SCHEDULE OBFUSCATION

[RTAS 2016]

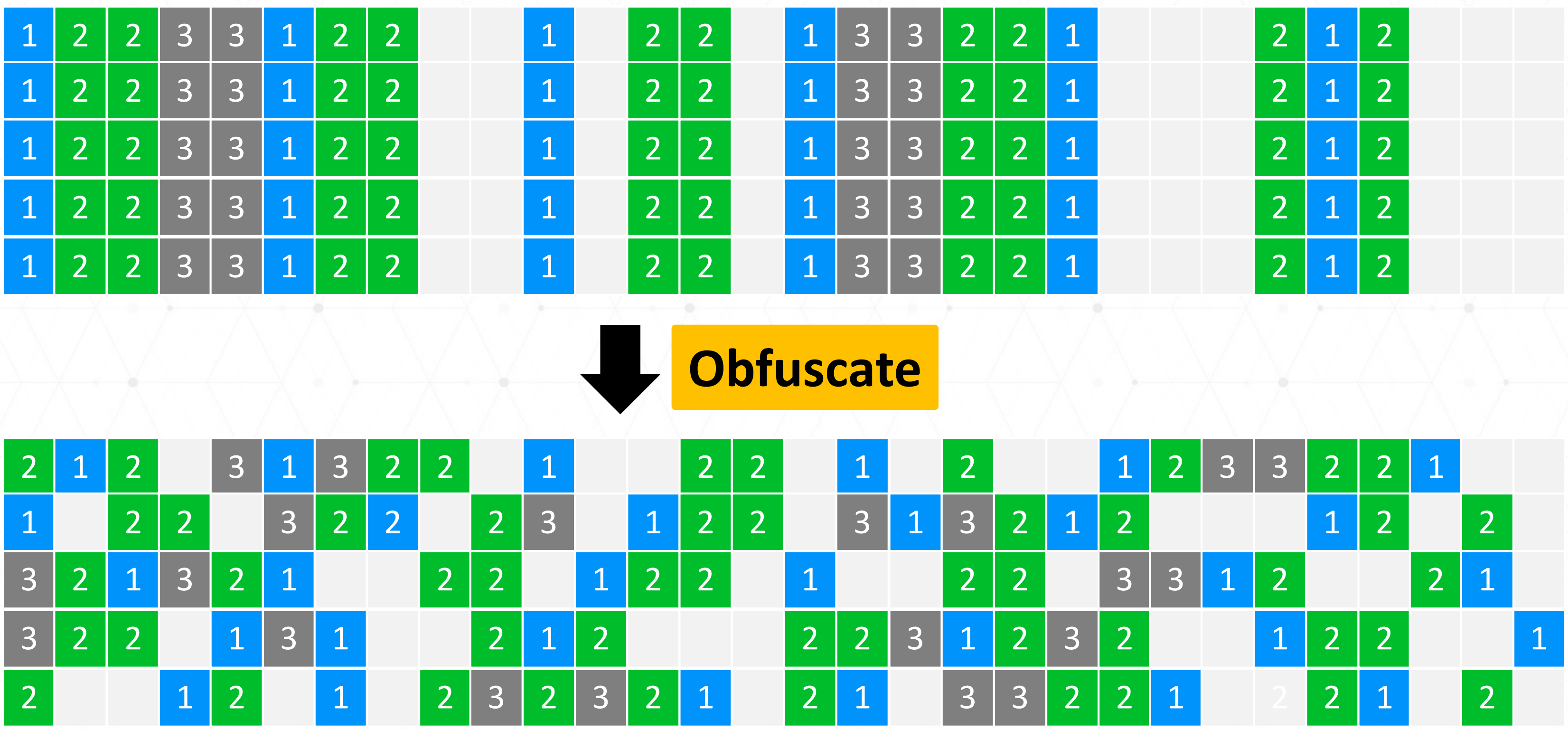
- Integrating security in Real-Time CPS
- Prevent attacks by randomizing schedule

“is it possible to reduce the regularity in real-time task schedules while still guaranteeing that the timing constraints (deadlines) are met?”

REAL-TIME SCHEDULE OBFUSCATION

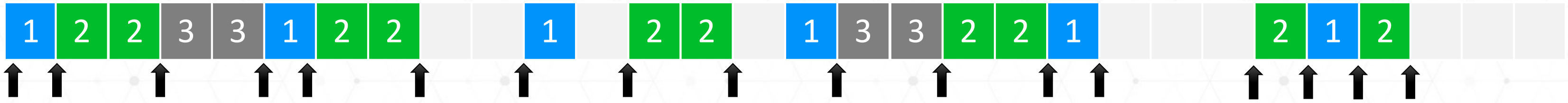
1	2	2	3	3	1	2	2			1		2	2		1	3	3	2	2	1				2	1	2			
1	2	2	3	3	1	2	2			1		2	2		1	3	3	2	2	1				2	1	2			
1	2	2	3	3	1	2	2			1		2	2		1	3	3	2	2	1				2	1	2			
1	2	2	3	3	1	2	2			1		2	2		1	3	3	2	2	1				2	1	2			
1	2	2	3	3	1	2	2			1		2	2		1	3	3	2	2	1				2	1	2			

REAL-TIME SCHEDULE OBFUSCATION



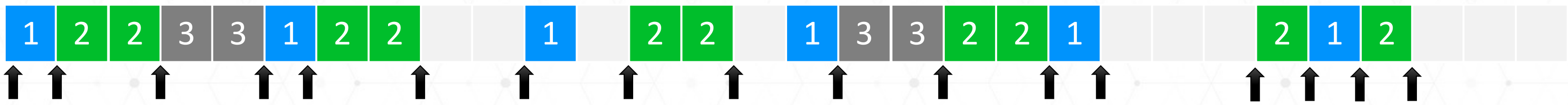
SCHEDULE OBFUSCATION: CONCEPT

SCHEDULE OBFUSCATION: CONCEPT



⬆ **At each scheduling point**

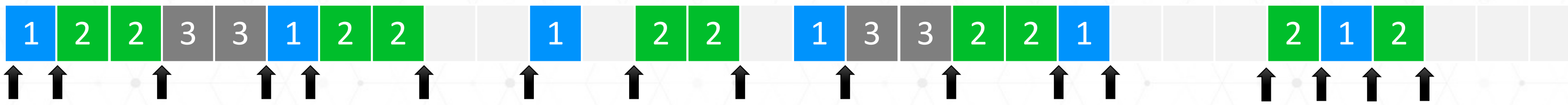
SCHEDULE OBFUSCATION: CONCEPT



⬆ **At each scheduling point**

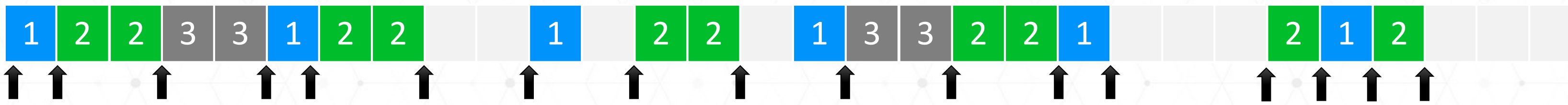
- **Pick a random task from the ready queue**
- **Not always the highest priority one**

SCHEDULE OBFUSCATION: CONCEPT



- ⬆ **At each scheduling point**
  - **Pick a random task from the ready queue**
  - **Not always the highest priority one**
- **Allow priority inversion**

