Byungchul Park
LG Electronics

System S/W Engineer
Kernel Developer
Computer Science
Physics
Dance
Outline

- Basic SMP Load Balancing
- Per-Entity Load Tracking
- Attempts for Power Saving
- Energy Aware Scheduling
Outline

- Basic SMP Load Balancing
- Per-Entity Load Tracking
- Attempts for Power Saving
- Energy Aware Scheduling
Load Balancing Cases

- When a CPU becomes idle or periodically
  - Migrate tasks from the most loaded CPU to a less loaded CPU

- When a task is fork(2)ed or exec(2)ed
  - Good chance to change task’s CPU, due to its small cache footprint
  - Select the least loaded CPU

- When a task is woken up
  - Select the previous CPU or current CPU if possible
  - Select the least loaded CPU
Steps for Load Balancing (Migration)

Original Scheduling

1. Measure all CPUs’ load
2. Decide the busiest CPU
3. Decide the idkest CPU
4. Migrate tasks from the busiest CPU to the idlest CPU
Load Balancing

Who is busiest?

CPU 0

CPU 1
Load Balancing
Load Balancing

BALANCED !!!
Steps for Load Balancing (Wakeup)

Original Scheduling

1. Measure all CPUs’ load
2. Decide the idlest CPU
3. Set the woken task’s CPU to the idlest CPU
Load Balancing

Who is idlest?

wait queues

CPU 0

CPU 1
Load Balancing

wait queues

wake up

Take the load!

CPU 0

CPU 1
Load Balancing

wait queues

BALANCED !!
Load Balancing

- We have to measure each CPU load
- What is the CPU load?
CPU Load

- CPU load = \(\sum\) every task load on the CPU

- Task load = a value decided by nice(2)
  - High priority task is considered as a large loaded task
  - Low priority task is considered as a small loaded task

- CPU load average = CPU load \(\times\) CPU utilization (conceptually)
  - Ultimately used for load balancing
  - More precisely, CPU load \(\times\) moving averaged CPU utilization

\[
\text{CPU utilization} = \frac{\text{CPU busy time}}{\text{total CPU time}}
\]
Moving Average

- A method to get the average for sequential data set
- Consider already passed values after reducing its contribution
- Useful to track the **tendency** of values (not instant values)
Moving Average

Thanks to moving average,

- Scheduler can track tendency of CPU load,
- Not only current CPU load at that moment updating it
CPU Load Tracking

Task load
CPU Load Tracking

CPU load
CPU Load Tracking

CPU load

Moving averaged CPU utilization

CPU load average
CPU Load Tracking

CPU load

Moving averaged CPU utilization

CPU load average

Moving averaged CPU utilization

Time
CPU Load Tracking

CPU load

Moving averaged CPU utilization

Moving averaged CPU utilization

Moving averaged CPU utilization

CPU load average

Time
Outline

- Basic SMP Load Balancing
- Per-Entity Load Tracking
- Attempts for Power Saving
- Energy Aware Scheduling
What is entity?

- Scheduling unit
- Basically a task is an entity
- A task group can be an entity in group scheduling
Per-entity Load Tracking

Developed by Paul Turner

- **Before**
  - Apply moving average on the **CPU utilization**
  - Consider CPU’s utilization behavior

- **After**
  - Apply moving average on the each **entity runnable ratio**
  - Consider entity’s running(or sleep) behavior
Original CPU Load Tracking

CPU load → Moving averaged CPU utilization → CPU load average

Time
Original CPU Load Tracking

Entity load

CPU load

CPU load average

Moving averaged CPU utilization

Time
Per-Entity Load Tracking

Entity load
Per-Entity Load Tracking

Entity load

Moving averaged Entity runnable ratio

Entity load average
Per-Entity Load Tracking
Per-Entity Load Tracking
Outline

- Basic SMP Load Balancing
- Per-Entity Load Tracking
- Attempts for Power Saving
- Energy Aware Scheduling
Steps for Load Balancing (Migration)
Power Saving Scheduling

1. **Measure all CPUs’ load**
   - which CPU is chosen is depending on implementation

2. **Choose a CPU from available CPUs**

3. **Decide the idlest CPU**

4. **Migrate tasks from the idlest CPU to the chosen CPU**
Load Balancing for Power Saving

I'm chosen.
I will help another CPU go to idle!
Load Balancing for Power Saving

Take the load!
Load Balancing for Power Saving

SAVE POWER !!!
Steps for Load Balancing (Wakeup)

Power Saving Scheduling

1. Measure all CPUs’ load
2. Choose a CPU from available CPUs
3. Set the task’s CPU to the CPU
Load Balancing for Power Saving

I'm chosen. I will help another CPU go to idle!

CPU 0

I am about to be idle.

CPU 1
Load Balancing for Power Saving

wait queues

wake up

Take the load!

