A Workload-Aware DVFS Robust to Concurrent Tasks for Mobile Devices

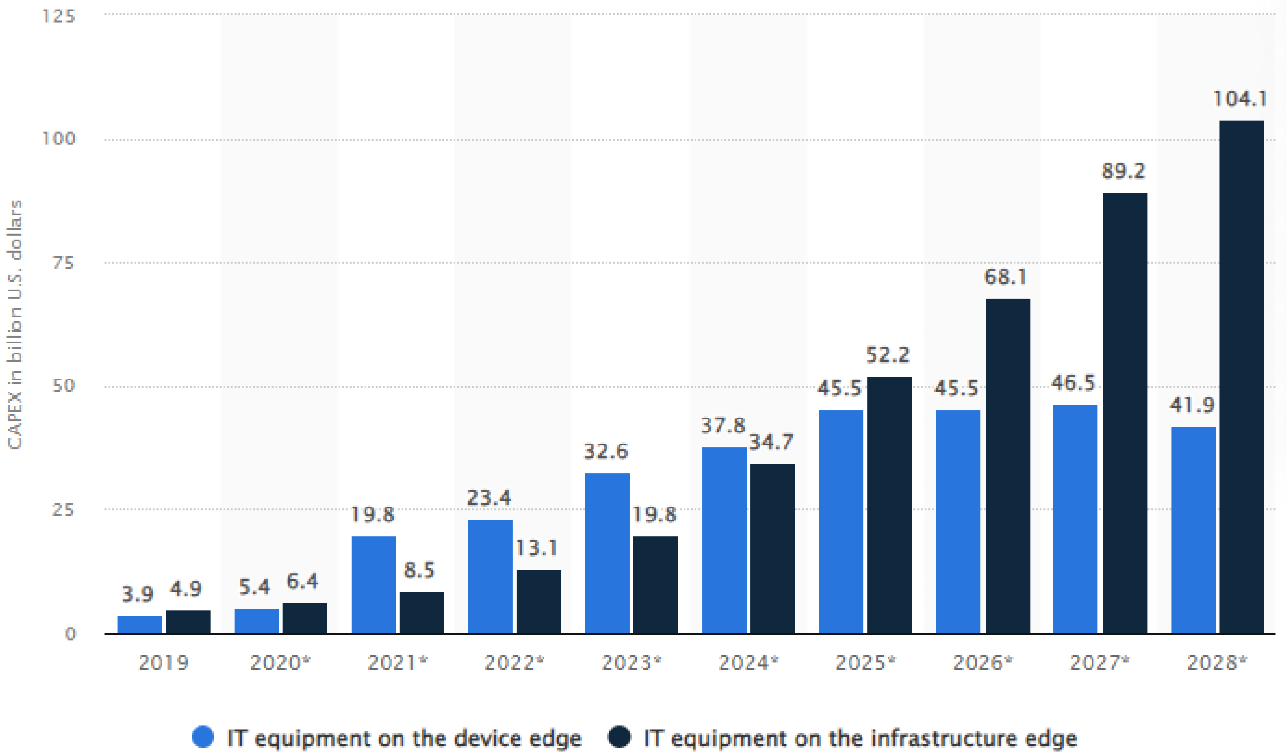
Chengdong Lin[1,2], Kun Wang[1], Zhenjiang Li[1], Yu Pu[2]

City University of Hong Kong[1], Alibaba DAMO Academy[2]

Motivation - Power and Thermal Management

Mobile Edge Devices

https://www.statista.com/statistics/1104591/worldwide-edge-computing-capex/



Overheating

Computing **power drops** significantly e.g., 40~50% reduction

Compromising device and application **reliability**

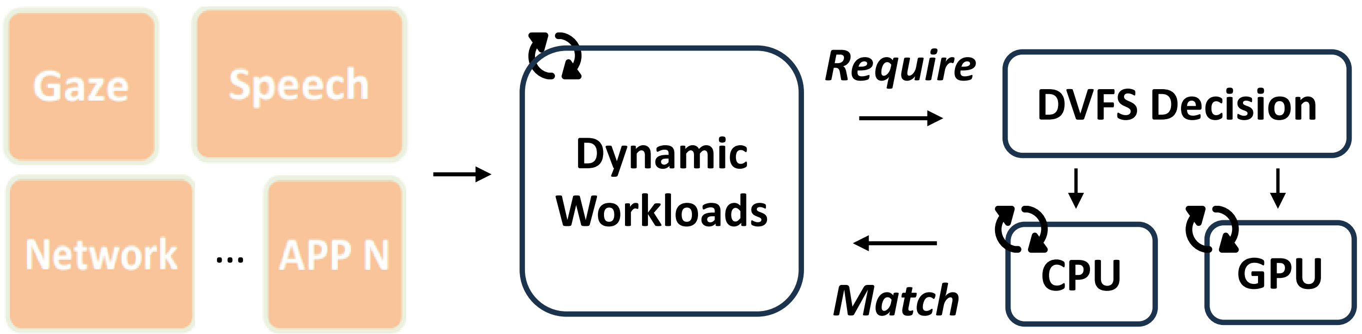
DVFS (Dynamic Voltage and Frequency Scaling) - OS-level Tool

$$P \propto f^3$$

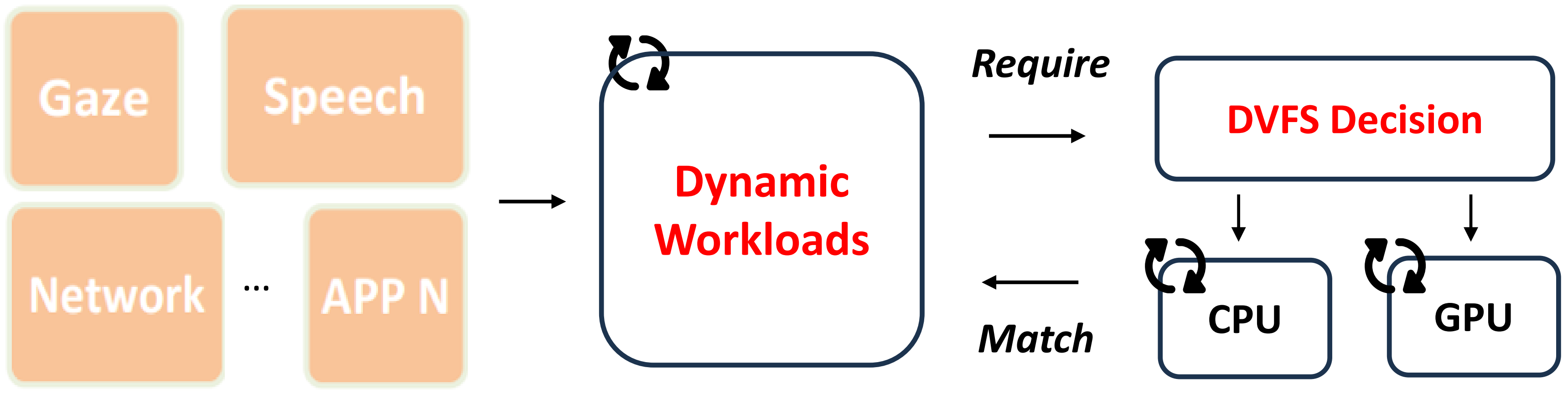
Power Consumption ***Processor Frequency***

Frequency Table

A15 Cluster of Exynos 5422		A7 Cluster of Exynos 5422	
Frequency(KHz)	Voltage(uV)	Frequency(KHz)	Voltage(uV)
2000000	1250000	1400000	1250000
1900000	1250000	1300000	1250000
1800000	1250000	1200000	1250000
1700000	1250000	1100000	1250000
1600000	1250000	1000000	1100000



Questions



Q1. How to represent dynamic workloads?