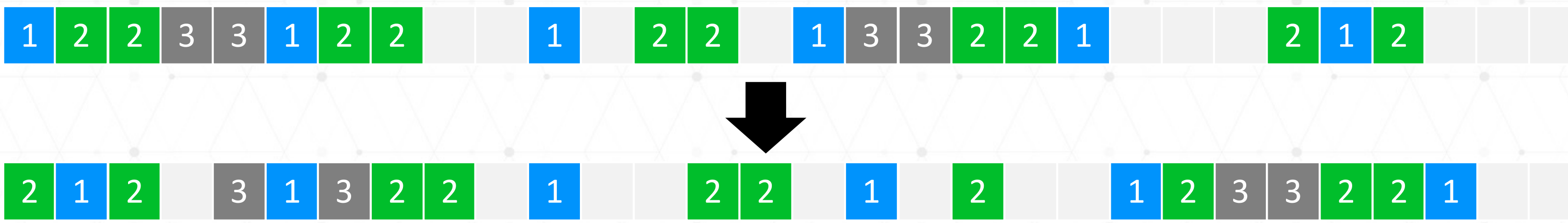
SCHEDULE OBFUSCATION: CONCEPT



- At each scheduling point
  - Pick a **random task** from the ready queue
  - Not always the highest priority one
- **Allow priority inversion**

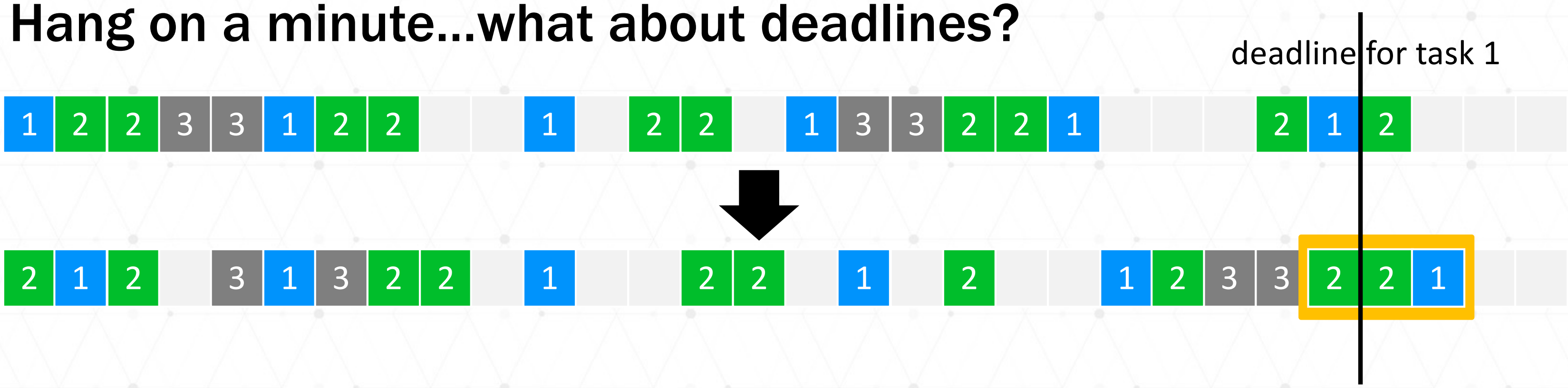
SCHEDULE OBFUSCATION: CONCEPT

- **Hang on a minute…what about deadlines?**



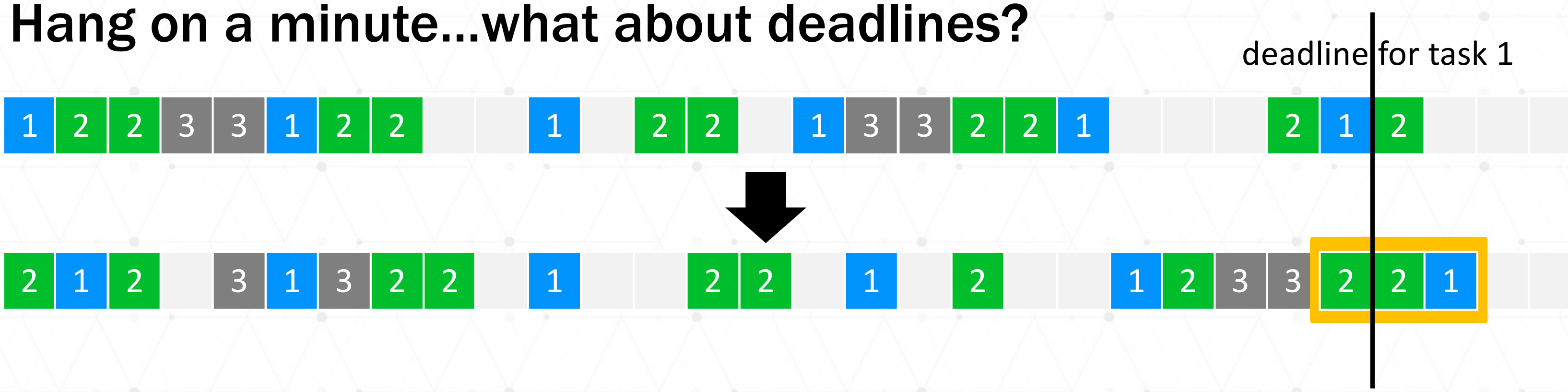
SCHEDULE OBFUSCATION: CONCEPT

- **Hang on a minute...what about deadlines?**



SCHEDULE OBFUSCATION: CONCEPT

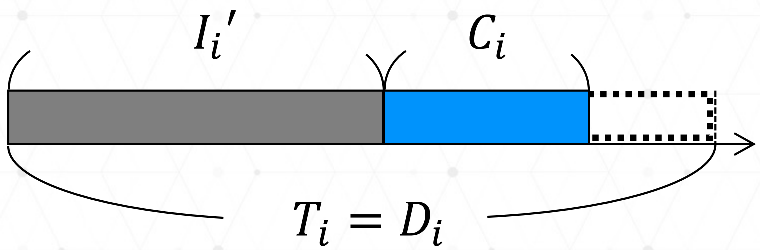
- **Hang on a minute…what about deadlines?**



- **Allow bounded priority inversion**
- **Tasks should still meet their original deadlines**
- **We must calculate ‘bounds’**
  - **how long can a higher priority task suffer inversion?**

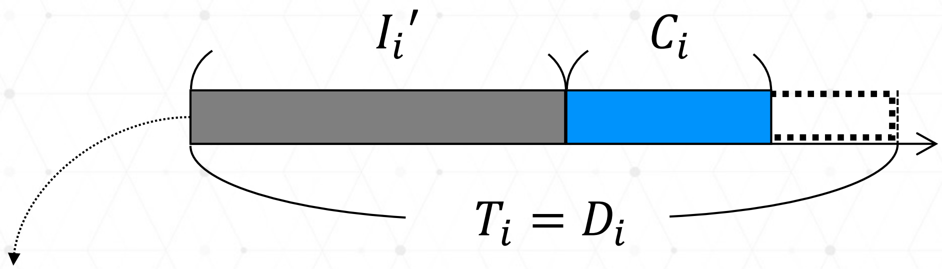
BOUNDING PRIORITY INVERSIONS

- **Let's consider a periodic task** τ_i



BOUNDING PRIORITY INVERSIONS

- **Let's consider a periodic task τ_i**

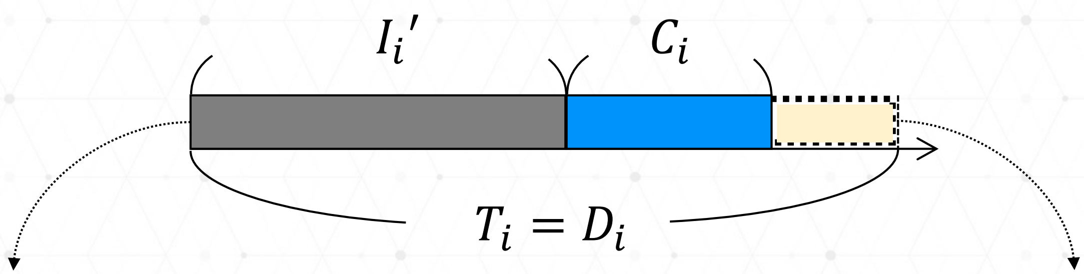


Interference induced by:
- higher priority tasks and
- priority inversion
needs to be taken into account

$$I_i' = \sum_{\tau_j \in hp(\tau_i)} \left(\left\lceil \frac{D_i}{T_j} \right\rceil + 1 \right) \cdot C_j$$

