What is still missing in load tracking ?

Vincent Guittot

OSPM'18

16th April 2018

LEADING
COLLABORATION
IN THE ARM
ECOSYSTEM

Agenda

- What happened since last year ?
- What still remains ?
- What next ?

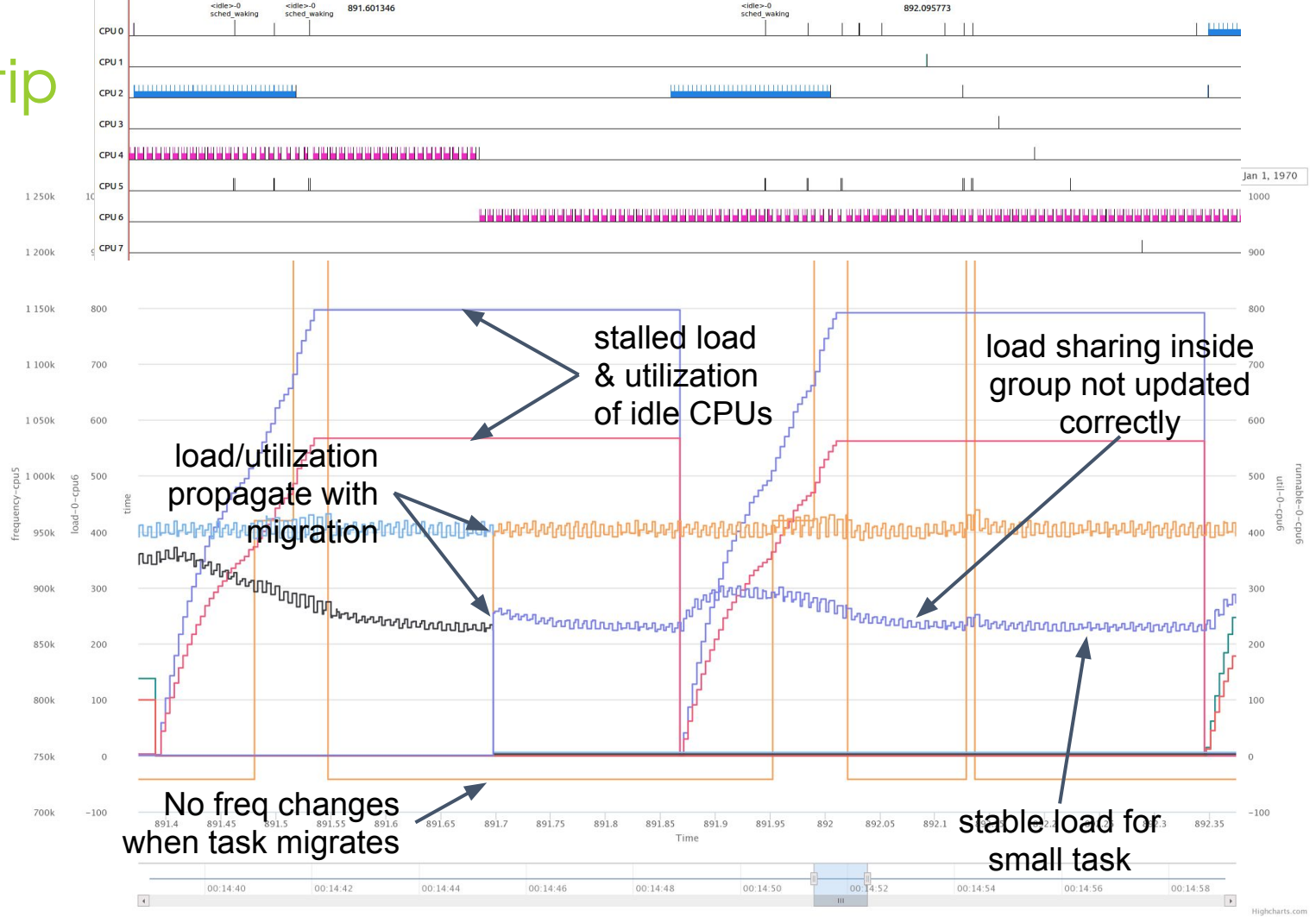
What happened since last year ?