Q2. How to make suitable frequency decision?

Prior Arts: Application Specific

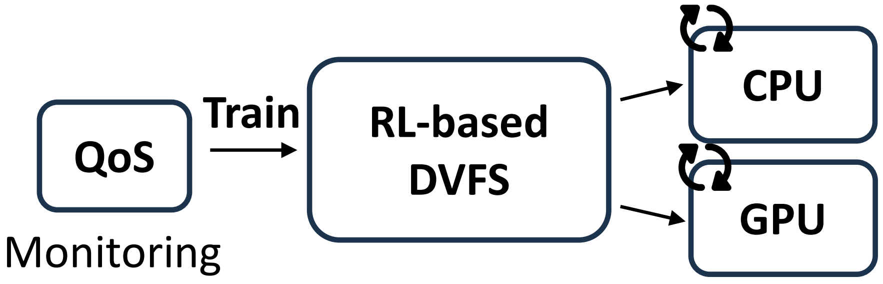
Application based Solutions:

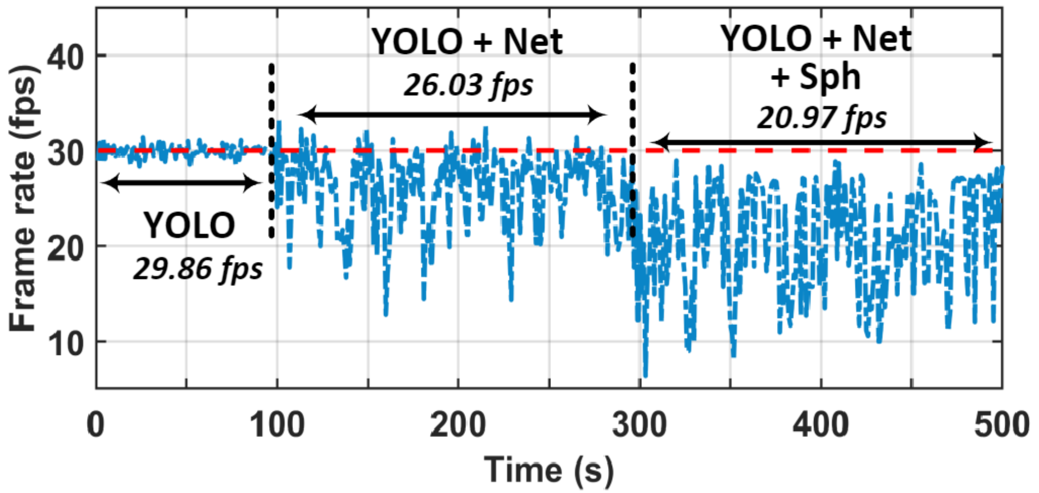
- **QoS as Indicator**

  QoS: frame rate, response latency, ….

- **Rationale: keep QoS at a desired level**

  e.g., fps = 30





Multitasks have multi QoS

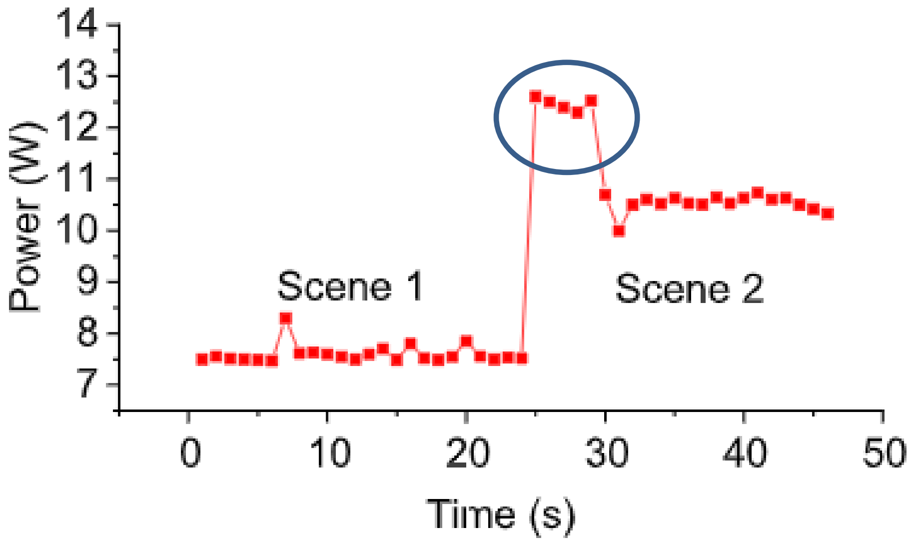
Prior Arts: Application Agnostic

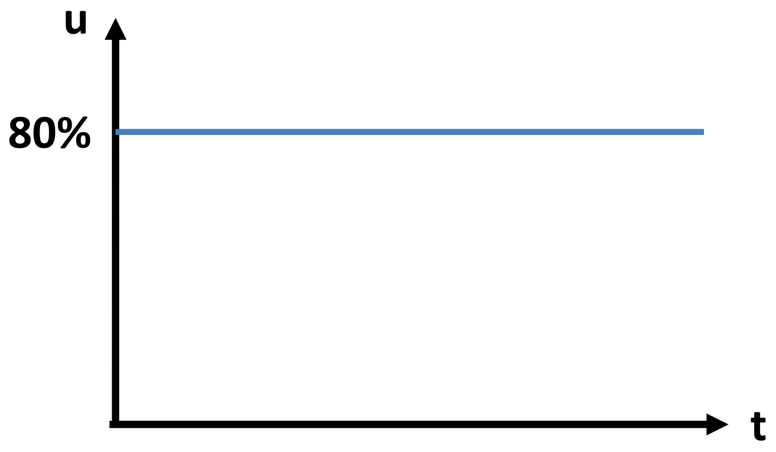
General OS Solutions:

- **Utilization as Indicator**

$$u = \frac{T_{busy}}{T_{win}}$$

- **Rationale: keep utilization at a desired level e.g. u = 80%**





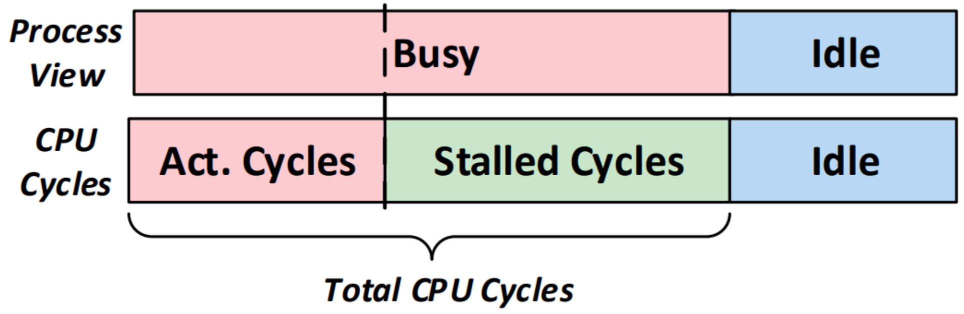
Not Accurate, why?

