Contribution ID: **262** Type: **not specified**

QoS Hinting APIs: If we had them, what would they actually do?!

Wednesday, 18 September 2024 18:05 (22 minutes)

At OSPM we had a number of discussions around the need for QoS APIs for applications to hint their needs for latency and throughput for SCHED_NORMAL/FAIR tasks, as opposed to the typical global tuning of scheduler knobs.

Folks seemed generally supportive of adding some sort of hinting API. However, while any hinting API will be coy and resistant to making any hard promises to userland about exactly what behavior may result from the hint, there seems to be a fair amount of fog around what we might actually do when an application hints that a certain thread would prefer lower latency response or more throughput.

Some potential actions we might take for low-latency hinting:
* When the task wakes up, allow it to preempt the currently running task
* Tweak its scheduler state so that the task's placement in the runqueue will result in it being selected sooner
* Adjust cpu placement, so that when it wakes up, it's more likely to be placed on idle cpus (though we must be careful not to pick ones in deep sleep states).
* Increase the cpu freq so running tasks can finish what they are doing allowing us to switch to our hinted task faster.
* [Other ideas?]

And for throughput focused tasks we might:
* Tweak the task's placement so it will be placed on bigger cpus
* Try to avoid other tasks preempting the hinted task, by placing woken tasks on other cpus
* Let the task run for longer slices
* More aggressively ramp up the cpufreq, by increasing the utilization estimation.
* [Other ideas?]

A big issue: the right thing to do in each case may very well depend on the hardware. So we may need some way to understand and abstract these choices. For instance: placement will need to be aware of the idle cpu wakeup latencies.

So once we've enumerated the possible actions, how do we configure which actions to take on which hardware?

Primary author: STULTZ, John (Google)

Presenter: STULTZ, John (Google)

Session Classification: Sched MC