BOUNDING PRIORITY INVERSIONS

- **Let's consider a periodic task τ_i**



Interference induced by:
- higher priority tasks and
- priority inversion
needs to be taken into account

$$I_i' = \sum_{\tau_j \in hp(\tau_i)} \left(\left\lceil \frac{D_i}{T_j} \right\rceil + 1 \right) \cdot C_j$$

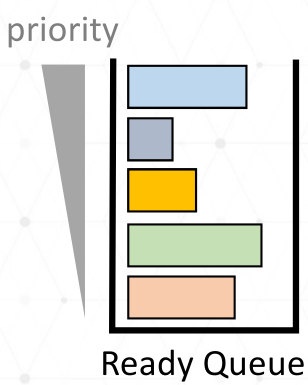
Extra delay that τ_i can tolerate without missing its deadlines

The worst-case inversion budget = V_i

TASKSHUFFLER RANDOMIZATION PROTOCOL

FIXED-PRIORITY SCHEDULING ALGORITHMS

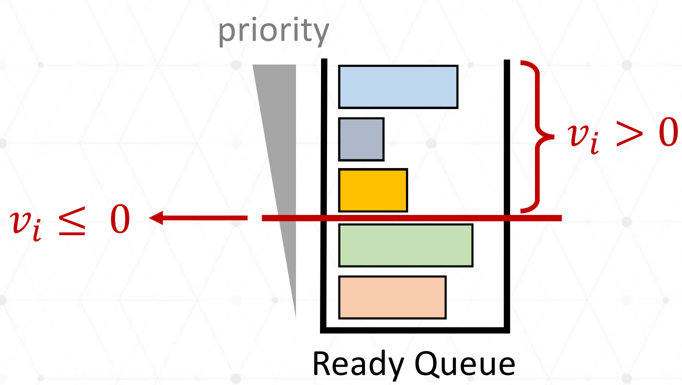
- **At each scheduling point**



TASKSHUFFLER RANDOMIZATION PROTOCOL

FIXED-PRIORITY SCHEDULING ALGORITHMS

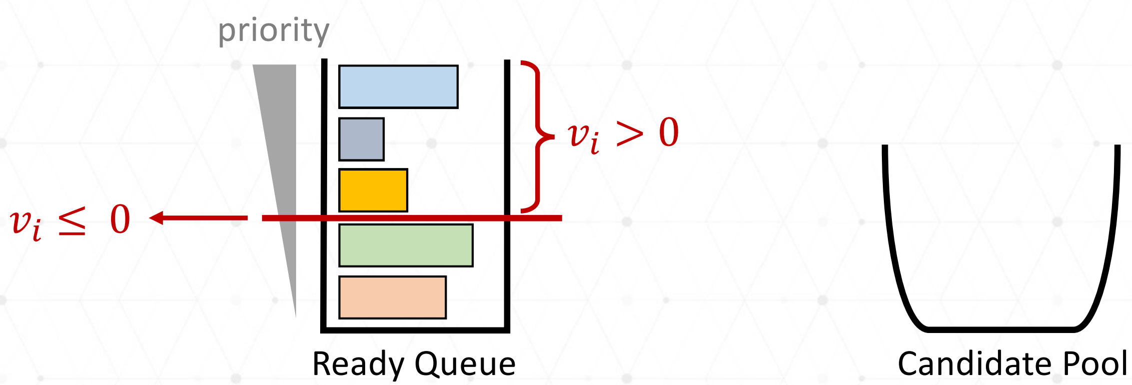
- **At each scheduling point**
  1. **Determine job candidates**



TASKSHUFFLER RANDOMIZATION PROTOCOL

FIXED-PRIORITY SCHEDULING ALGORITHMS

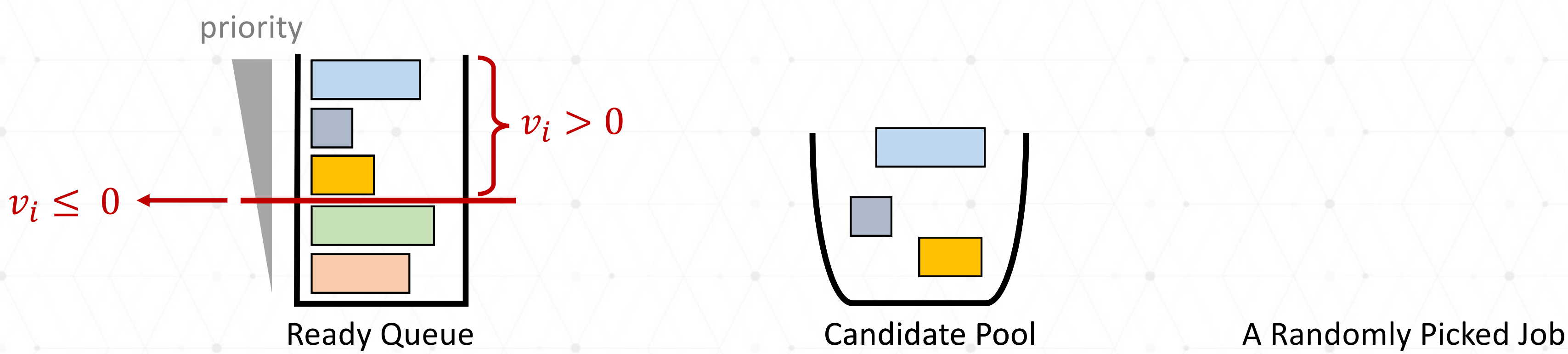
- **At each scheduling point**
  1. **Determine job candidates**



TASKSHUFFLER RANDOMIZATION PROTOCOL

FIXED-PRIORITY SCHEDULING ALGORITHMS

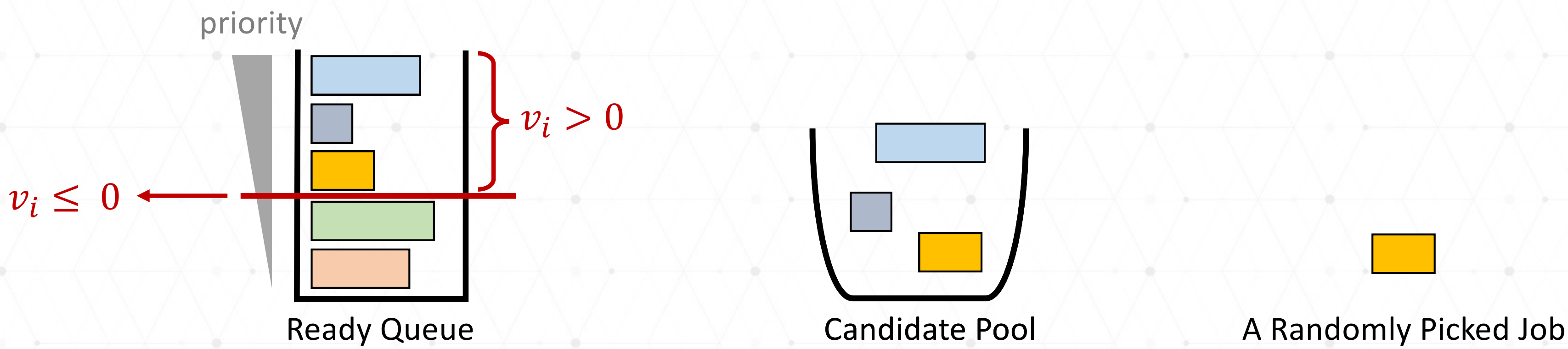
- **At each scheduling point**
  1. **Determine job candidates**
  2. **Randomly pick a job from candidate pool**



TASKSHUFFLER RANDOMIZATION PROTOCOL

FIXED-PRIORITY SCHEDULING ALGORITHMS

- **At each scheduling point**
  1. **Determine job candidates**
  2. **Randomly pick a job from candidate pool**
  3. **Set next scheduling point and run picked job**



