Scheduler Fairness

Vincent Guittot
LPC 2020
Scheduler MC
Fairness problem description

- How to balance a use case that can't be balanced?
- When the load granularity doesn't match with the topology
  - Odd number of tasks on even number of cpus as an example
- Apply only when system is overloaded
  - For one sched group at least
  - Otherwise tasks have enough runtime
  - Scheduling latency not taken into account
Fairness problem example

- 9 tasks on 8 CPUs
  - Can’t be balanced
  - But should be fair

- Test with 9 always running rt-app tasks on 2x4 CPUs
  - Count effective work per task: number of run events
  - 16 test iterations

- Average unfairness above 20%
  - Ratio between min and max values

- One iteration with more than 40% difference
  - Can’t be higher than 50%
Current rules

- Migrate task with load/2 < imbalance OR
- Any tasks when nr_balance_failed > cache_nice_tries
- The result is an unfair decision
Fairness issues

- Simultaneous busy Load Balance (LB)
  - Smaller domain faster to pull task
  - Some CPUs fail to pull task

  Decrement by 1 jiffies the busy interval

- Huge busy interval on large system

<table>
<thead>
<tr>
<th></th>
<th>2 x 4 cores</th>
<th>2 nodes x 28 cores x 4 threads</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMT</td>
<td>4ms: 1 tick</td>
<td>4ms: 1 tick</td>
</tr>
<tr>
<td></td>
<td>128ms: 32 ticks → 64ms</td>
<td>128ms: 32 ticks → 64ms</td>
</tr>
<tr>
<td>MC</td>
<td>4ms: 1 tick</td>
<td>112ms: 28 ticks</td>
</tr>
<tr>
<td></td>
<td>128ms: 32 ticks → 64ms</td>
<td>3584ms: 896 ticks → 1792ms</td>
</tr>
<tr>
<td>DIE</td>
<td>8ms: 2 ticks</td>
<td>8ms: 2 ticks</td>
</tr>
<tr>
<td></td>
<td>256ms: 64 ticks → 128ms</td>
<td>256ms: 64 ticks → 128ms</td>
</tr>
<tr>
<td>NUMA</td>
<td></td>
<td>224ms: 56 ticks</td>
</tr>
<tr>
<td></td>
<td></td>
<td>7168ms: 1792 ticks → 3584ms</td>
</tr>
</tbody>
</table>

Reduce default interval: use ilog(weight) instead of weight?
Reduce busy_factor?
Fairness issues

- Task selection
  - load/2
  - Any task: LB > sd->cache_nice_tries
    - Monitor average min_vruntime increase per sched_slice
    - Use load_avg/util_avg ratio to calculate imbalance
    - Accumulate imbalance of failed LB in shared data
    - Increase margin during task selection as nr_failed increase

- Active migration
  - LB > sd->cache_nice_tries+2
    - Remove this active migration use case
Test results

● Same test environment as previous test

● Average unfairness around 8%

● One iteration with 12% difference
  ○ Can’t be higher than 50%

● Don’t solve completely the “always same task” problem
  ○ Only reduce the occurrence
Fairness issues

- Often the same task that is pulled
  - Kind of sync between sched_slice and LB interval

- imbalance_pct threshold discards task migration
  - 25% at DIE, NUMA levels
  - Test with 11 always running rt-app tasks on 8 cpus
Thank you