Power Efficient Scheduling

Project Summary

Overview

The objective of the proposed work is to schedule processes in a manner that the overall user experience is not impacted and there is a reduction in the power consumption. If processes that require cpu frequency scaled up are scheduled simultaneously on different cores, then the power wastage in scaling cpu frequency up and then down for each of those processes individually, can be reduced.

Intellectual Merit

The proposed work will provide several intellectual benefits. It is a step in the direction of green computation. It is in a way building extra intelligence into the scheduler, further scheduler can be made a more adaptive sub system of kernel. We can work on bringing in such policies that can minimise the energy wastage.

This work can also form the basis of how system softwares can be made more efficient by studying and combining shortfalls and benefits of different subsystems. While it is possible to have individual core frequency scaled, it complex and expensive, it can be beneficial to have frequency scaling across all cores and running processes that require high frequency on them at a time.

Broader Impact of the proposed work

If successful, this work can have significant reduction in the power numbers on the present day computational setups. Many countries are trying to put their best foot forward in saving energy and saving environment. Considering the number of systems available today, any small amount of power reduction can also be a good thing to work on. This can also be a good way of reducing power consumption on battery run devices, like smartphones, tablets.

Project description

Introduction to proposed study

Process scheduler is an important subsystem in the Operating System. It today is implemented as a completely fair scheduler, which gives equal time for all processes. Computer performance depends on the computational power, which is proportional to the frequency at which the system clock runs. System clock is a function of power consumed, so that tells we can have higher performance with increased frequency and increased power consumption. Higher performance is desirable but increased power consumption is not. Frequency governor algorithm can be used to scale frequency in order to have a balance between expected performance and the system power consumption.

It is also known that input output operations take longer than the cpu instructions, and hence for system performance to be good, it is suggestible to scale cpu frequency for I/O intense processes. This work is based on the idea that while the cpu frequency is scaled up for a certain input output operation to take place, and if there are other hardware subsystems that can require input or output to happen from other available interfaces, then scheduling such processes on other processors and using the scaled up frequency for doing I/O operations, can save power. This can save the energy wastage in scaling cpu frequency up and down for processes as and when needed individually. An example to this could be frequency scaling needed for one of the cpus to do dma access and modem to simultaneously fetch data on one of its downlink channels

In order to accomplish this, we need to train the scheduler to know if a thread is I/O intense. This can be done by monitoring a thread everytime it gets scheduled. The number of architectural instructions executed to the sleep time within the scheduler window for a process can give an estimate of whether the process is I/O bound or cpu bound. If an I/O bound thread can be tagged by the scheduler, after monitoring initial few execution cycles, it could look up for those tags if there is an I/O bound process running on one of the processors. Scheduling such a process on another core can let us utilise the bumped up frequency. It is possible to do this if the nice values of processes are not being affected much, that is to say overall system performance is maintained.

An example implementation for the above suggested scheduler model could be utilising the bumped up frequency for some dma for a process scheduled on one cpu, while fetching data from ethernet on modem. The frequency scaling for cpu, I/O subsystems would be required to be done just once if both these threads are scheduled in proximity. Also since every cycle of scaling frequency up and down has a power wastage associated with it, reducing such cycles can save power.

References Citation

https://en.wikipedia.org/wiki/Completely_Fair_Scheduler

https://wiki.archlinux.org/index.php/CPU_frequency_scaling

http://www.informit.com/articles/article.aspx?p=101760&seqNum=2

https://en.wikipedia.org/wiki/Dynamic_frequency_scaling