LINUX LATENCY BREAKDOWN

Oliver Yang (http://oliveryang.net)

08/01/2015

Agenda

- Latency Analysis Requirements
- Current Tools
- Future Work

Agenda

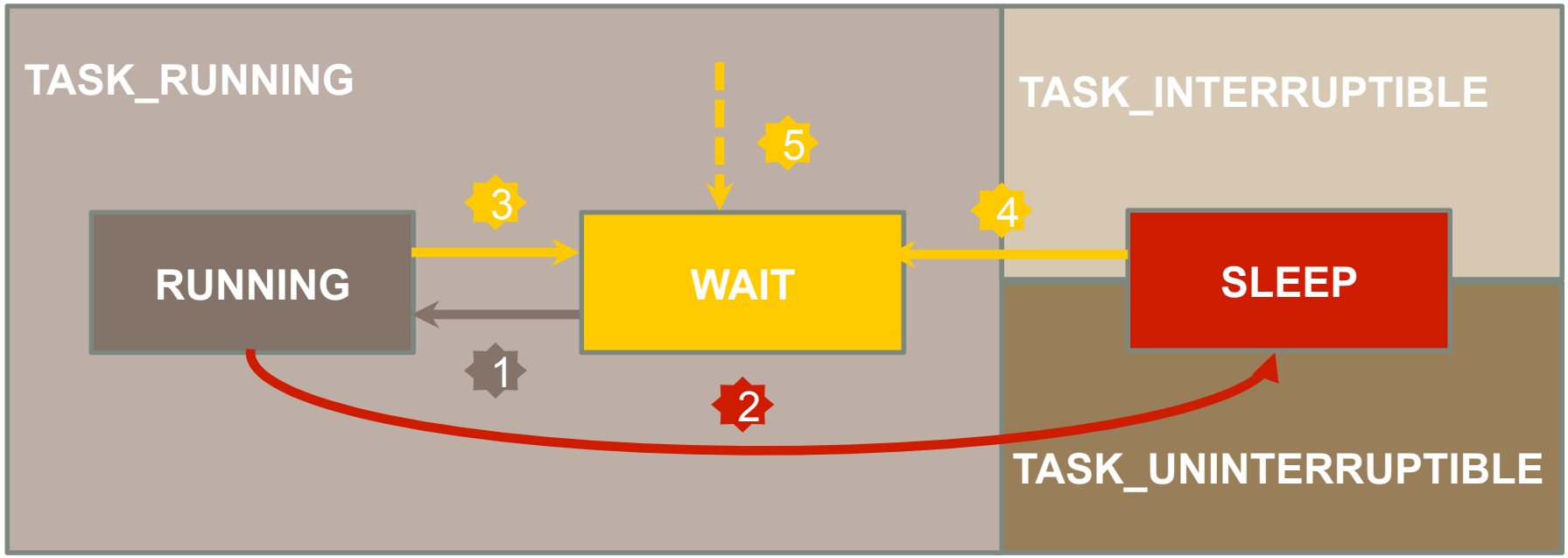
- Latency Analysis Requirements
- Current Tools
- Future Work

Latency analysis challenges

- Burst randomly
- In a very short period
- Reported/found very late
- Always postmortem analysis
- Without enough debug data
- Bug reproduce cost is high
  - eg. Not reproducible by micro-benchmark

Task State Transitions

- Basic concepts
  - 1, 3, 4 - involuntary context switch
  - 2 - voluntary context switch
  - 5 - load balance