RANDOMIZATION SCHEMES

- **Without Randomization**

HP 1	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 2	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 3	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 4	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 5	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 6	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 7	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 8	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 9	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 10	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 11	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 12	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 13	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 14	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 15	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 16	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 17	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 18	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 19	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 20	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3

RANDOMIZATION SCHEMES

- **Without Randomization**

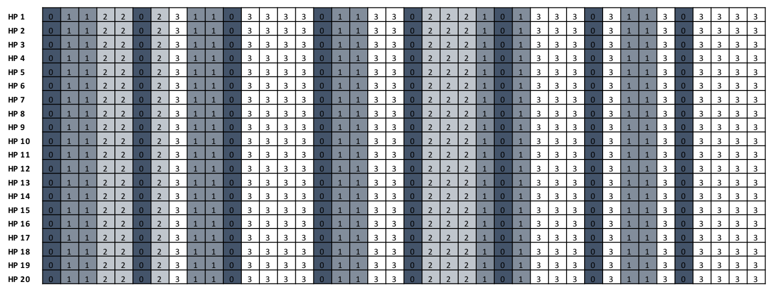
HP 1	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 2	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 3	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 4	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 5	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 6	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 7	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 8	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 9	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 10	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 11	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 12	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 13	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 14	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 15	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 16	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 17	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 18	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 19	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 20	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3

- **Task-only Randomization**

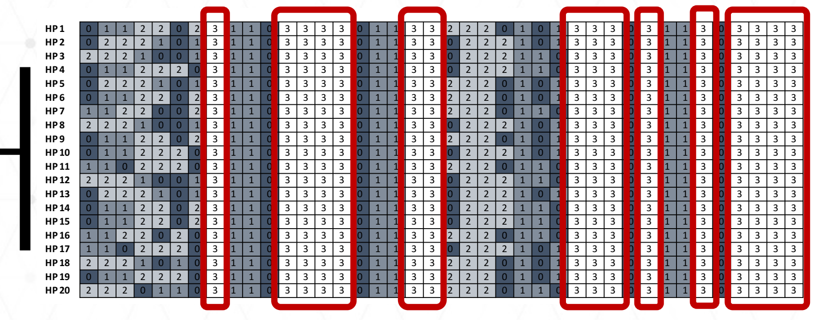
HP 1	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	2	2	2	0	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 2	0	2	2	2	1	0	1	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 3	2	2	2	1	0	0	1	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	1	0	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 4	0	1	1	2	2	2	0	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	1	0	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 5	0	2	2	2	1	0	1	3	1	1	0	3	3	3	3	0	1	1	3	3	2	2	2	0	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 6	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	2	2	2	0	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 7	1	1	2	2	0	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	2	2	2	0	1	1	0	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 8	2	2	2	1	0	0	1	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 9	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	2	2	2	0	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 10	0	1	1	2	2	2	0	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 11	1	1	0	2	2	2	0	3	1	1	0	3	3	3	3	0	1	1	3	3	2	2	2	0	1	1	0	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 12	2	2	2	1	0	0	1	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	1	0	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 13	0	2	2	2	1	0	1	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 14	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	1	0	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 15	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	1	0	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 16	1	1	2	2	0	2	0	3	1	1	0	3	3	3	3	0	1	1	3	3	2	2	2	0	1	1	0	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 17	1	1	0	2	2	2	0	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 18	2	2	2	1	0	1	0	3	1	1	0	3	3	3	3	0	1	1	3	3	2	2	2	0	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 19	0	1	1	2	2	2	0	3	1	1	0	3	3	3	3	0	1	1	3	3	2	2	2	0	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 20	2	2	2	0	1	1	0	3	1	1	0	3	3	3	3	0	1	1	3	3	2	2	2	0	1	1	0	3	3	3	0	3	1	1	3	0	3	3	3	3