$$\mathtt{freq}_{\mathtt{next}} = \mathtt{1.25} * \mathtt{freq}_{\mathtt{curr}} * \mathtt{util}$$

Representation of Workload-Awareness context

Core Utilization Formula

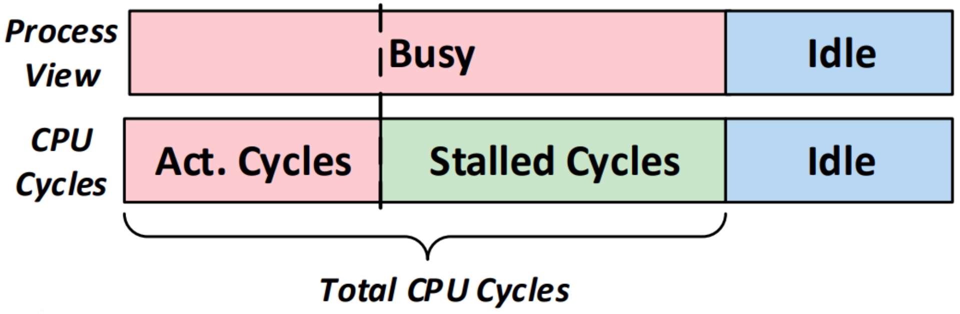
$$u = \frac{T_{busy}}{T_{win}}$$

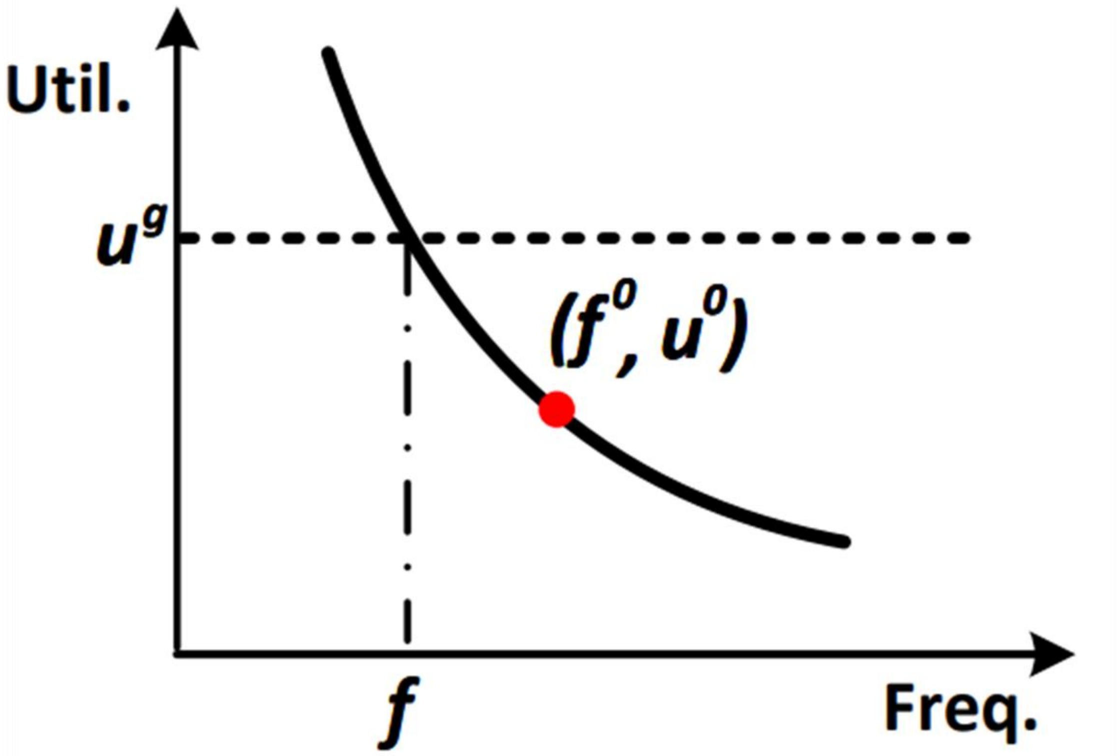


Representation of Workload-Awareness context

Core Utilization Formula

$$u = \frac{T_{busy}}{T_{win}}$$





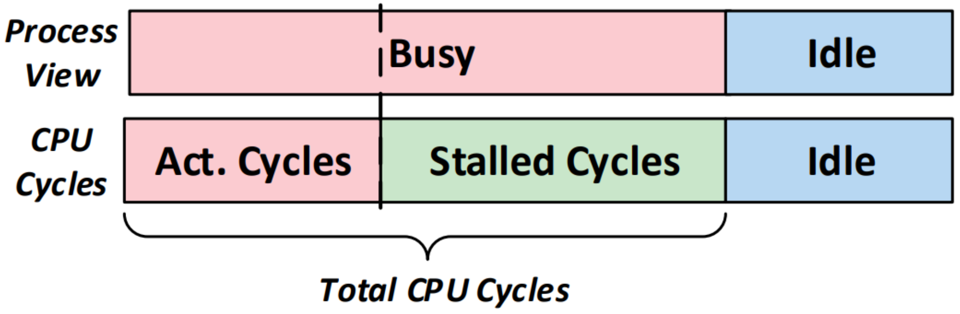
Representation of Workload-Awareness context

Core Utilization Formula

$$u = \frac{T_{busy}}{T_{win}} = \frac{T_{act} + T_{sta}}{1} = \beta \times \frac{W_{act}}{f} + T_{sta}$$

