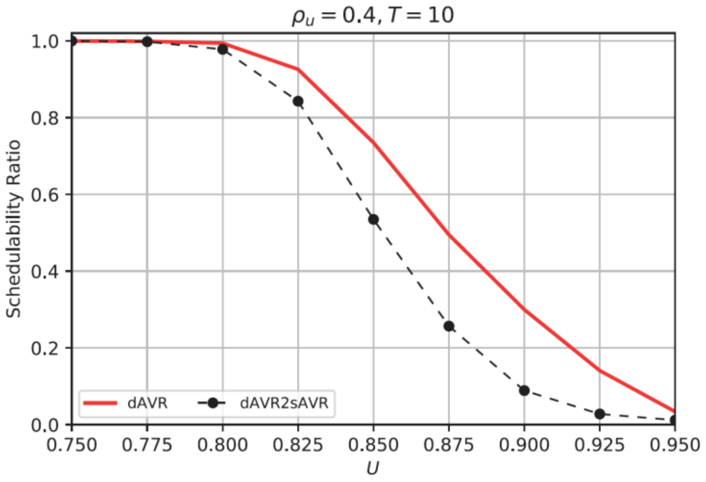
Experimental Setting

- Schedulability analysis methods
  - **dAVR**: our proposed approach
  - **dAVR2sAVR**: the sufficient-only analysis by transforming dAVR tasks to sAVR tasks and then apply the existing analysis on sAVR [1]
  - **UB**: the necessary-only analysis proposed in [2]
- Random task systems are generated following [1]
  - 20 Periodic tasks with constrained deadline:
    - period is randomly chosen from [3,100] ms
  - One AVR task: the execution modes are set as [1], with
    - Reconfiguration time interval = 500ms (typical driving cycle sample time)

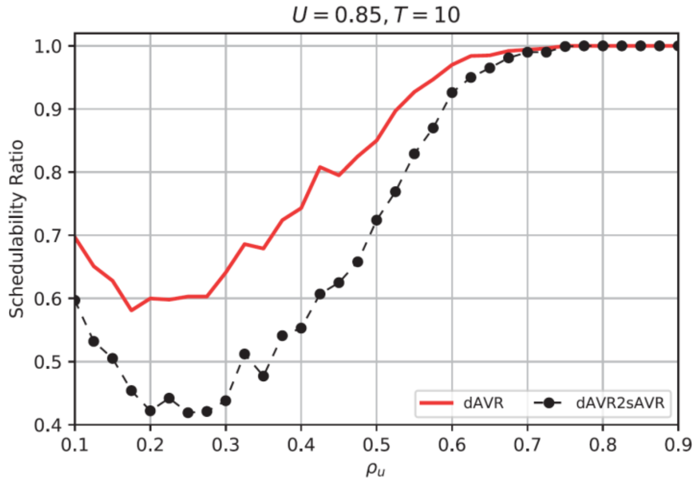
[1] Biondi et al., "Response-time analysis for real-time tasks in engine control applications," in IEEE/ACM Conference on Cyber-Physical Systems, 2015.
[2] Biondi et al., "Performance-driven design of engine control tasks," in IEEE/ACM Conference on Cyber-Physical Systems, 2016.

Experimental Results

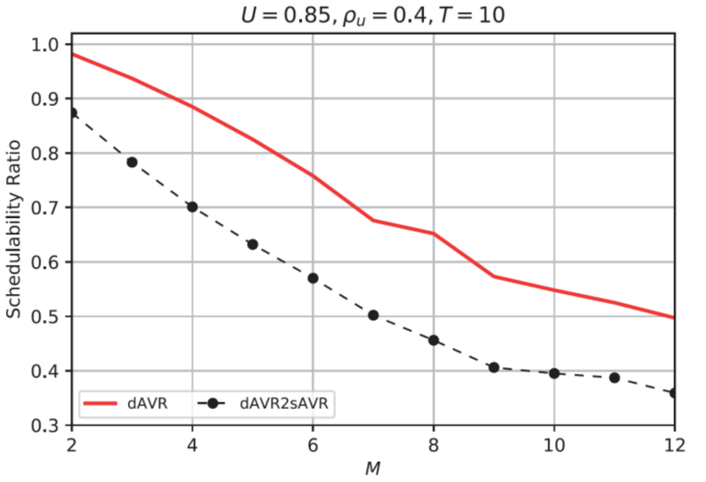
dAVR always has a higher analysis accuracy than dAVR2sAVR (typically 10%-26% in terms of schedulability ratio)



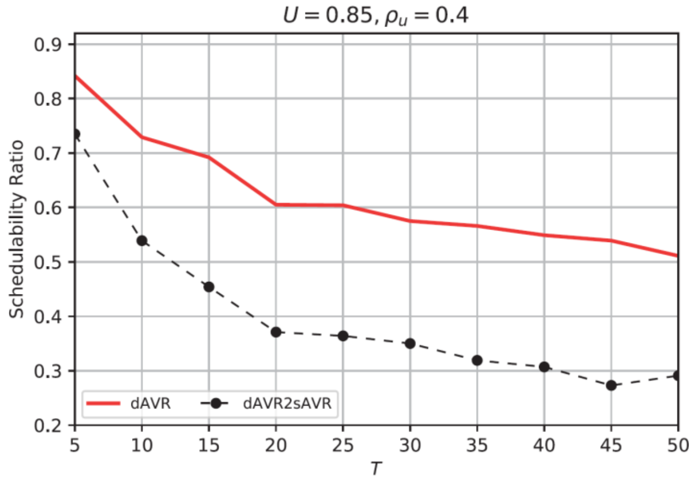
Schedulability ratio vs. system utilization U



Schedulability ratio vs. dAVR utilization fraction ρ_u



Schedulability ratio vs. number of modes M



Schedulability ratio vs. number of configurations T

Conclusion and Future Work

- Contributions:
  - Dynamic AVR task model that reconfigures the switching speeds at runtime
  - Response time analysis by partitioning
    - the speed space (sufficient-only)
    - release time space (exact)
- Typically 10%-26% more accurate than transforming it to static AVR

- Future work:
  - The approximation quality of speed partition
  - Optimization of switching speeds at runtime

Thank you!

Email: hbzeng@vt.edu