RANDOMIZATION SCHEMES

- **Without Randomization**

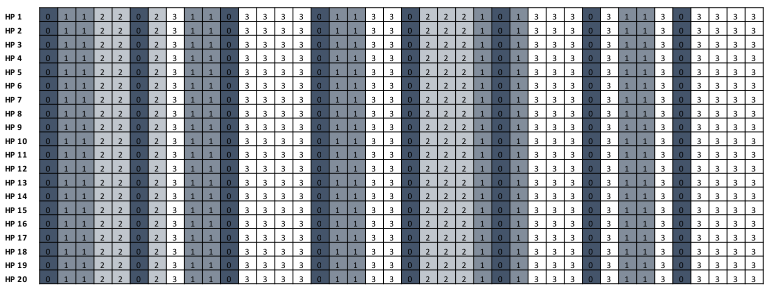


- **Task-only Randomization**

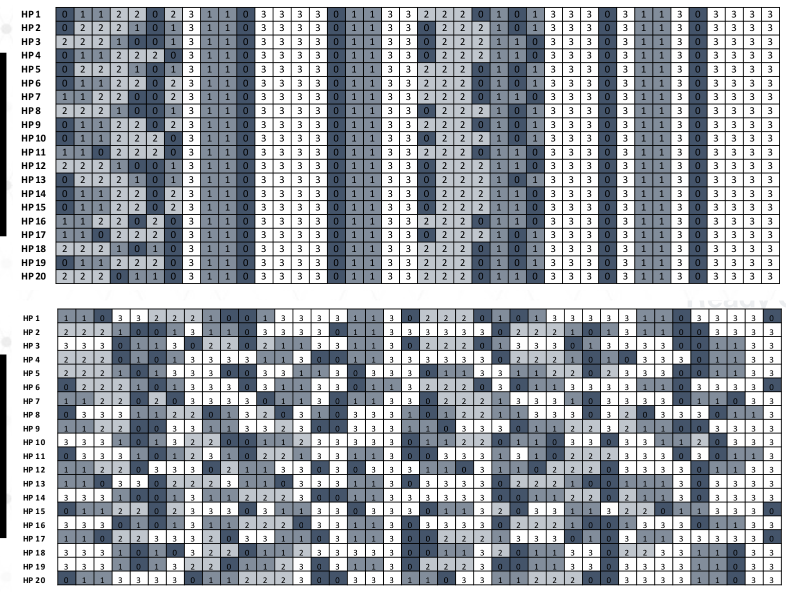


RANDOMIZATION SCHEMES

- **Without Randomization**

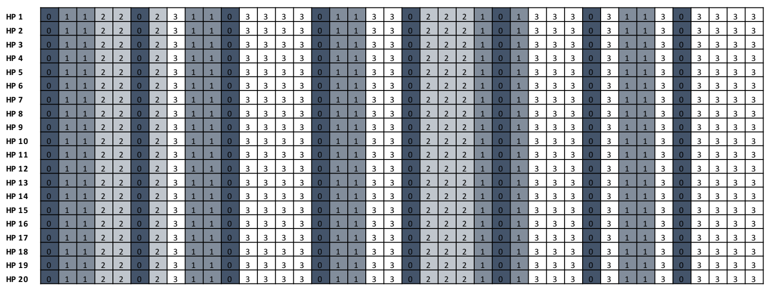


- **Task-only Randomization**
- **With Idle Time Scheduling**

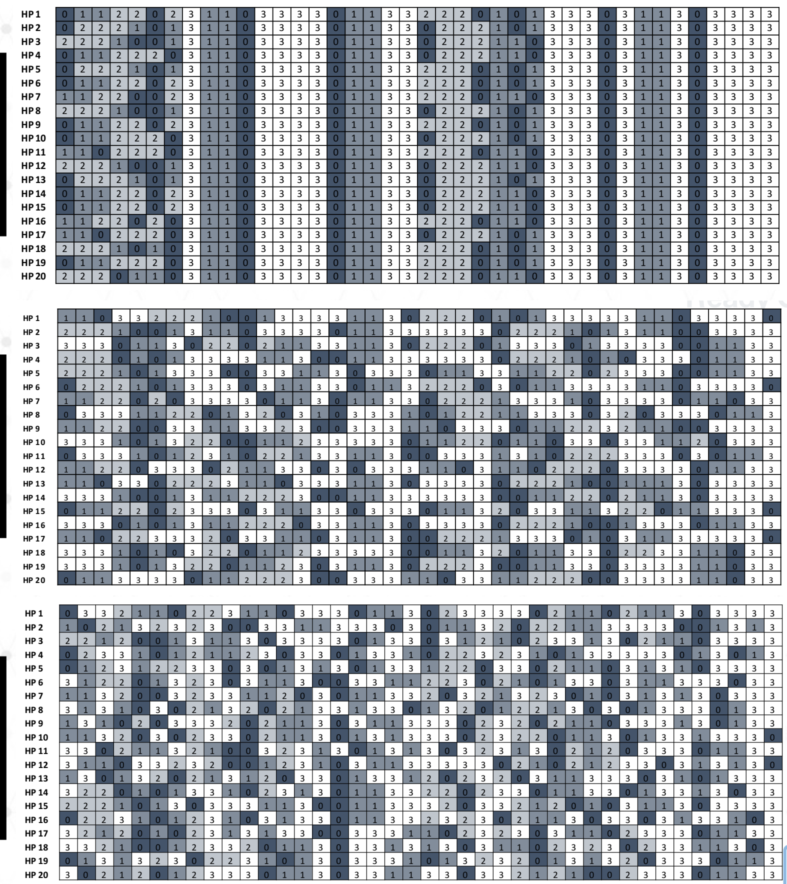


RANDOMIZATION SCHEMES

- **Without Randomization**

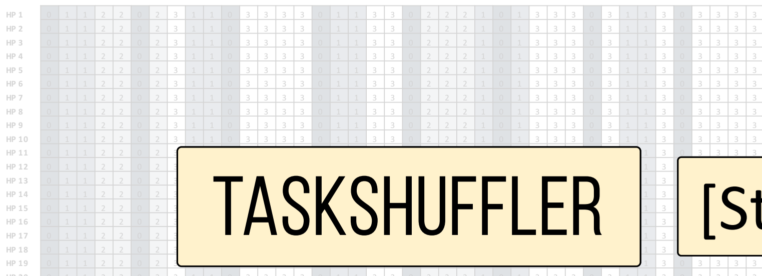


- **Task-only Randomization**
- **With Idle Time Scheduling**
- **Fine-grained Switching**



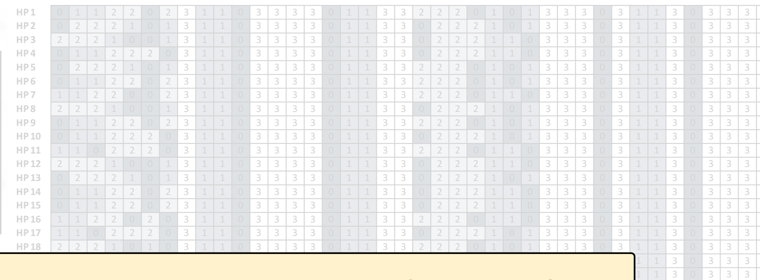
RANDOMIZATION SCHEMES

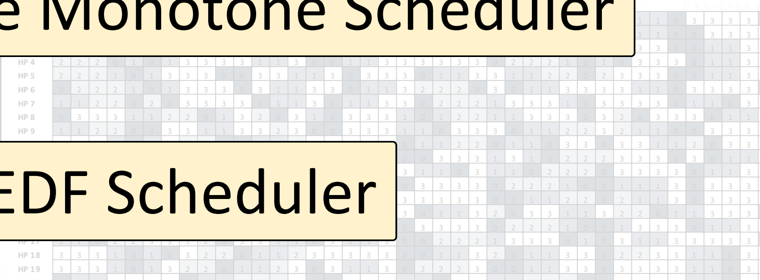
- Without Randomization



TASKSHUFFLER

[Static] Rate Monotone Scheduler





REORDER

[Dynamic] EDF Scheduler

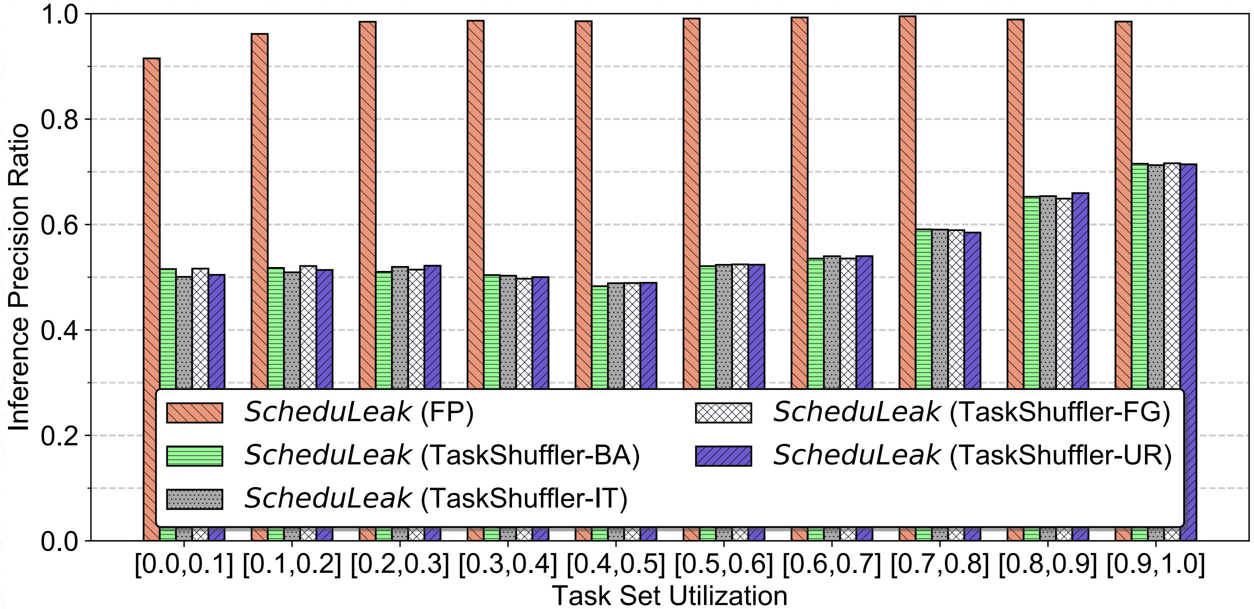
- Task-only R n
- With Idle Time Scheduling
- Fine-grained Switching

IMPLEMENTATION

- Platform
  - **Raspberry P**i 3 Model B
  - 1.2 GHz 64-bit quad-core ARM Cortex-A53
- Operating System
  - **Linux** kernel version: 4.9.48
  - Raspbian (a variant of Debian Linux)
  - Patched with **PREEMPT_RT**