- New propagation mechanism
  - Including runnable load of sched_group
- Deadline bandwidth
  - Implemented deadline “utilization”
  - Implemented invariance and OPP selection for SCHED_DEADLINE
- Blocked idle
  - Decay blocked load and utilization
- Util est
  - Estimate final CFS utilization level

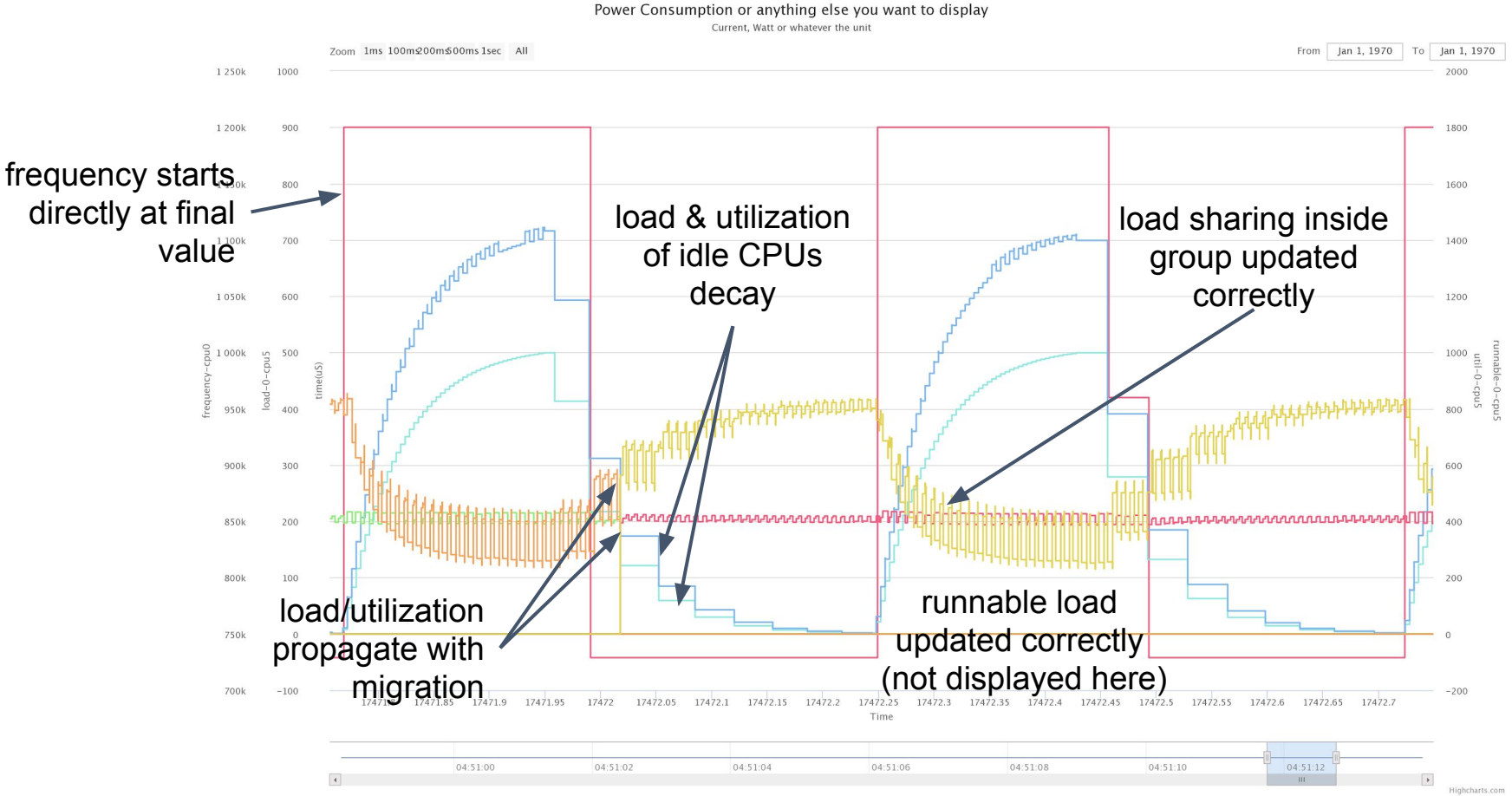
v4.9


CPU 0
CPU 1
CPU 2
CPU 3
CPU 4
CPU 5
CPU 6
CPU 7
190.738481
191.237895
Jan 1, 1970
stalled load
& utilization
of idle CPUs
no load/utilization
propagation with
migration
unstable load
for small task
freq decreases when
task migrates
load sharing inside
group not updated
correctly
frequency-cpu5
load-0-cpu0
time
util-0-cpu0
runnable-0-cpu0
Time