$$u = a/f + b$$

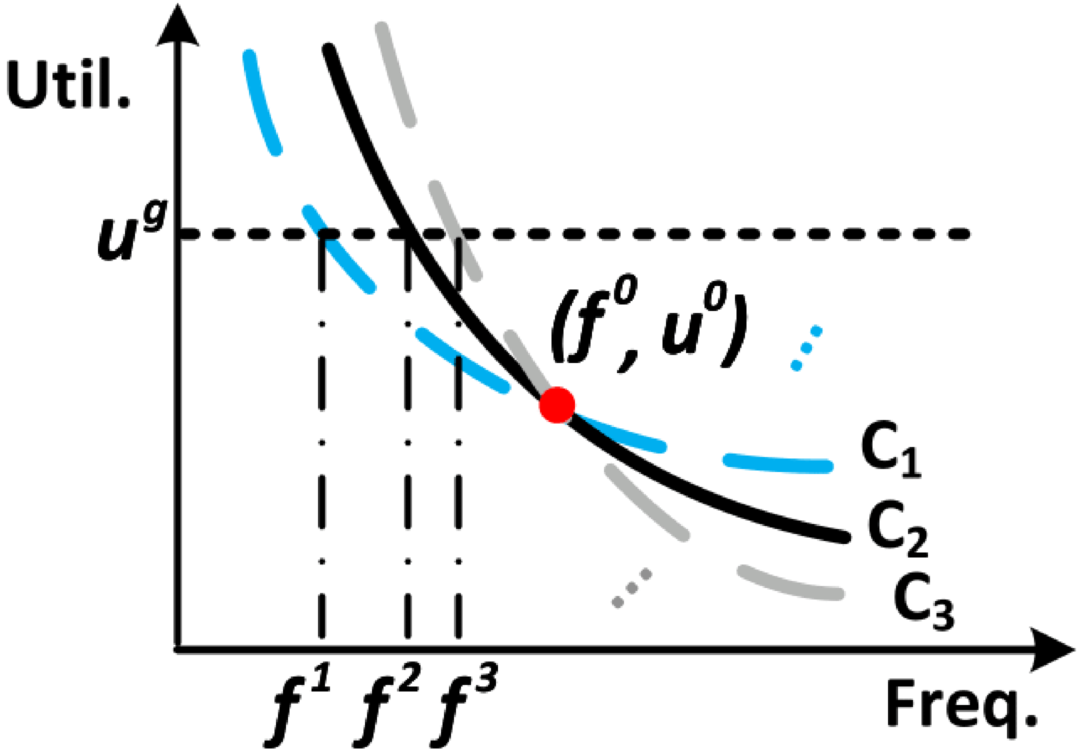
workload-awareness context



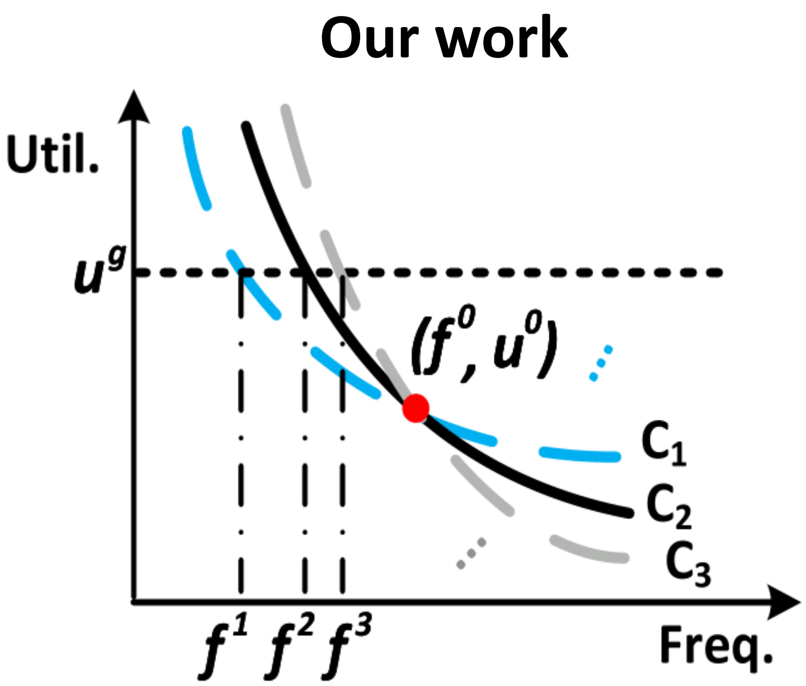
Different <a,b>

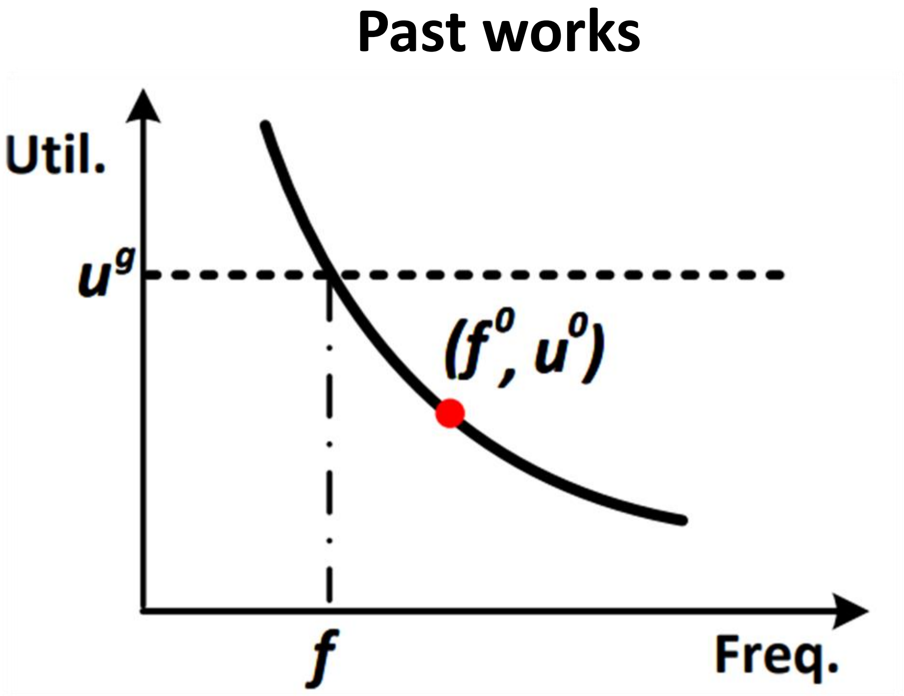
↓

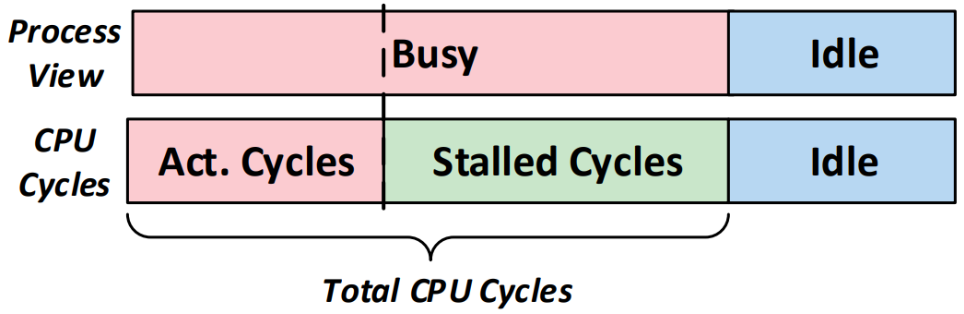
Different curves



Representation of Workload-Awareness context

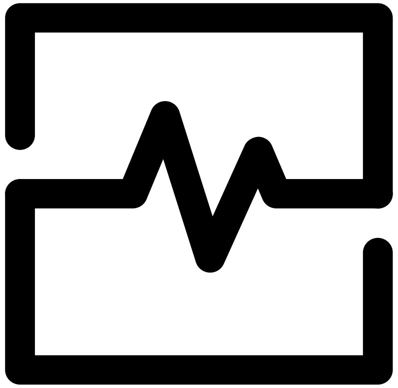




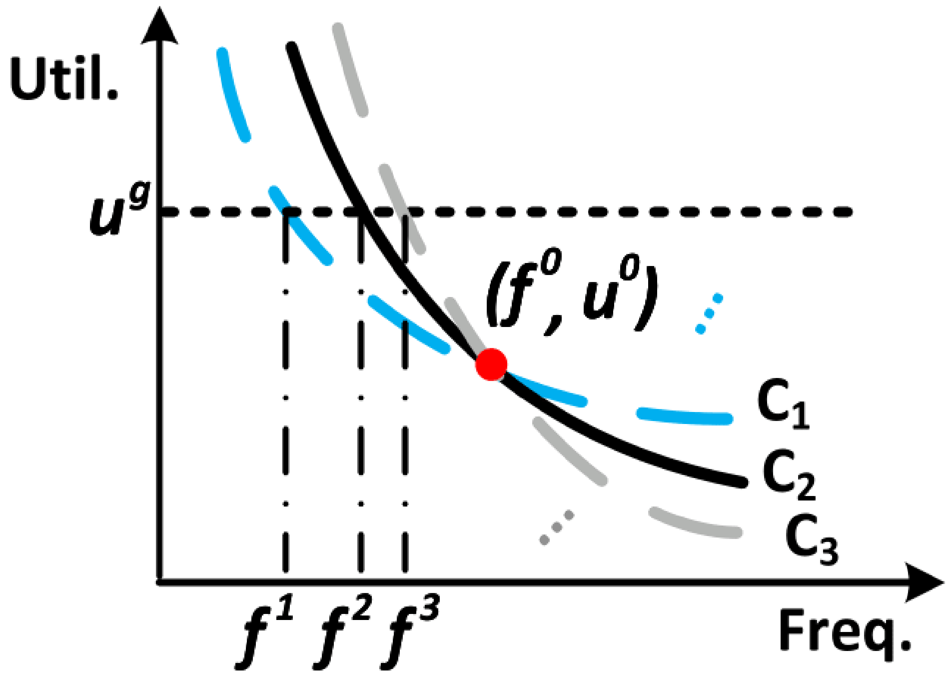


Representation of Workload-Awareness context