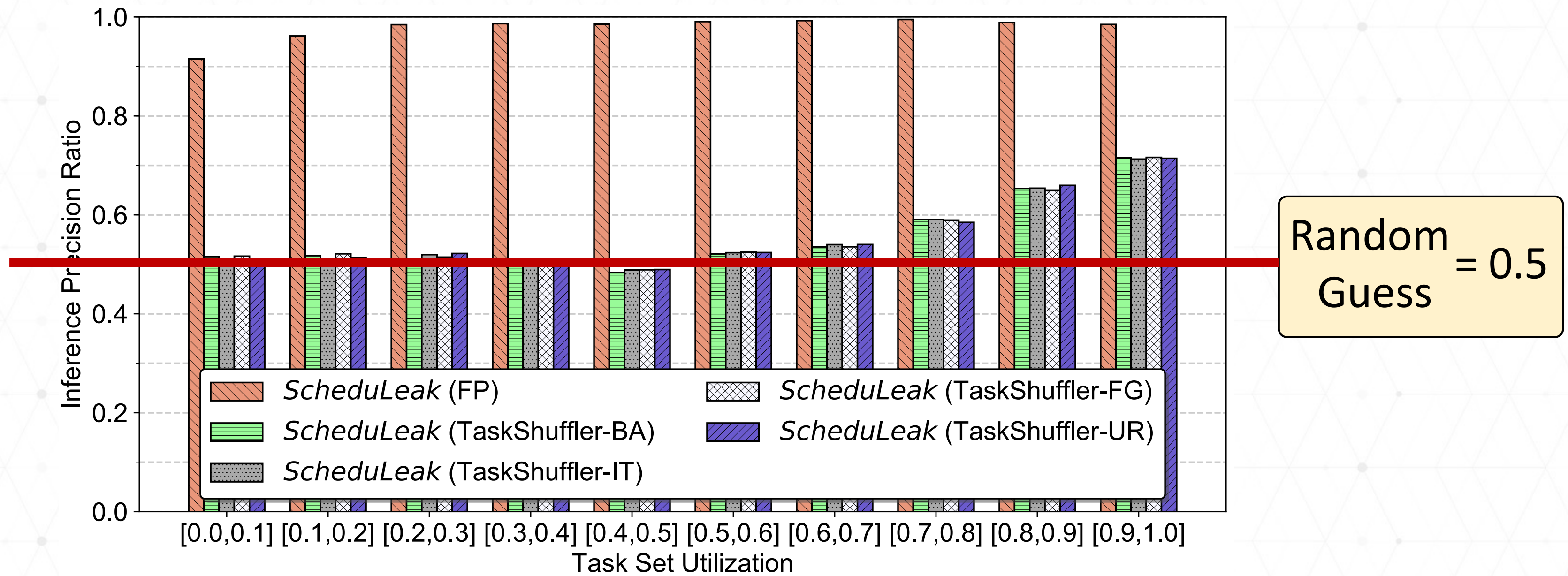
TASKSHUFFLER VS SCHEDULEAK

TASKSHUFFLER VS SCHEDULEAK



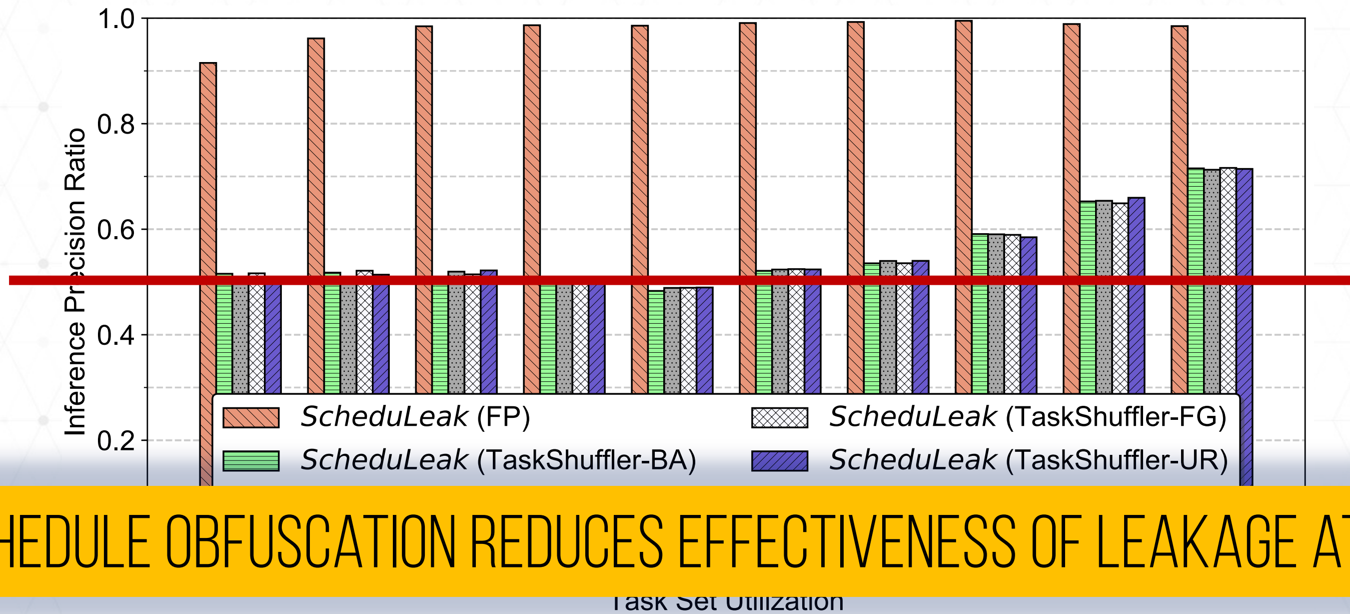
- Experiment Configurations
  - 6000 task sets tested
  - 600 each utilization group
  - 5, 7, 9, 11, 13, 15 tasks per task set
  - Each bar is averaged from 600 task sets

TASKSHUFFLER VS SCHEDULEAK



- Experiment Configurations
  - 6000 task sets tested
  - 600 each utilization group
  - 5, 7, 9, 11, 13, 15 tasks per task set
  - Each bar is averaged from 600 task sets

TASKSHUFFLER VS SCHEDULEAK



Random Guess = 0.5

SCHEDULE OBFUSCATION REDUCES EFFECTIVENESS OF LEAKAGE ATTACKS

- Experiment Configurations
  - 6000 task sets tested
  - 600 each utilization group
  - 5, 7, 9, 11, 13, 15 tasks per task set
  - Each bar is averaged from 600 task sets

METRICS

- **How do we model a successful obfuscation?**

"can we compare two obfuscated schedules and measure which one is better (in terms of protection against information leakage)?"

MEASURE OF RANDOMNESS?

HP 1	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	2	2	2	0	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 2	0	2	2	2	1	0	1	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 3	2	2	2	1	0	0	1	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	1	0	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 4	0	1	1	2	2	2	0	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	1	0	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 5	0	2	2	2	1	0	1	3	1	1	0	3	3	3	3	0	1	1	3	3	2	2	2	0	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 6	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	2	2	2	0	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 7	1	1	2	2	0	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	2	2	2	0	1	1	0	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 8	2	2	2	1	0	0	1	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 9	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	2	2	2	0	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 10	0	1	1	2	2	2	0	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 11	1	1	0	2	2	2	0	3	1	1	0	3	3	3	3	0	1	1	3	3	2	2	2	0	1	1	0	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 12	2	2	2	1	0	0	1	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	1	0	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 13	0	2	2	2	1	0	1	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 14	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	1	0	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 15	0	1	1	2	2	0	2	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	1	0	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 16	1	1	2	2	0	2	0	3	1	1	0	3	3	3	3	0	1	1	3	3	2	2	2	0	1	1	0	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 17	1	1	0	2	2	2	0	3	1	1	0	3	3	3	3	0	1	1	3	3	0	2	2	2	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 18	2	2	2	1	0	1	0	3	1	1	0	3	3	3	3	0	1	1	3	3	2	2	2	0	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 19	0	1	1	2	2	2	0	3	1	1	0	3	3	3	3	0	1	1	3	3	2	2	2	0	1	0	1	3	3	3	0	3	1	1	3	0	3	3	3	3
HP 20	2	2	2	0	1	1	0	3	1	1	0	3	3	3	3	0	1	1	3	3	2	2	2	0	1	1	0	3	3	3	0	3	1	1	3	0	3	3	3	3