Last year tip



tip/sched/core

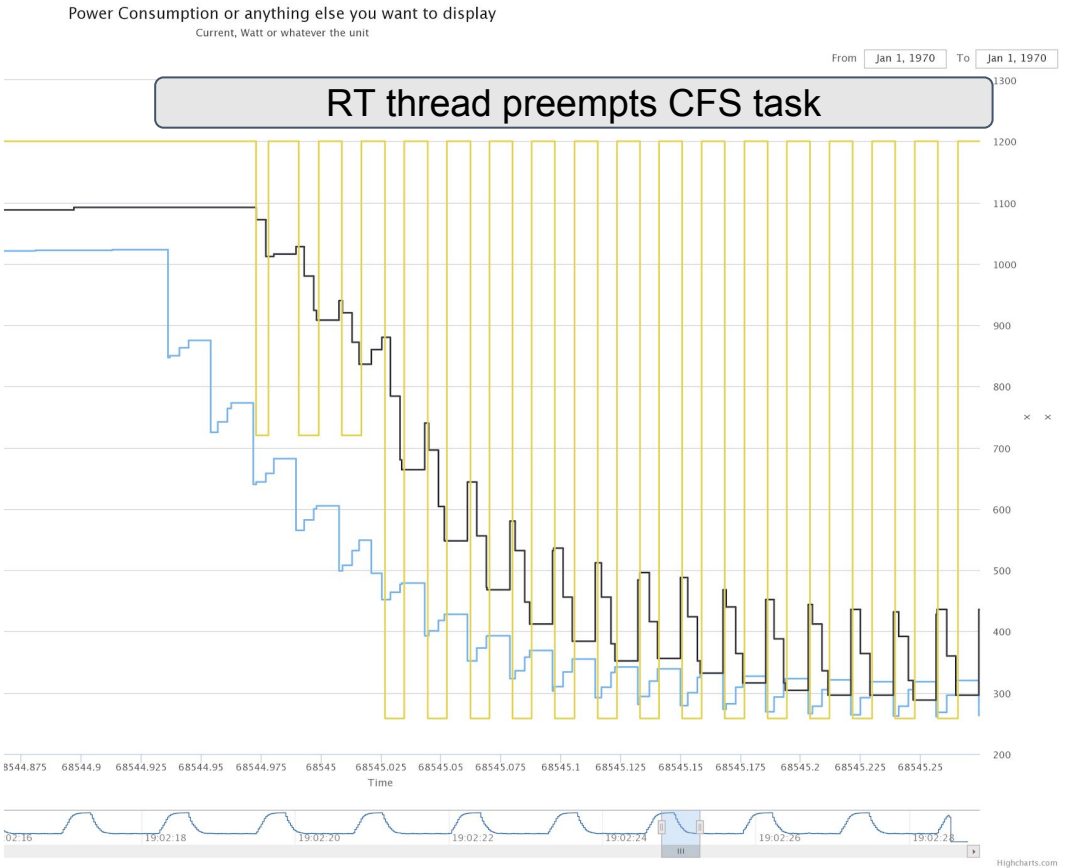


What still remains ?

- CFS' stolen time
  - frequency drop
- System/CPU utilization
  - Estimate the whole system utilization
- Scale invariance
  - Load inaccuracy