$$u = \underline{a}/f + \underline{b}$$



System Workloads <a,b>



Curves

Learn Workload-Awareness context - Hardware Meta-data

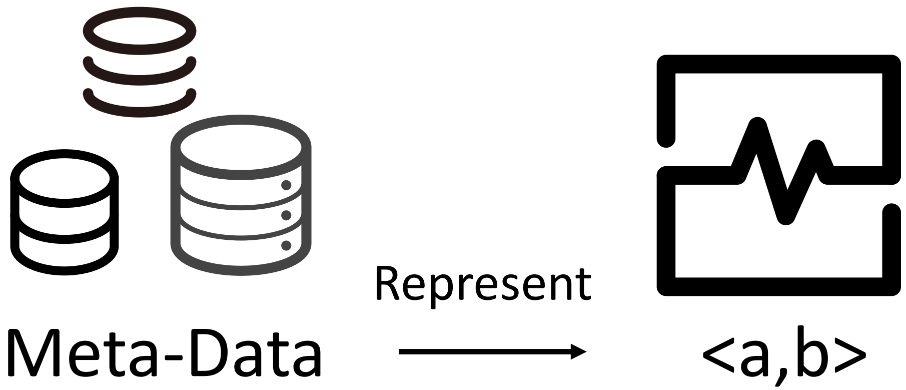
- **Hardware statistics of Processors**

  - CPU\GPU utilization (active & stalled cycles)

  - Cache hits/misses

  - Frequency

  - Temperature



- **No Labeling effort**

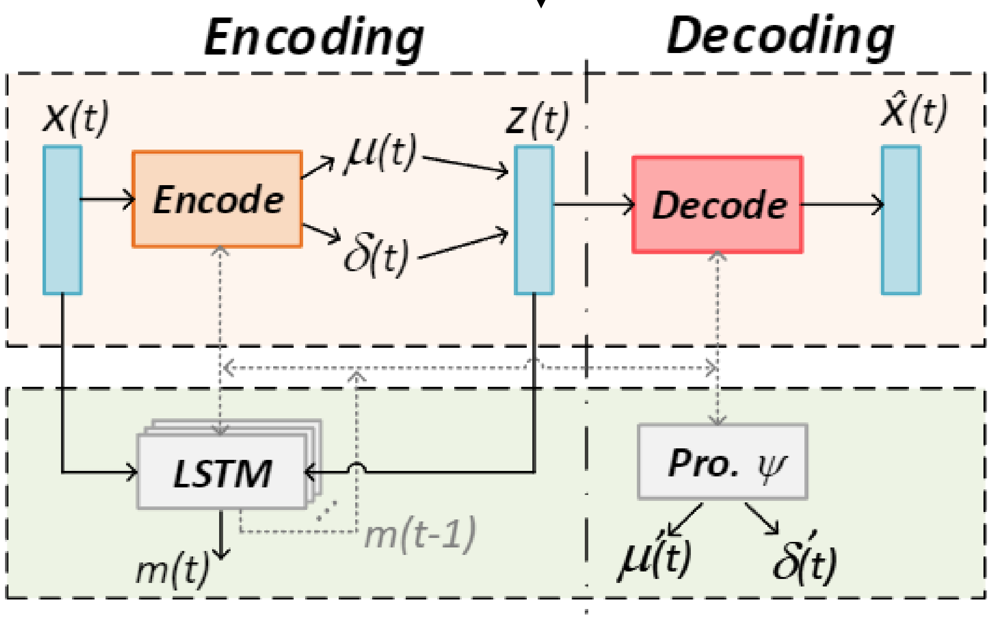
  - OS generate hardware statistics automatically

How?