<

HP 1	0	3	3	2	1	1	0	2	2	3	1	1	0	3	3	3	0	1	1	3	0	2	3	3	3	3	0	2	1	1	0	2	1	1	3	0	3	3	3	3
HP 2	1	0	2	1	3	2	3	2	3	0	0	3	3	1	1	3	3	3	0	3	0	1	1	3	2	0	2	2	1	1	3	3	3	3	0	0	1	3	1	3
HP 3	2	2	1	2	0	0	1	3	1	1	3	0	3	3	3	3	0	1	3	3	0	3	1	2	1	0	2	3	3	1	3	0	2	1	1	0	3	3	3	3
HP 4	0	2	3	3	1	0	1	2	1	1	2	3	0	3	3	0	1	3	3	1	0	2	2	3	2	3	1	0	1	3	3	3	3	3	0	1	3	0	1	3
HP 5	0	1	2	3	1	2	2	3	3	0	3	0	1	3	1	3	0	1	3	3	1	2	2	0	3	3	0	2	1	1	0	3	1	3	1	0	3	3	3	3
HP 6	3	1	2	2	0	1	3	2	3	0	3	1	1	3	0	0	3	3	1	1	2	2	3	0	2	1	0	1	3	3	0	3	1	1	3	3	3	0	3	3
HP 7	1	1	3	2	0	0	3	2	3	3	1	1	2	0	3	0	1	1	3	3	2	0	3	2	1	3	2	3	0	1	0	3	1	3	1	3	0	3	3	3
HP 8	3	1	3	1	0	3	0	2	1	3	2	0	2	1	3	3	1	3	3	0	1	3	2	0	1	2	2	1	3	0	3	0	1	3	3	3	0	1	3	3
HP 9	1	3	1	0	2	0	3	3	3	2	0	2	1	1	3	0	3	1	1	3	3	3	0	2	3	2	0	2	1	1	0	3	3	3	1	3	0	1	3	3
HP 10	1	1	3	2	0	3	0	2	3	3	0	2	1	1	3	0	1	3	1	3	3	3	0	2	3	2	2	0	1	1	3	0	1	3	3	1	3	3	3	0
HP 11	3	3	0	2	1	1	3	2	1	0	0	3	2	3	1	3	0	1	3	1	3	0	3	2	3	1	3	0	2	1	2	0	3	3	3	0	1	1	3	3
HP 12	3	1	1	0	3	3	2	3	2	0	0	1	2	3	1	0	3	1	1	3	3	3	3	3	0	2	1	0	2	1	2	3	3	0	3	1	3	1	3	0
HP 13	1	3	0	1	3	2	0	2	1	3	1	2	0	3	3	0	1	3	3	1	2	0	2	3	2	0	3	1	1	3	3	3	0	3	1	0	1	3	3	3
HP 14	3	2	2	0	1	0	1	3	3	1	0	2	3	1	3	0	1	1	3	3	2	2	0	2	3	3	0	1	1	3	3	0	1	3	3	1	3	0	3	3
HP 15	2	2	2	1	0	1	3	0	3	3	3	1	1	3	0	0	1	1	3	3	3	2	0	3	3	2	1	2	0	1	0	3	1	1	3	0	3	3	3	3
HP 16	0	2	2	3	1	0	1	2	3	1	0	3	1	3	3	0	1	1	3	3	2	3	2	3	0	2	1	1	3	0	3	3	0	3	1	3	3	1	0	3
HP 17	3	2	1	2	0	1	0	2	3	1	3	1	3	3	0	0	3	3	3	1	1	0	2	3	2	3	0	3	1	1	0	2	3	3	3	0	1	1	3	3
HP 18	3	3	2	1	0	0	1	2	3	3	2	0	1	1	3	0	3	3	1	3	1	3	0	3	1	1	0	2	3	2	3	0	2	3	3	1	1	3	0	3
HP 19	0	1	3	1	3	2	3	0	2	2	3	1	0	1	3	0	3	3	3	1	0	1	3	2	3	2	0	1	3	1	3	2	0	3	3	3	0	1	1	3
HP 20	3	0	2	1	2	0	1	2	3	3	3	0	1	1	3	0	3	3	1	1	3	3	0	3	3	2	1	2	1	0	0	2	3	3	3	0	1	1	3	3

UPPER APPROXIMATION OF SCHEDULE ENTROPY

- **Upper-approximation = sum of slot entropies**

$$\widetilde{H}_\Gamma(S) = \sum_{slot} H_\Gamma(S_t)$$

$$= -\sum_{slot}\sum_{task} \Pr(task\ at\ slot)\log_2 \Pr(task\ at\ slot)$$

UPPER APPROXIMATION OF SCHEDULE ENTROPY

- **Upper-approximation = sum of slot entropies**

$$\widetilde{H}_{\Gamma}(S) = \sum_{slot} H_{\Gamma}(S_t)$$

Uncertainty in seeing a particular task at a specific slot

$$= -\sum_{slot}\sum_{task} \Pr(task\ at\ slot) \log_2 \Pr(task\ at\ slot)$$

UPPER APPROXIMATION OF SCHEDULE ENTROPY

- **Upper-approximation = sum of slot entropies**

$$\widetilde{H}_\Gamma(S) = \sum_{slot} H_\Gamma(S_t)$$

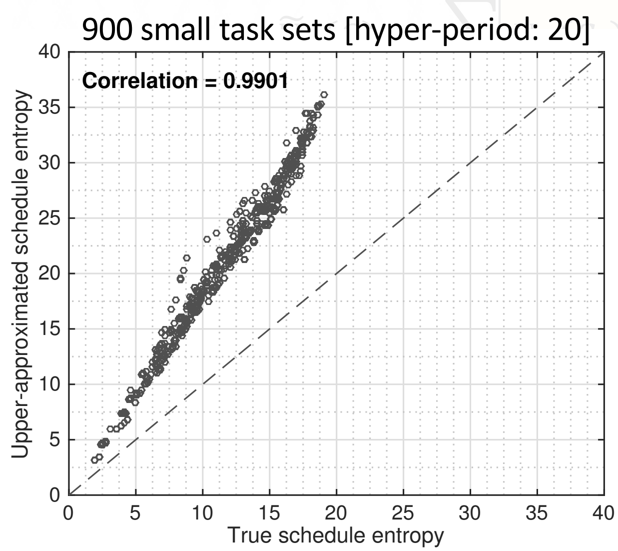
Uncertainty in seeing a particular task at a specific slot

$$= -\sum_{slot}\sum_{task} \Pr(task\ at\ slot)\log_2 \Pr(task\ at\ slot)$$

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Pr(Task 1 at slot i)	0.25	0.19	0.14	0.15	0.27	0.43	0.20	0.15	0.07	0.15	0.34	0.26	0.20	0.10	0.10	0.50
Pr(Task 2 at slot i)	0.25	0.27	0.26	0.28	0.34	0.30	0.30	0.00	0.39	0.37	0.27	0.30	0.30	0.30	0.07	0.00
Pr(Task 3 at slot i)	0.25	0.27	0.30	0.29	0.20	0.13	0.24	0.36	0.21	0.17	0.14	0.14	0.14	0.15	0.00	0.00
Pr(Task 4 at slot i)	0.25	0.27	0.29	0.29	0.20	0.13	0.26	0.50	0.33	0.30	0.25	0.30	0.35	0.45	0.84	0.50
$H_\Gamma(S_t)$	**2.00**	**1.99**	**1.95**	**1.95**	**1.96**	**1.82**	**1.98**	**1.44**	**1.80**	**1.90**	**1.94**	**1.95**	**1.92**	**1.78**	**0.80**	**1.00**

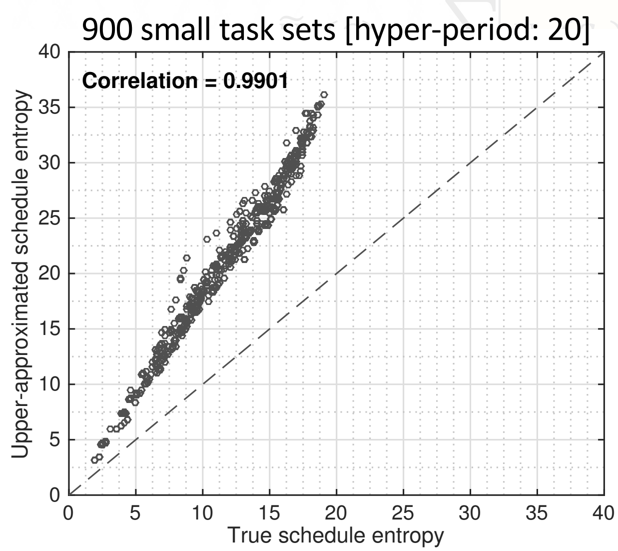
UPPER APPROXIMATION OF SCHEDULE ENTROPY

- Upper-approximation = sum of slot entropies



STRONG CORRELATION BETWEEN TRUE AND UPPER-APPROXIMATED SCHEDULE ENTROPIES

UPPER APPROXIMATION OF SCHEDULE ENTROPY



STRONG CORRELATION BETWEEN TRUE AND UPPER-APPROXIMATED SCHEDULE ENTROPIES

But does this accurately capture the randomness across task sets?