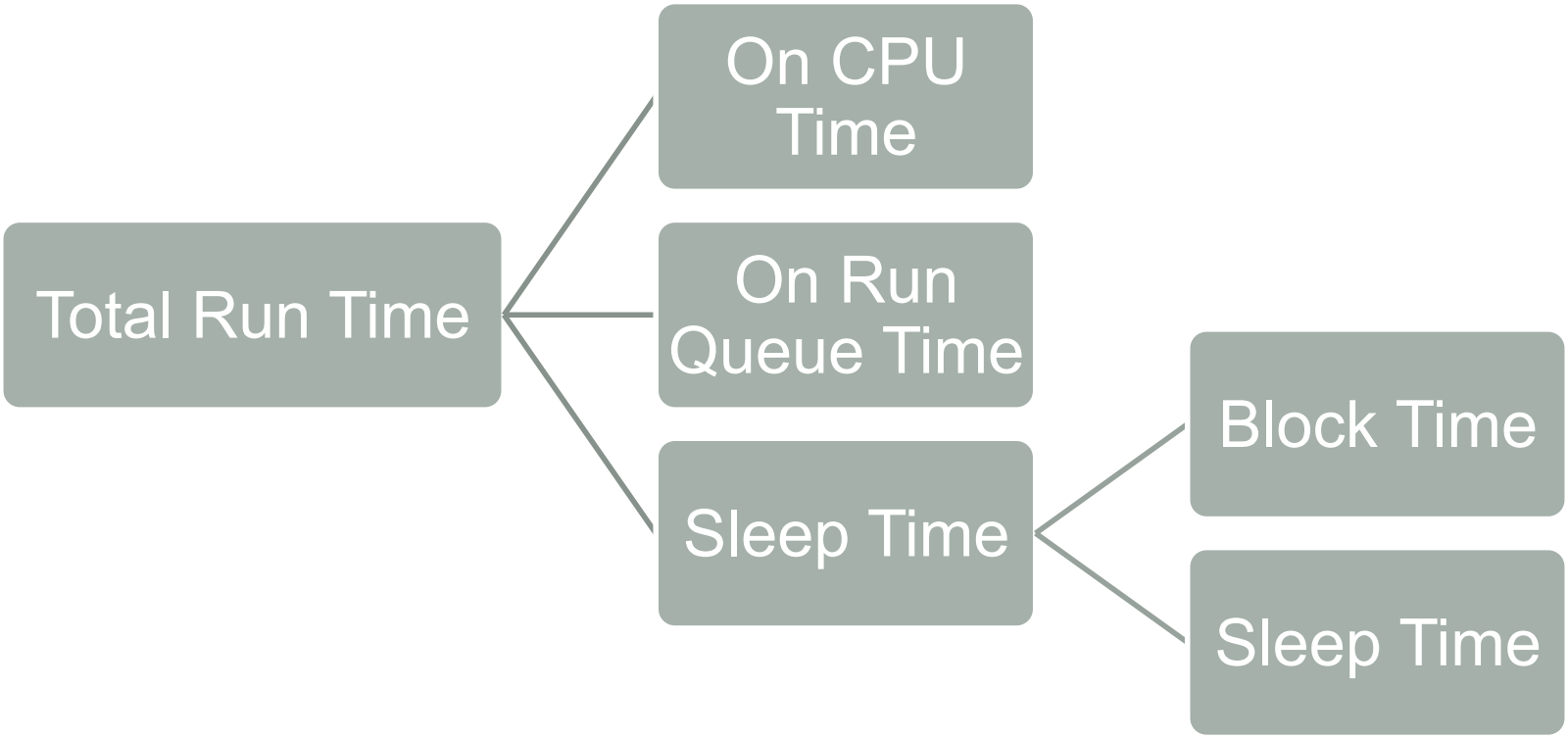


Performance Trade-off

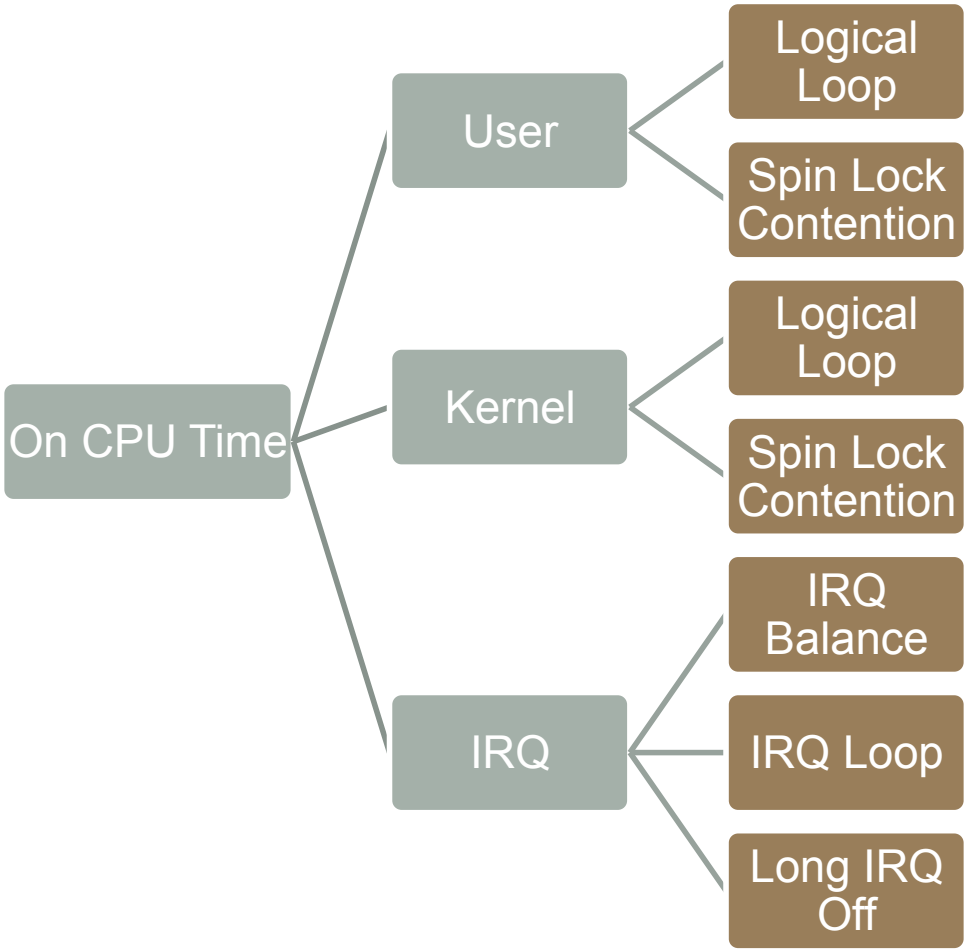
- Latency VS. Throughput/Utilization
  - Throughput could be improved by increase latency
    - eg: how many packets in one interrupts
    - eg: queue depth in IO stack
  - Signs of real latency problems
    - Latency is bad, CPU is overloaded
      - On-CPU analysis
        - High CPU run time in application code?
        - runq waiting?
        - Overhead of context switch?
    - Latency is bad, CPU has low utilization at same time
      - Off-CPU analysis
        - Block time issue?
        - Sleep time issue?