Learn Workload-Awareness context - Meta-state Learner

Z(t): Intermediate latent variable

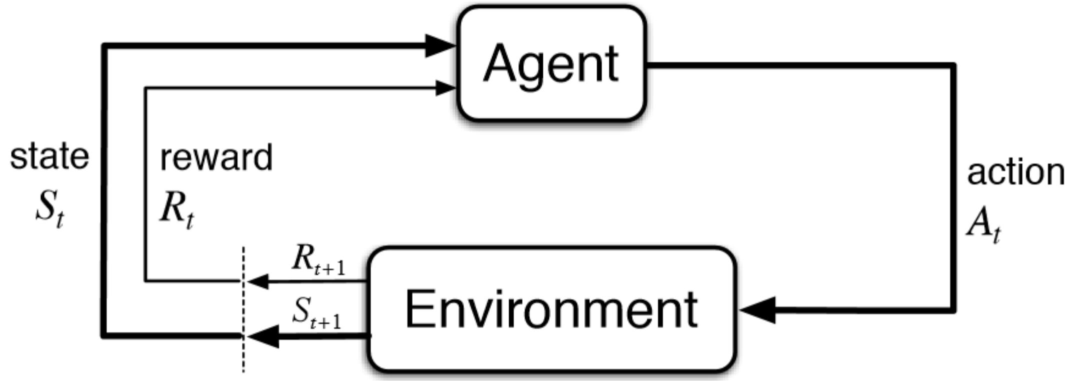
X(t): Hardware raw observation data



m(t) : Using LSTM to capture the temporal feature

RL-based solution

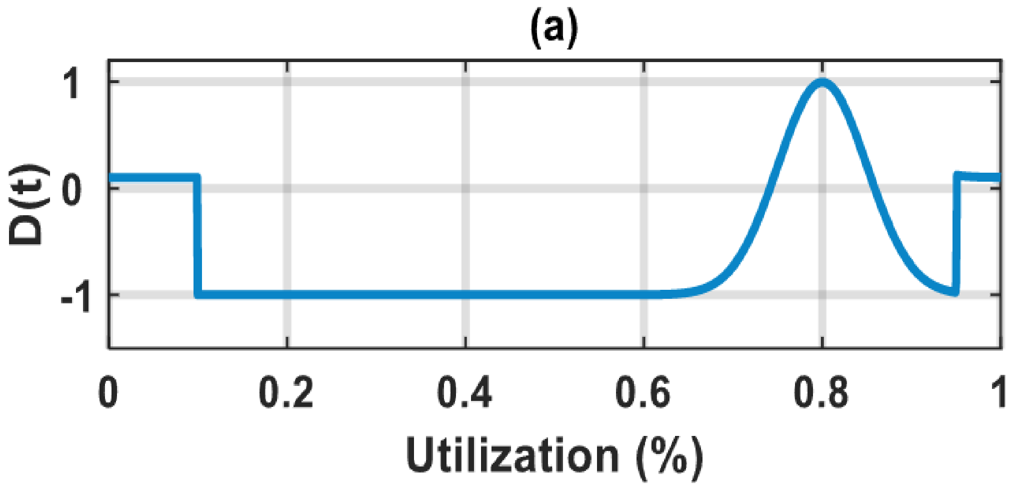
$$(P): \quad \min \ \frac{1}{T\times M} \sum_{t=1}^{T} \sum_{i=1}^{M} |u_i(t) - u_i^g|,$$
$$s.t. \qquad f_i^{min} \le f_i(t) \le f_i^{max}, \ \forall i, t,$$
$$c_i(t) \le c_i^{thermal}, \ \forall i, t.$$



State: Meta-State

Action: Select the Frequency

Reward Function: Need to Design



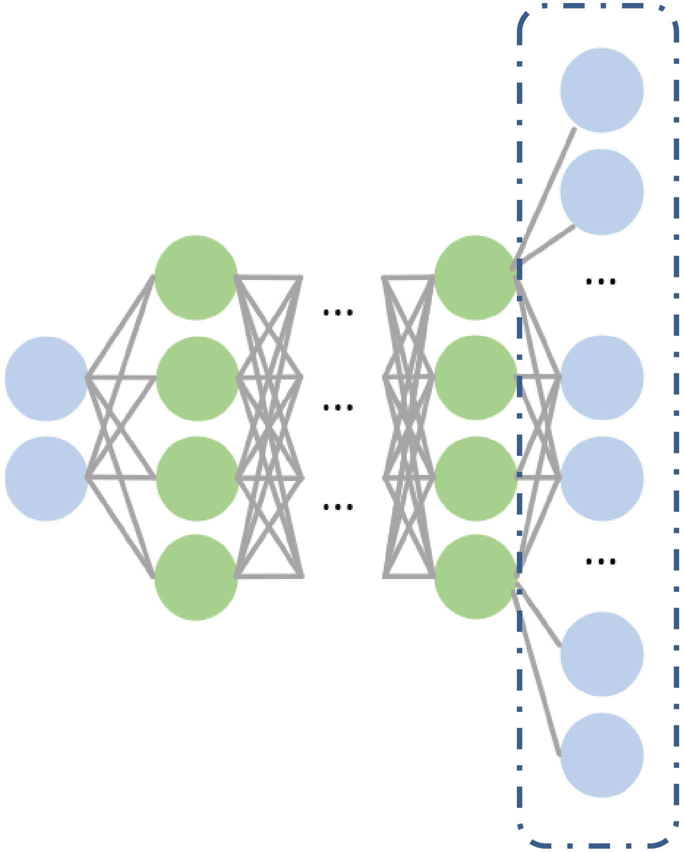
Make DVFS decision - Large action space challenge

Frequency Action Search Space:

12 *	12 *	13 =	1872
CPU (big)	CPU (little)	GPU	

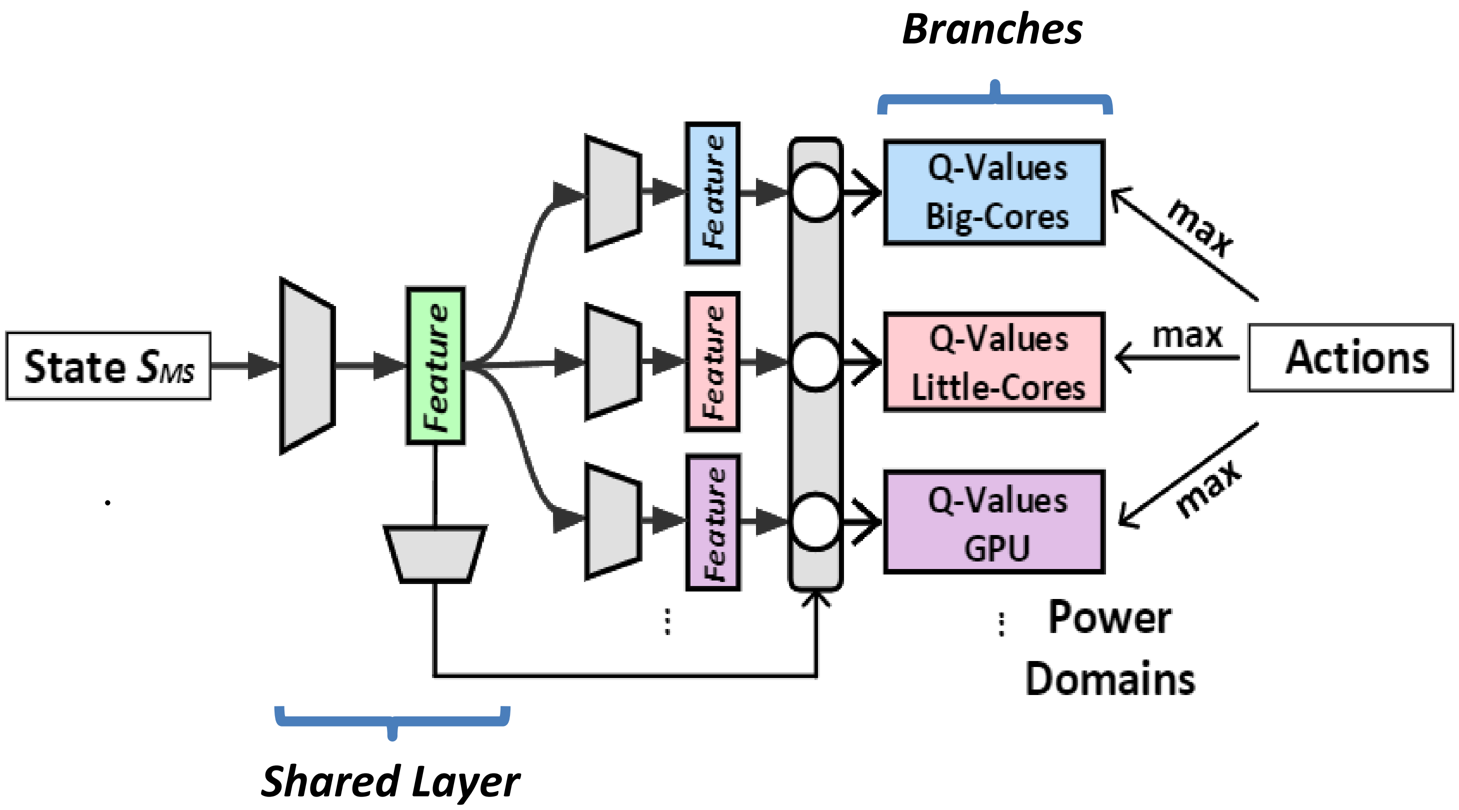
A15 Cluster of Exynos 5422		A7 Cluster of Exynos 5422	
Frequency(KHz)	Voltage(uV)	Frequency(KHz)	Voltage(uV)
2000000	1250000	1400000	1250000
1900000	1250000	1300000	1250000
1800000	1250000	1200000	1250000
1700000	1250000	1100000	1250000
1600000	1250000	1000000	1100000
1500000	1100000	900000	1100000

...