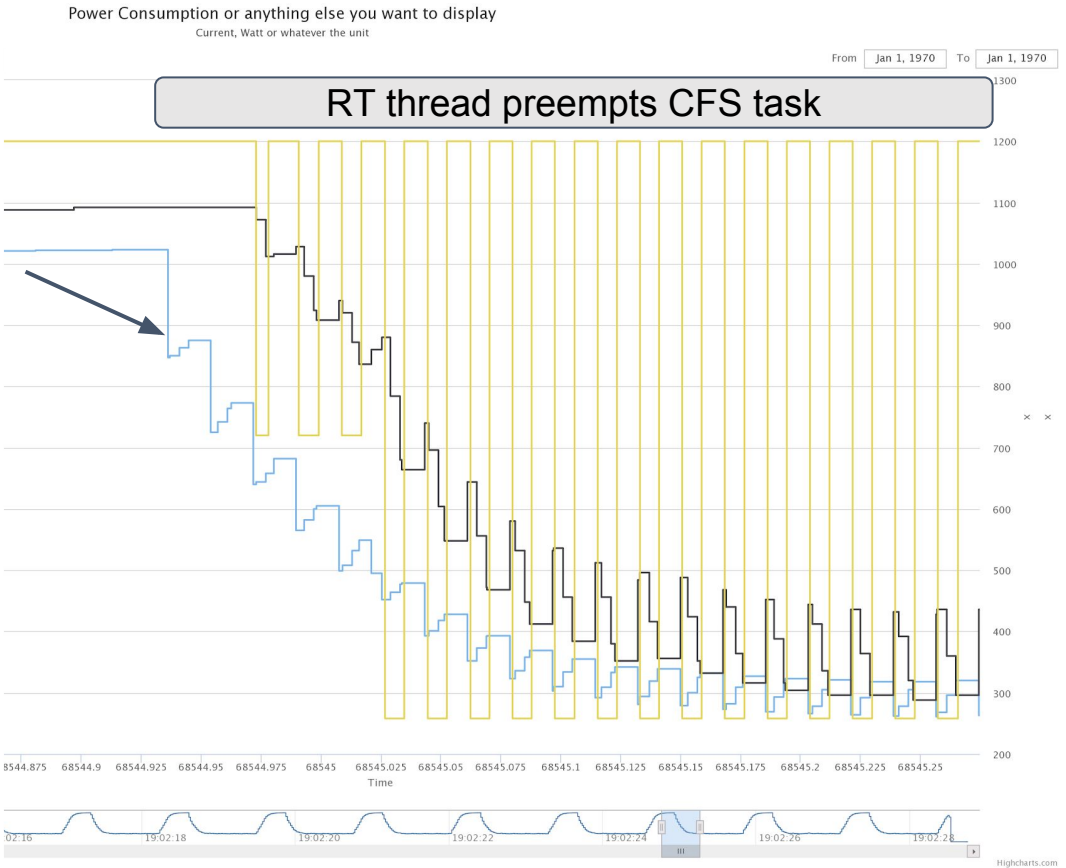
CFS' stolen time

- Case of a CFS always running task
  - Preempted by a RT task



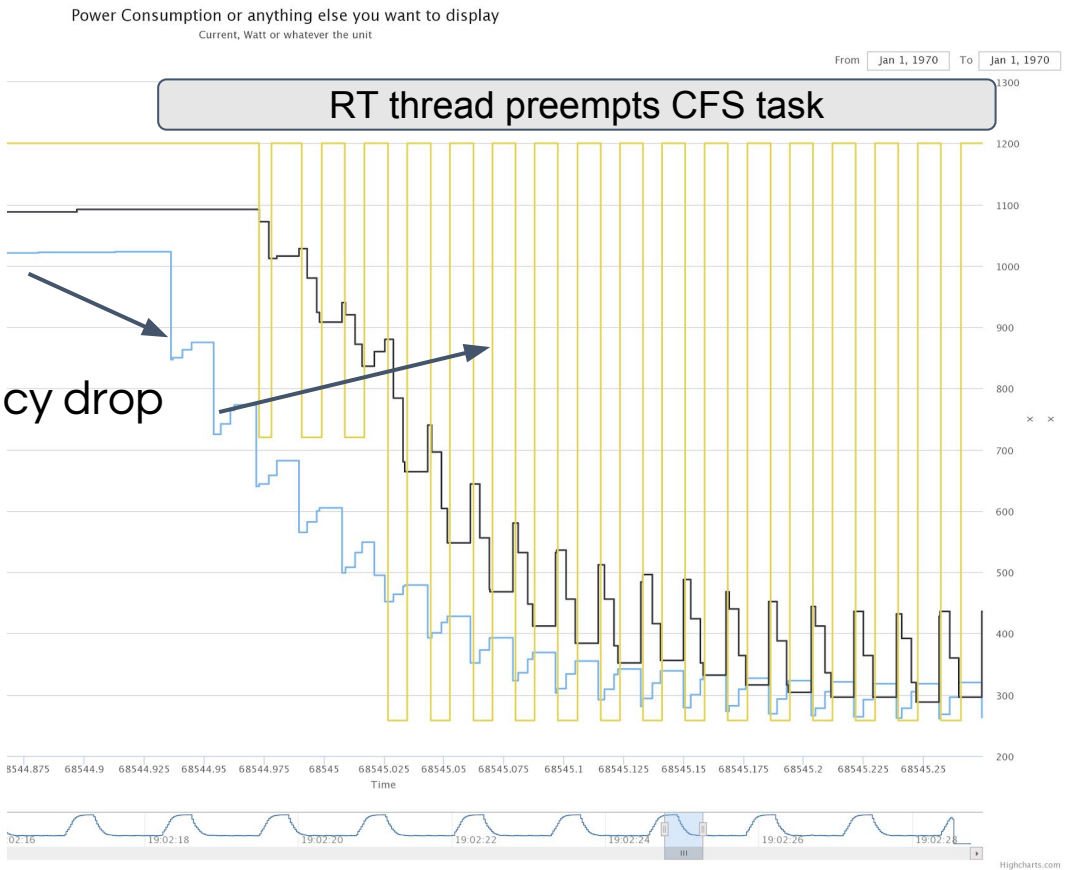
CFS' stolen time

- Case of a CFS always running task
  - Preempted by a RT task
- util_avg reflects real running time