Application Time Distribution

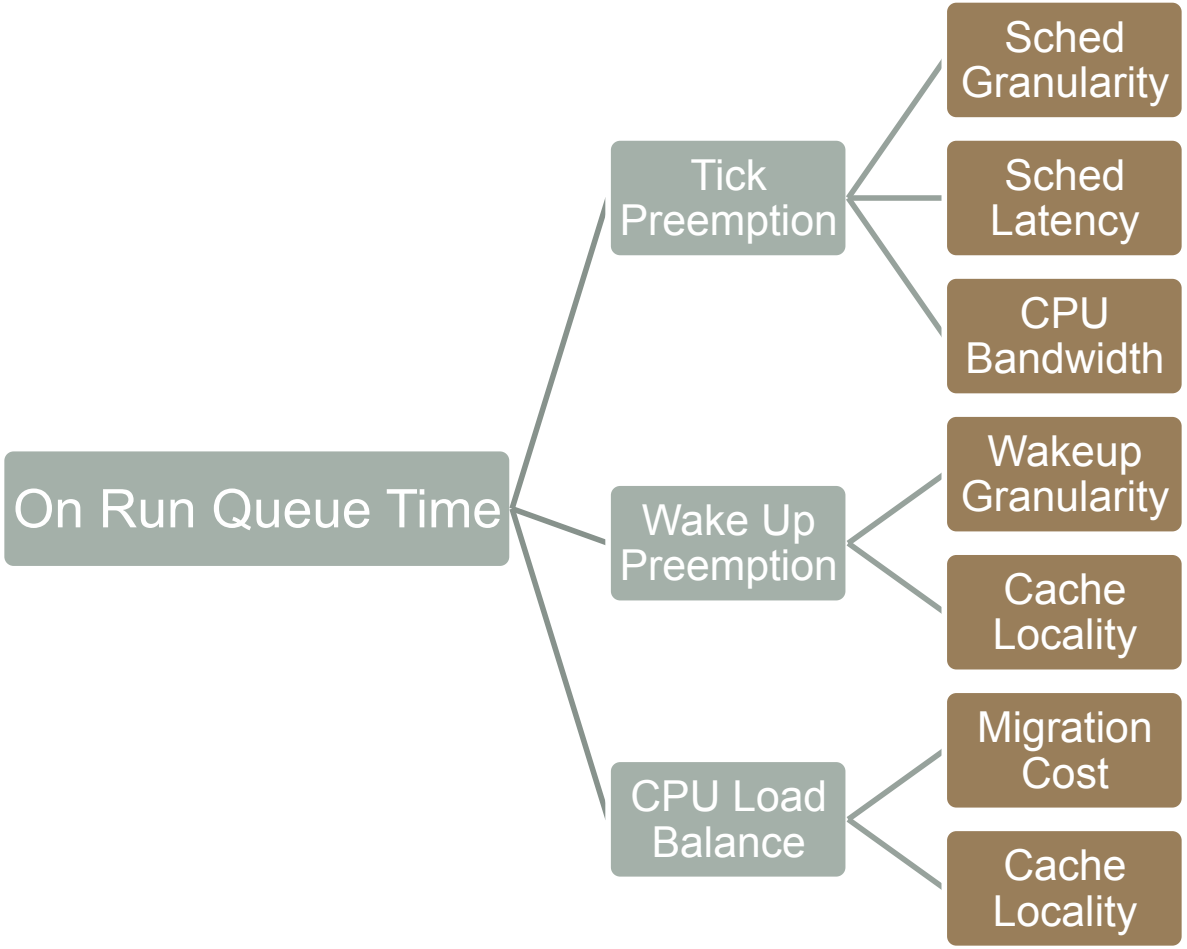
- RUN TIME = ON CPU TIME + ON RUNQ TIME + SLEEP TIME