CPU 0

CPU 1
Load Balancing for Power Saving

wait queues

CPU 0

CPU 1
Load Balancing for Power Saving

![Diagram showing CPU usage and wait queues](image)

**SAVE POWER !!!**

- CPU 0: Active
- CPU 1: Idle
- Wait queues: Empty!
Attempts for Power Saving

- Accelerated by “per-entity load tracking” patch
- Important for Big.LITTLE and server farm environment
- Trade off between performance and power saving
Attempts for Power Saving

- To prevent performance regression
  - Perform power saving load balancing, only when the system is not busy
  - Perform original load balancing, when the system is busy,
Small Task Packing
Developed by Vincent Guittot

- **Goal**
  - Pack tasks to fewer CPUs so that more CPUs can be idle
  - achieved by modifying original load balancing

- **How**
  - Migrate tasks to a CPU not sharing hardware power line
  - Kernel cannot put an empty CPU to a deep idle state if it shares hardware power line with busy CPU
Goal

- Pack tasks to fewer CPUs so that more CPUs can be idle
- achieved by modifying original load balancing

How

- Migrate tasks to the busiest CPU among available CPUs
- Not consider hardware power line
Outline

- Basic SMP Load Balancing
- Per-Entity Load Tracking
- Attempts for Power Saving
- Energy Aware Scheduling
Consider CPU Frequency

- Until now only consider CPU idle
- What if consider CPU frequency?
Consider CPU Frequency

![Graph showing the relationship between CPU frequency and power consumption. The x-axis represents CPU frequency, and the y-axis represents power (mW). The graph indicates a significant increase in power consumption as CPU frequency increases from lowest to highest.]
Consider CPU Frequency

Take the load?

CPU 0

CPU 1
Consider CPU Frequency

HIGH FREQUENCY !!!

CPU 0

CPU 1 (Idle)
Consider CPU Frequency

Take the load!
Consider CPU Frequency

SAVE POWER !!!

CPU 0

CPU 1
Consider CPU Frequency

wait queues

CPU 0

CPU 1

wake up

Take the load?
Consider CPU Frequency

wait queues

HIGH FREQUENCY !!!

CPU 0

CPU 1
Consider CPU Frequency

wait queues

wake up

Take the load!
Consider CPU Frequency

wait queues

SAVE POWER !!!
Energy Cost Model

- For saving power consumption,
  - Is it better to pack tasks to fewer CPUs?
  - Is it better to spread tasks to all CPUs?
Consider Frequency States

Architecture A

Architecture B
Consider Idle States

Architecture A

Architecture B
Energy Cost Model

- For saving power consumption,
  - Better to pack tasks to fewer CPUs where power difference between CPU idle states is large
  - Better to spread tasks to all CPUs where power difference between CPU frequency states is large
Energy Cost Model

For saving power consumption, with considering time

• Better to pack tasks to fewer CPUs where energy cost difference between CPU idle states is large
  \[ \text{energy cost} = \text{power integrated by time} \]

• Better to spread tasks to all CPUs where energy cost difference between CPU frequency states is large
OK!

It would be better if it performs the load balancing based on the energy cost.
CPU States (Example)

Idle states

Frequency states

Power (mW)

Idle states

Frequency states

Power (mW)

deepest -> shallowest

lowest -> highest
## Energy Cost Table (Example)

<table>
<thead>
<tr>
<th>CPU States</th>
<th>Energy Cost</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>300</td>
<td>non-idle + frequency state 0</td>
</tr>
<tr>
<td>B</td>
<td>200</td>
<td>non-idle + frequency state 1</td>
</tr>
<tr>
<td>C</td>
<td>100</td>
<td>non-idle + frequency state 2</td>
</tr>
<tr>
<td>D</td>
<td>50</td>
<td>idle state 1</td>
</tr>
<tr>
<td>E</td>
<td>10</td>
<td>idle state 2</td>
</tr>
<tr>
<td>F</td>
<td>1</td>
<td>idle state 3</td>
</tr>
</tbody>
</table>

**System’s total energy cost** = \( \sum \) each CPU energy cost
It looks very simple to perform the load balancing based on the energy cost
Modules Related to Energy Cost

- **Scheduler**
  - Calculate the CPU load *in its own way*
  - Perform load balancing using the load

- **Frequency governor**
  - Calculate the CPU utilization *in its own way*
  - Decide a CPU frequency using the utilization

- **Idle governor**
  - Track the CPU idle pattern *in its own way*
  - Decide a proper CPU idle state using the pattern
Modules Related to Energy Cost

- They are working in its own way *individually*
  - They does not communicate with each other
  - They can conflict with each other easily
  - Changing CPU frequency can affect scheduler load
  - Scheduler does not know the CPU state decided by other modules
Energy Aware Scheduling

[source: https://www.linaro.org/blog/core-dump/energy-aware-scheduling-eas-project/]
Energy Aware Scheduling

- Integrate the scheduler with CPU frequency states
- Integrate the scheduler with CPU idle states
- Keep a table describing energy cost for each CPU state
- Do load balancing based on the energy cost with the table
CPU Idle State Integration

- **Problem**
  - Scheduler cannot distinguish between shallow idle states and deep idle states

- **Solution**
  - To modify the `find_idlest_cpu()` scheduler function
  - To make it distinguishable between different CPU idle states
  - The function is used when it needs to wake up an idle CPU
  - Now, scheduler can wake up the shallowest idle state CPU
CPU Frequency State Integration

Problem
- CPU frequency affects to loads used in scheduler
- A job needs more time to complete in lower CPU frequency
- The load should be regarded as smaller in lower CPU frequency
- Scheduler and frequency governor can conflict with each other easily

Solution
- To register the scheduler itself as a frequency governor
- Scheduler can decide a CPU frequency state directly
- Scheduler can use the CPU frequency to calculate loads
Steps for EAS Load Balancing

1. Identify the state of each CPU
2. Calculate the total energy cost with the states
3. Migrate tasks so that the cost is minimized
Done

We saved power consumption
http://lwn.net/Articles/639543/
http://lwn.net/Articles/520857/
http://lwn.net/Articles/602479/
https://www.linaro.org/blog/core-dump/energy-aware-scheduling-eas-project/
THANK YOU

Questions are always welcome,

max.byungchul.park@gmail.com