  - Utilization decreases with preemption



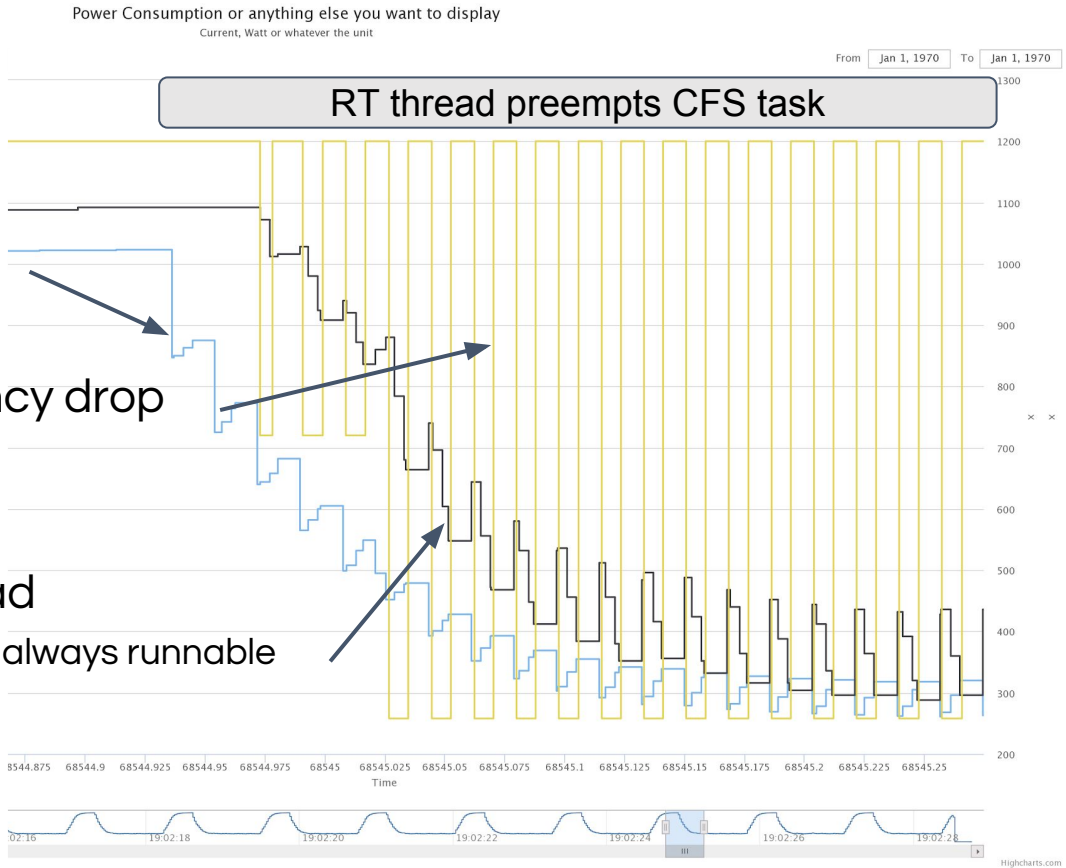
CFS' stolen time

- Case of a CFS always running task
  - Preempted by a RT task
- util_avg reflects real running time
  - Utilization decreases with preemption
- Lower utilization generates frequency drop
  - CFS task runs a low frequency



