Per-task I/O boost tracking
What is I/O boost?

- Reduce devices wait time when feeding it data
- Increase throughput by reducing the time the device stays active awaiting incoming requests
- Aide for I/O-bound tasks that usually have low utilization while performing I/O requests and as such will not trigger a frequency update on their own

I/O wait task

![Diagram](image)

- 1 ms
- 0.1 ms
- 1 GHz (max)

- task utilization = 1/11
- => lower OPP

 cycle increased from 1.1 ms to 2 ms
- => ~80% decrease in throughput

20170522082154.f57cqoqverd2qajv@hirez.programming.kicks-ass.net
Current design

- Sugov: per-CPU boost driven by CFS tasks with iowait flag set
- Discrete boost values [128, 256, 512, 1024]

Boost value:

- Increase at each consecutive wake-up from I/O (< TICK_NSEC and freq update in between)
- Maintain if non-I/O update and rate-limit applies or cross CPU request not possible
- Reduce if no pending boost request during freq update
- Reset if no update for > TICK_NSEC
Issues with the current design

- Frequency spikes for sporadic I/O - increasing boost by unrelated I/O tasks
- Task migration/termination - not covered since tracking per-CPU
- I/O Devices with long response time - timing constraints (TICK_NSEC)
- Boost on CPU A can be affected by frequency updates on CPU B within same shared policy
- No real control over boosting – per-task uclamp vs boost value
- No per-task boost for task placement on asymmetric CPU capacity systems
New design proposal

- Concept: decision on I/O boosting moved from sugov to be performed on a per-task basis
- For each task that performs I/O requests track its time:
  - being blocked on I/O (sleep time)
  - between wake-ups from I/O (sleep + runnable time)
- Tracking 2 separate yet overlapping signals allows deriving conclusion on the I/O waiting pattern
- Used to decide whether further boosting is sensible (from performance and energy consumption perspective)
- With pre-defined margins and bit of coin tossing
New design proposal

Comparing:

- I/O sleep time
- I/O sleep time + runnable time

vs
• Boost value is being represented in a form of 'boost levels' with max level of 4 → this resembles sugov’s boosting with doubling boost value
• Boost levels are being max-aggregated on a rq level upon enqueueing the task
• Additional signal to retain boost between wakeups (rq level)
Discussion ...

- Boost decision points: increase / maintain / reduce ...
- Boost level max-aggregation for enqueued I/O bound task on CPU rq level
- Countermeasures for irregular patterns for I/O-bound workloads
  - Kernel worker threads that might cause disruption
- Decision making placed on the wake-up path -> additional overhead !
  - Sched vs CPUFreq gov realm