Contribution ID: **230** Type: **not specified**

# TurboSched: Core capacity Computation and other challenges

*Monday, 9 September 2019 18:00 (15 minutes)*

Turbosched is a proposed scheduler enhancement that aims to sustain turbo frequencies for a longer duration by explicitly marking small tasks that are known to be jitters and pack them on a smaller number of cores. This ensures that the other cores will remain idle, and the energy thus saved can be used by CPU intensive tasks for sustaining higher frequencies for a longer duration.

The current TurboSched RFCv4 (https://lkml.org/lkml/2019/7/25/296) has some challenges:

- **Core Capacity Computation**: Spare core capacity defines the upper bound for task packing above which jitter tasks should not be packed further into a core, else it hurts the performance of the other tasks running on that core. To achieve this we need a mechanism to compute the capacity of the cores in terms of its active SMT threads. But the computation of CPU Capacity itself if arguable and non-reliable in case of CPU hotplug events. This makes the TurboSched to have unexpected behavior in case of hotplugs or in presence of asymmetric CPU capacities. The discussion also involves the use of other parameters like nr_running with utilization to decide upper bound for task packing.
- **Interface**: There are multiple approaches to mark a small-task as a jitter. A cgroup based approach is favorable to the distros as it is a well-understood interface requiring minimal modification for the existing tools. However, the kernel community has expressed objection to this interface since whether a task is jitter or not is a task-attribute and not a task-group attribute. Further, a task being a jitter is not a resource-partition problem, which is what cgroup aims to solve. The other approach would be to define this via a sched_attribute which can be updated via an existing syscall. Finally, we can support both the approaches as discussed on LWN https://lwn.net/Articles/792471/
- **Limiting the Search Domain for packing**: On systems with a large number of CPUs, searching all the CPUs where the small-tasks should be packed can be expensive in the task-wakeup path. Hence we should limit the
domain of CPUs over which the search is conducted. In the current implementation, TurboSched uses the DIE domain to pack tasks on PowerPC, but certain architectures might prefer the LLC or the NUMA domains. Thus we need to discuss a unified way of describing the search domain which can work across all architectures.

This topic is a continuation from the OSPM talk and aims to mitigate these problems generic across architectures.

## I agree to abide by the anti-harassment policy

Yes

## I confirm that I am already registered for LPC 2019

**Primary author:** SHAH, Parth

**Presenter:** SHAH, Parth

**Session Classification:** Scheduler MC