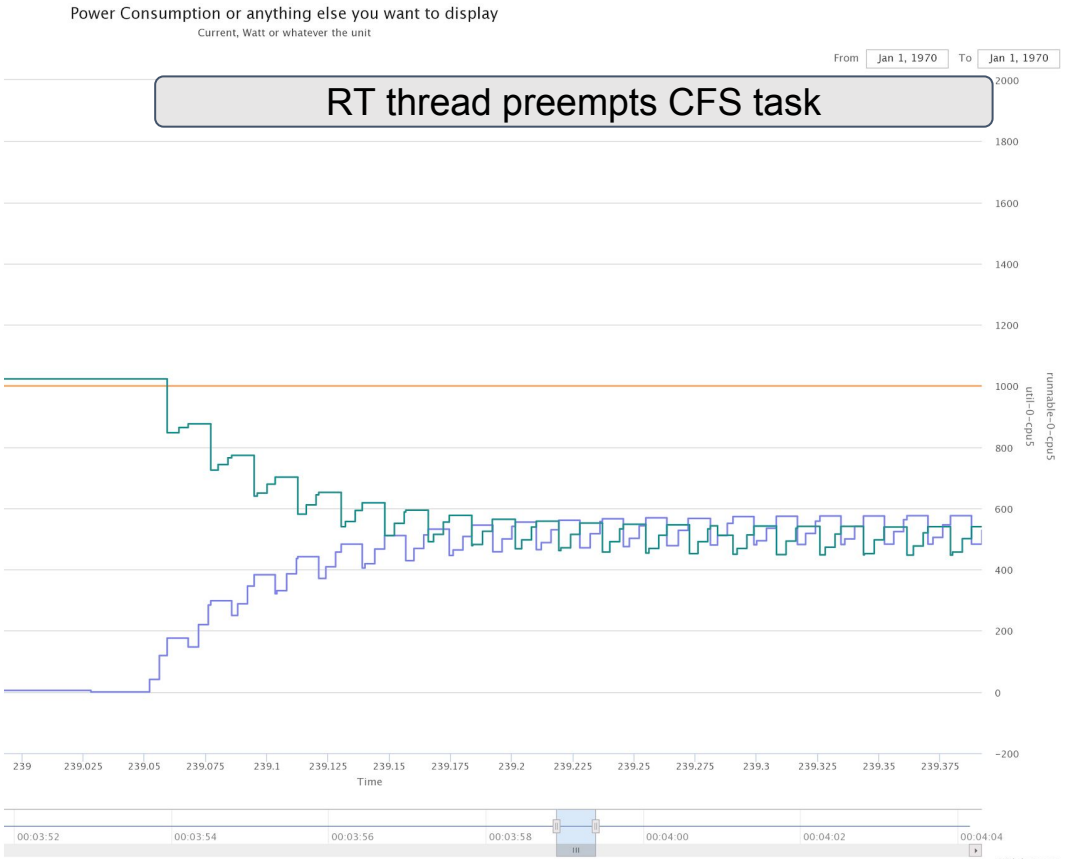
CFS' stolen time

- Case of a CFS always running task
  - Preempted by a RT task
- util_avg reflects real running time
  - Utilization decreases with preemption
- Lower utilization generates frequency drop
  - CFS task runs a low frequency
- Lower frequency decreases cfs' load
  - Load is not at max whereas CFS task is always runnable



CFS' stolen time

- Track RT utilization
  - Add it to system utilization
- Do the same for other activities ?
  - Interruption

