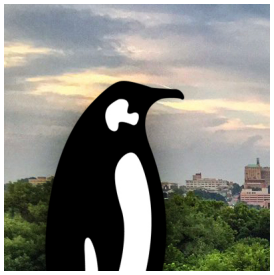

Contribution ID: **76** Type: **not specified**

Improving CPU Isolation with per-cpu spinlocks: performance cost and analysis

Monday, 13 November 2023 12:25 (15 minutes)

What do we want?
- Better CPU isolation, in order to run time-sensitive tasks without interruption

What is (one of the things) preventing this?
- queue_work_on(isolated_cpu)

While working on those, an interesting parallel programming strategy was noticed:
- Use per-cpu structures with local_lock, when a remote CPU needs any action performed, use queue_work_on(target_cpu).
- Works great for rare remote-cpu interactions, but is terrible for CPU isolation
Previous works (Mel Gorman, 01b44456) propose the usage of per-cpu spinlocks instead. But aren't spinlocks expensive?

The objective of this presentation is to show a performance analysis done on per-cpu spinlocks, presenting base info such as:
- Cache coherence & contention: why spinlocks can be expensive
- How per-cpu spinlocks prevent most of this cost?- How does that impact isolated cpus ?

And then showing the numbers:
- How many clock cycles does per-cpu spinlocks actually cost?
- What else can we do to save more cycles, and how those impact performance?
- How much of the impact can be 'hidden' by OOO execution?
- What about contention?
- How does that compare with the current solution?

Primary author: Mr SOARES PASSOS, Leonardo Bras (Red Hat)

Presenter: Mr SOARES PASSOS, Leonardo Bras (Red Hat)

Session Classification: Real-time and Scheduling MC

Track Classification: LPC Microconference: Real-time and Scheduling MC