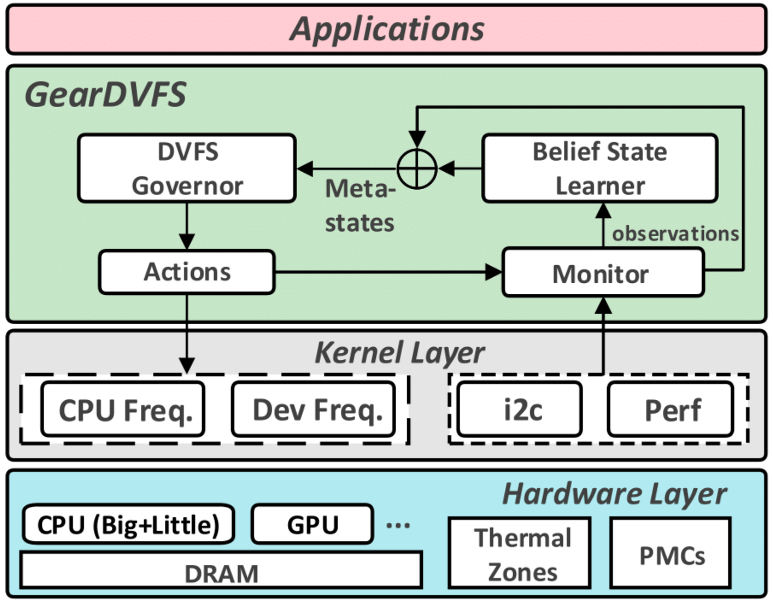
Make DVFS decision

Action Branching based Deep Q-Network

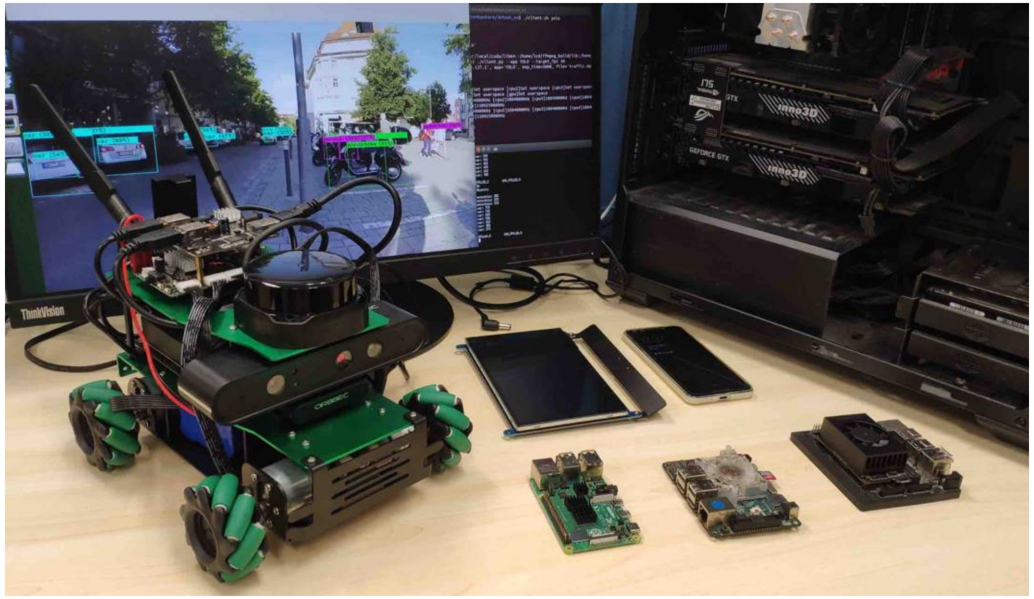


Search Space: **12 + 12 + 13 = 37**
BIG Little GPU

Implementation



Proposed DVFS Framework



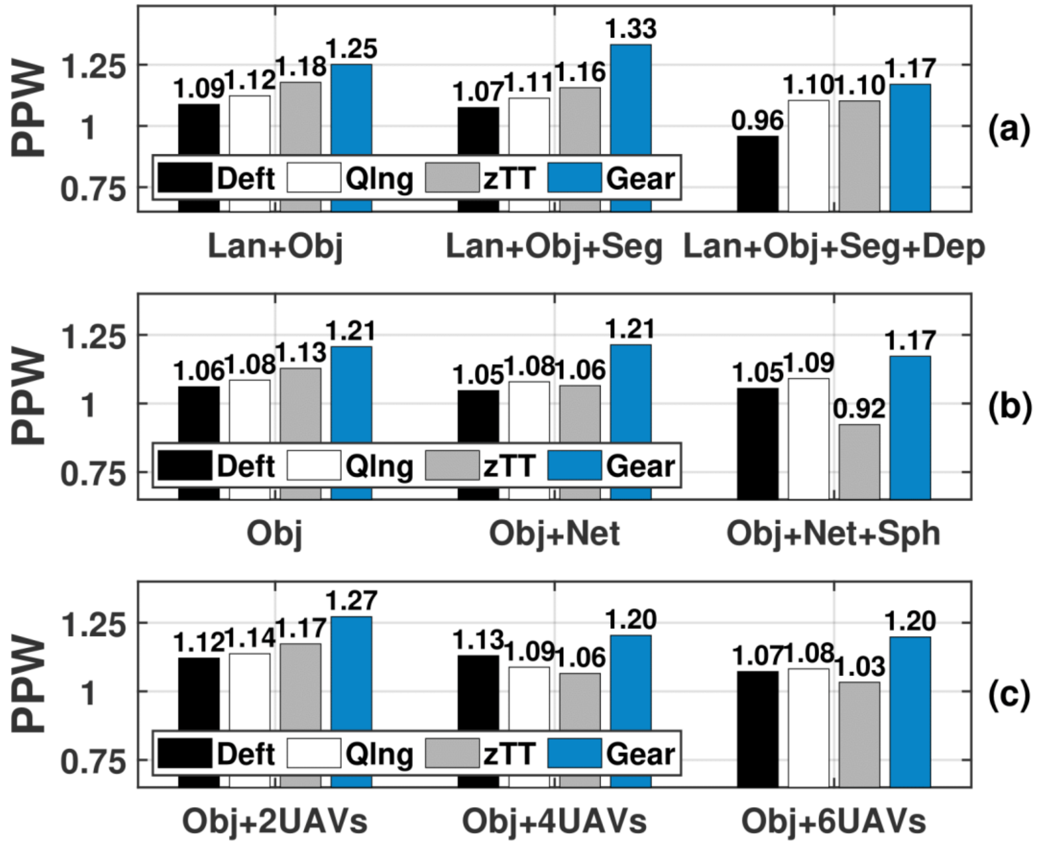
Test Bed

1. **Self-driving(4 tasks):** lane detection, object detection, segmentation, depth estimation
2. **Robot (3 tasks):** object detection, video uploading, speech recognition
3. **UAV ground station(2 tasks):** object detection, multi-stream video receiving
4. **Smartphone APPs(3 APPs):** Tik Tok, PUBG, Zoom