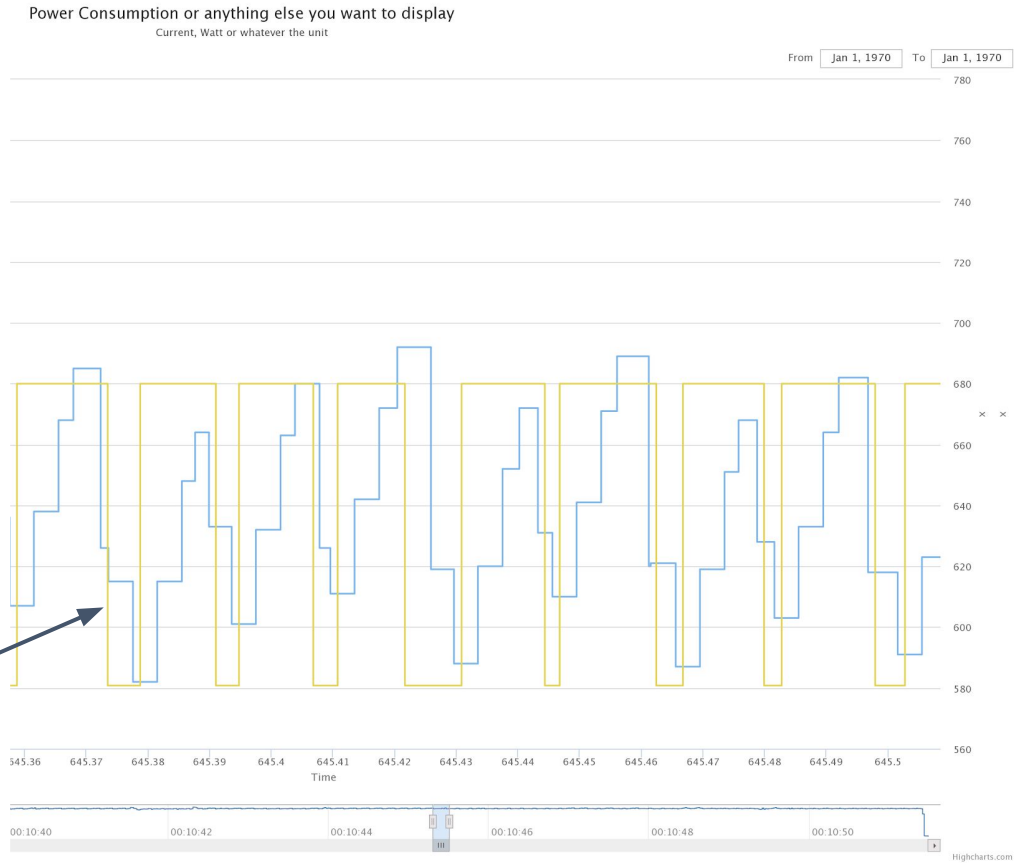


System/CPUs utilization

- How to define system/CPU utilization level ?

System/CPUs utilization

- Periodic CFS task
  - Frequency decreases to min OPP when CPU becomes idle
  - Need to wake up a CPU for that (slow path)
  - Need to go back to previous OPP at wake up
  - No power improvement because cluster Off idle state



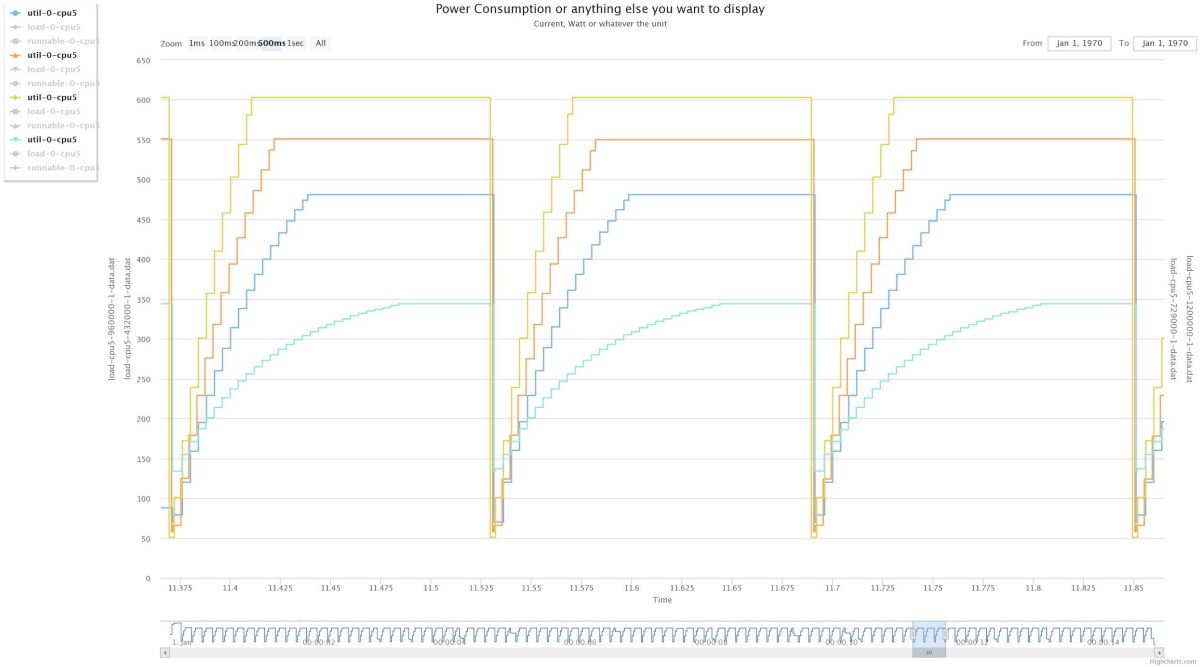
frequency switches at each idle transition