LIMITATIONS OF SLOT ENTROPY

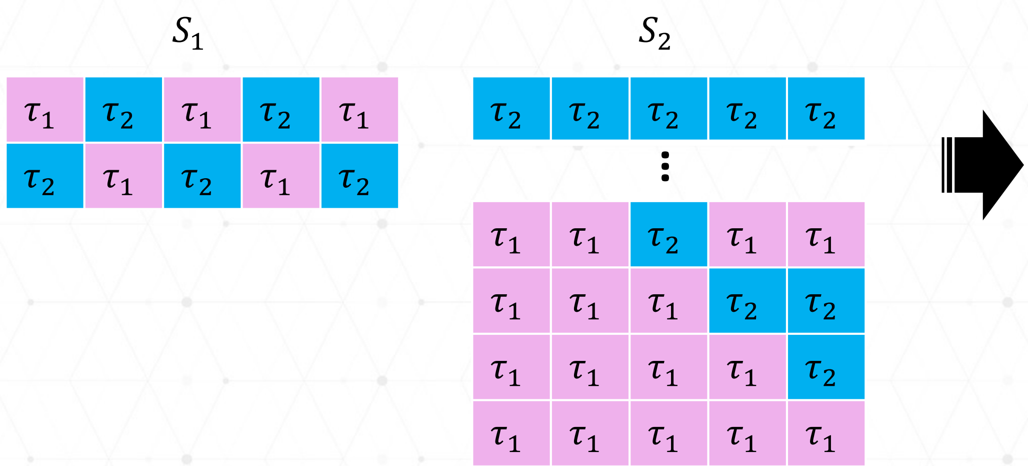
- $\widetilde{H}_{\Gamma}(S^k)$ **ignores the regularities that exist in** S^k

LIMITATIONS OF SLOT ENTROPY

- $\widetilde{H}_{\Gamma}(S^k)$ **ignores the regularities that exist in** S^k
- **Consider a task set** $\Gamma = \{\tau_1, \tau_2\}$

LIMITATIONS OF SLOT ENTROPY

- $\widetilde{H}_\Gamma(S^k)$ ignores the regularities that exist in S^k
- Consider a task set $\Gamma = \{\tau_1, \tau_2\}$



schedules that contain all possible vectors

$$\widetilde{H}_\Gamma(S_1) = \widetilde{H}_\Gamma(S_2)$$

LIMITATIONS OF SLOT ENTROPY

- $\widetilde{H}_{\Gamma}(S^k)$ **ignores the regularities that exist in** S^k
- **Consider a task set** $\Gamma = \{\tau_1, \tau_2\}$

S_1

τ_1	τ_2	τ_1	τ_2	τ_1
τ_2	τ_1	τ_2	τ_1	τ_2

S_2

τ_2	τ_2	τ_2	τ_2	τ_2
		⋮		
τ_1	τ_1	τ_2	τ_1	τ_1
τ_1	τ_1	τ_1	τ_2	τ_2
τ_1	τ_1	τ_1	τ_1	τ_2
τ_1	τ_1	τ_1	τ_1	τ_1

schedules that contain all possible vectors

$\widetilde{H}_{\Gamma}(S_1) = \widetilde{H}_{\Gamma}(S_2)$

while $H(S_2) > H(S_1)$

CANNOT CAPTURE THE RANDOMNESS CORRECTLY

APPROXIMATE SCHEDULE ENTROPY

- **Given,** $\mathbf{K}$ hyper-periods

  Each hyper-period has length of $\mathbf{L}$

APPROXIMATE SCHEDULE ENTROPY

- **Given,**

  $\mathbf{K}$ hyper-periods

  Each hyper-period has length of $\mathbf{L}$

- **Define**

$$X_t^k(m) = [s^k_{t \bmod L}, s^k_{(t+1) \bmod L}, \dots, s^k_{(t+m-1) \bmod L}]$$

the interval of size m starting from slot t at k-th hyper period

$$C_t^k = \frac{1}{K}\left|\{k' : \delta\left(X_t^k(m), X_t^{k'}(m)\right) \le \pi, 1 \le k' \le K\}\right|$$

normalized dissimilarity of intervals from slot t across K hyper-periods against interval at k-th hyper-period

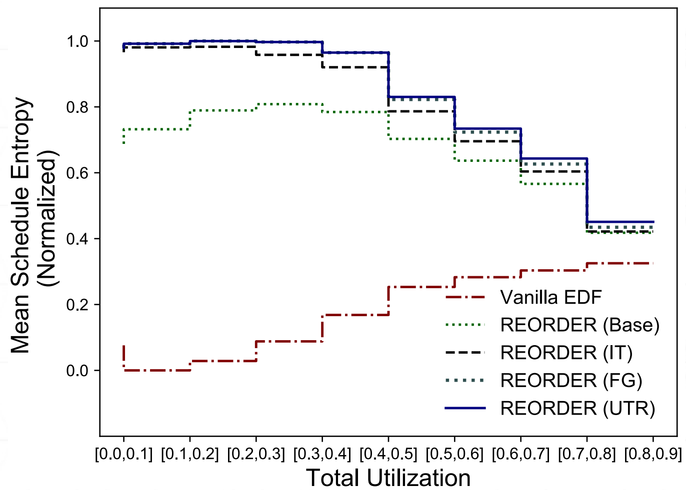
APPROXIMATE SCHEDULE ENTROPY

- **Estimated entropy of the slot t**

$$\eta_t = -\frac{1}{K}\sum_{k=1}^{K} \log_2 C_t^k$$

- **Approximate Entropy of the schedule S^k**

$$\hat{H}(S^k, m, \pi, K) = \frac{1}{m}\sum_{t=0}^{L-1} \eta_t$$



2250 tasksets tested for each scheme

L=100, K=100

m=0.35L, π=0.1L

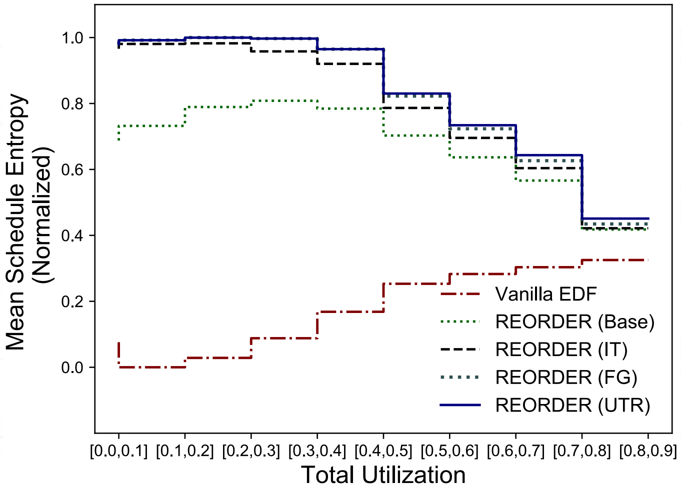
APPROXIMATE SCHEDULE ENTROPY

- **Estimated entropy of the slot t**

$$\eta_t = -\frac{1}{K}\sum_{k=1}^{K}\log_2 C_t^k$$

- **Approximate Entropy of the schedule S^k**

$$\hat{H}(S^k, m, \pi, K) = \frac{1}{m}\sum_{t=0}^{L-1}\eta_t$$



2250 tasksets tested for each scheme

L=100, K=100

m=0.35L, π=0.1L

Randomized EDF Schedules have significantly higher entropy than vanilla EDF, even at higher utilizations