Scenarios

Evaluation Results - Overall Performance

Overall Performance (PPW: performance per watt)

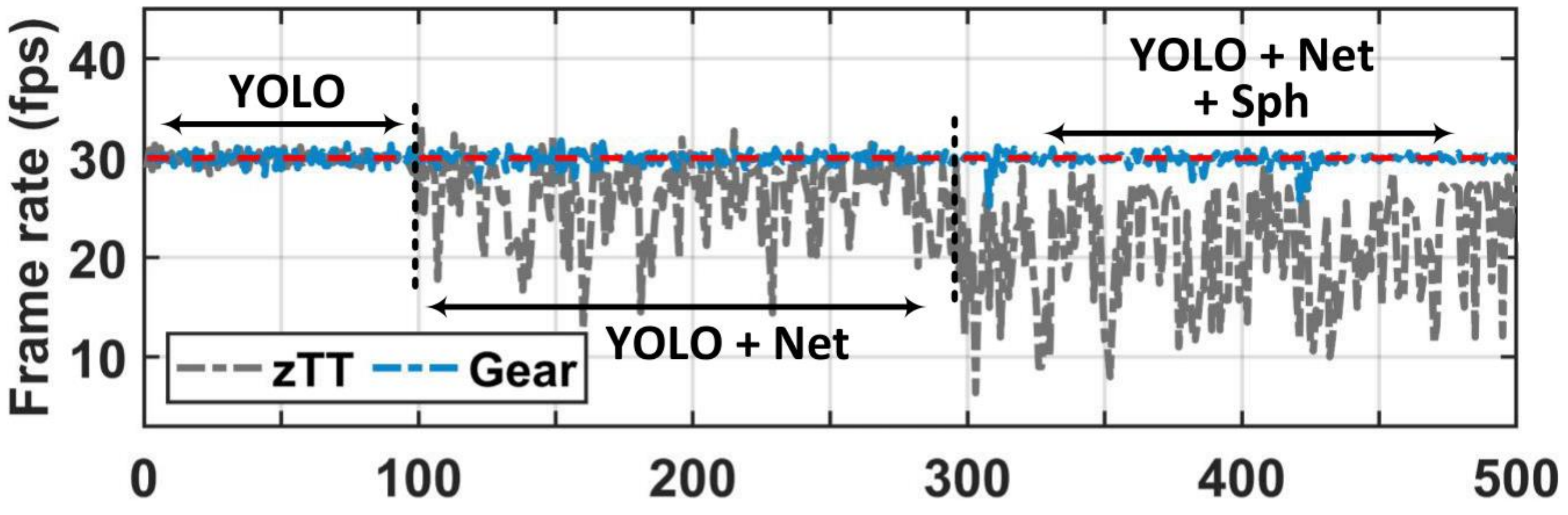


23.9%~26.9%

Power Efficiency

"zTT: Learning-based DVFS with Zero Thermal Throttling for Mobile Devices." ACM MobiSys 2021

Evaluation Results - Compare with the SOTA

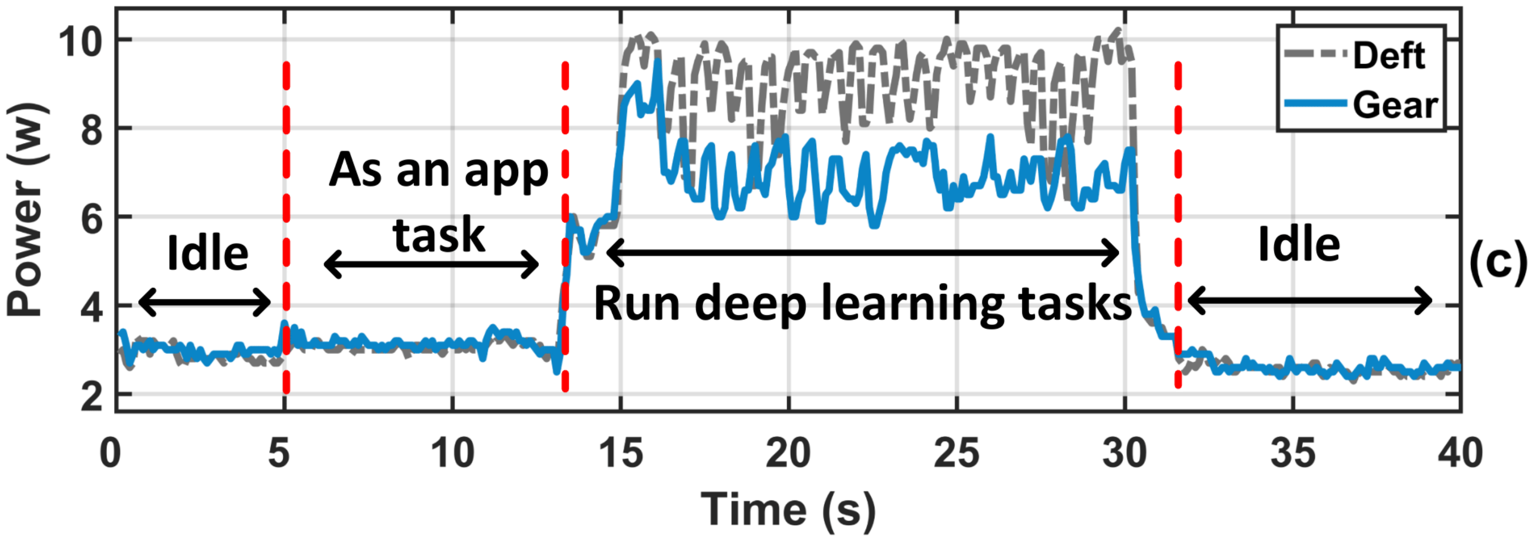


Performance of concurrent workloads

"zTT: Learning-based DVFS with Zero Thermal Throttling for Mobile Devices." ACM MobiSys 2021

Evaluation Results - System Overhead

Overhead: ~100 mw
Reduction: ~2W



System Overhead on Jetson NX

Conclusion

- ❑ We explored recent DVFS solution is not effective, introduced new hardware based DVFS metrics.
- ❑ We proposed a DVFS framework that could learn dynamic workload contexts in multitasks scenarios.
- ❑ We developed a prototype DVFS system to demonstrated its efficiency on different embedded platforms.

Thanks!

Scan to visit the project website,
or type GearDVFS.github.io