System/CPUs utilization

- How to define system/CPU utilization level ?
- Which metrics should we use ?
  - Number of running tasks
  - rq 's utilization (which included blocked utilization)
  - "Runnable" utilization
- Avoid under estimated utilization
  - preemption
- Avoid useless frequency transition
  - Useless freq drop

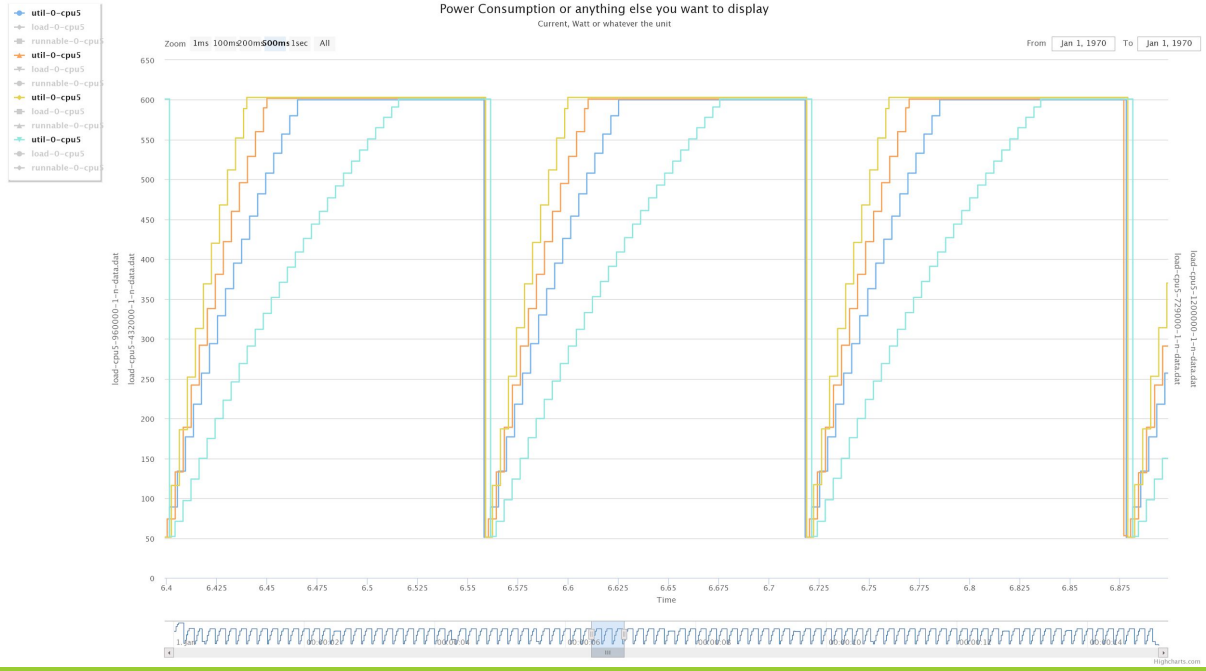
Scale invariance

- Current invariance
  - Based on run time
  - weight utilization and load
- Simple implementation
- Cap utilization and load
- Load might not reach max vc
  - see previous example



Scale invariance

- Same range of variation across freq and micro arch
- Same behavior for load and utilization



Scale invariance

- What do we want to track ?
  - Run time ?
  - Amount of work on the CPU?
- Try to track work instead of time
- Timer based or sleep based tracking activity
  - Current design is timer based

What else ?

- Cost of load tracking

Cost of load tracking ?

- More accurate load tracking means more computation
- ./perf bench sched pipe -l 100000
  - On hikey octo cores
  - Performance governor
  - Only WFI (shallowest) idle state
- Quickly tried to remove most of call to PELT related function
  - +5.66%

- Is it a problem ?
- Should we look at optimizing this ?
- Is it possible ?

Thanks