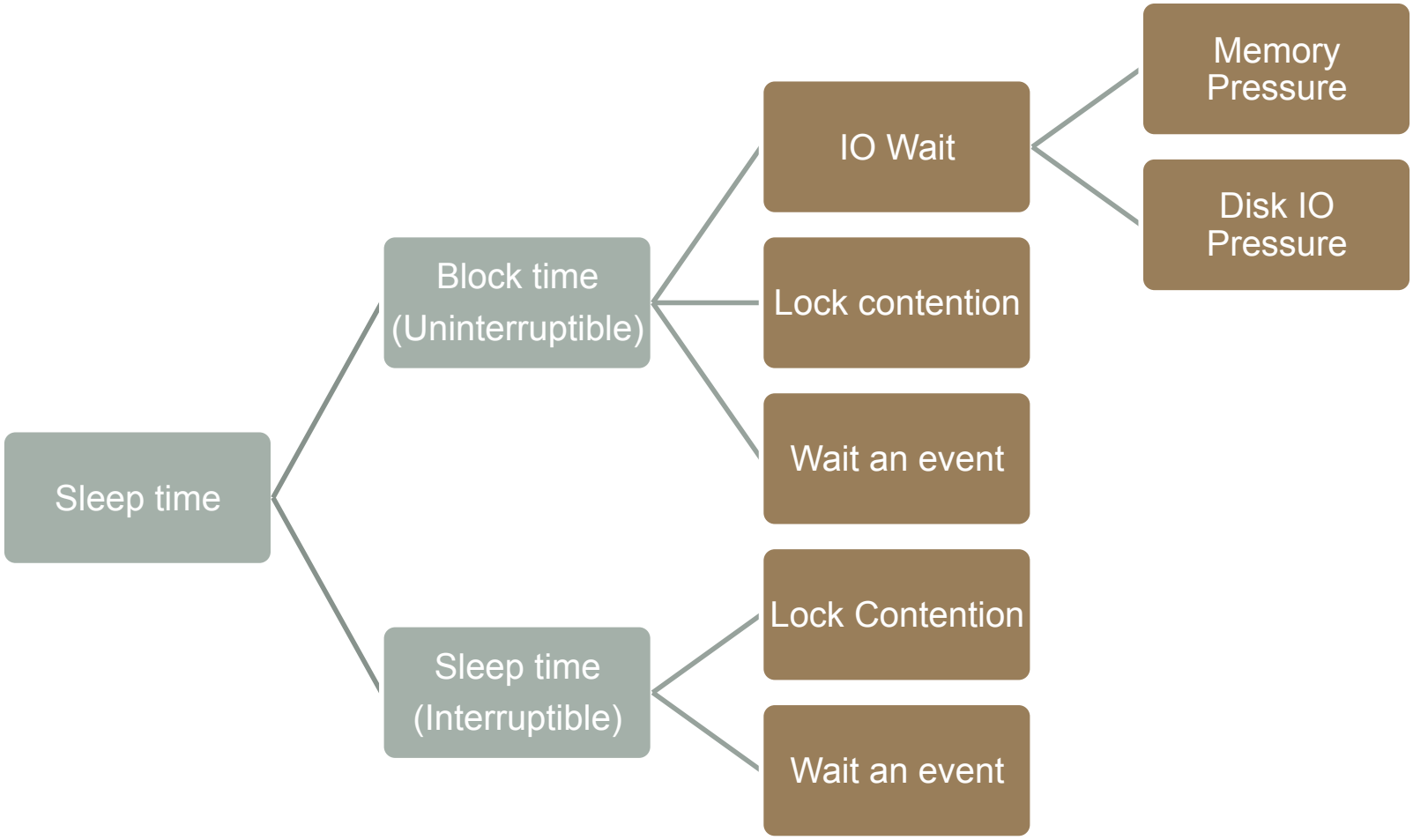
On CPU Time Distributions



On Run Queue Time Distribution



Sleep Time Distribution



Agenda

- Latency Analysis Requirements
- Current Tools
- Future Work

Scheduler Statistic Interfaces

- System wide
  - /proc/schedstat
  - /proc/sched_debug
- Per Process
  - /proc/<pid>/schedstat
  - /proc/<pid>/sched (CFS only)
- Per Thread
  - /proc/<pid>/task/<tid>/schedstat
  - /proc/<pid>/task/<tid>/sched (CFS only)

Scheduler Statistic Use Cases

- UC1 - A FS Job or c API time distribution
  - App perf bug triage: On CPU/On Runq/Sleep time
- UC2 - App global time distribution
  - Identify perf optimization room
- UC3 – Tail latency SLA caused by scheduling latency
  - Watchdog timeout requirement
  - HA timeout requirement
  - OS perf monitoring timeout requirement
- UC4 – Kernel scheduling characterization
  - eg. SCHED_RR side effects or regressions

The getrusage with RUSAGE_THREAD

- Interfaces:
  - Perf thread user and kernel time
  - Per thread voluntary and involuntary context switch
  - Use Cases
    - Perf bug triage by per thread resource usage data
      - Kernel or user problem?
      - SLEEP or RUN queue time problem?

LatencyTOP

- Interface
  - System wide - /proc/latency_stats
  - Per thread - /proc/<pid>/task/<tid>/latency
- Use Cases
  - Per DDFS job or c API sleep time top reasons
    - Perf bug triage
    - Identify DDFS perf optimization room

Kernel lock contention

- Interface
  - System wide - /proc/lock_stat
  - Per thread – perf lock
  - Stopper issue
    - Couldn’t be used on production mode
- Use Cases
  - Per thread or C API kernel hot lock
    - Perf bug triage
    - Identify kernel hot lock

Agenda

- Latency Analysis Requirements
- Current Tools
- Future Work

Kernel Statistics

- C APIs for per thread histogram
- User space system wide monitor tool
- Kernel scheduling perf profiling report
  - Scheduling latency analysis
  - Load balance analysis

Q&A

Questions and Comments?