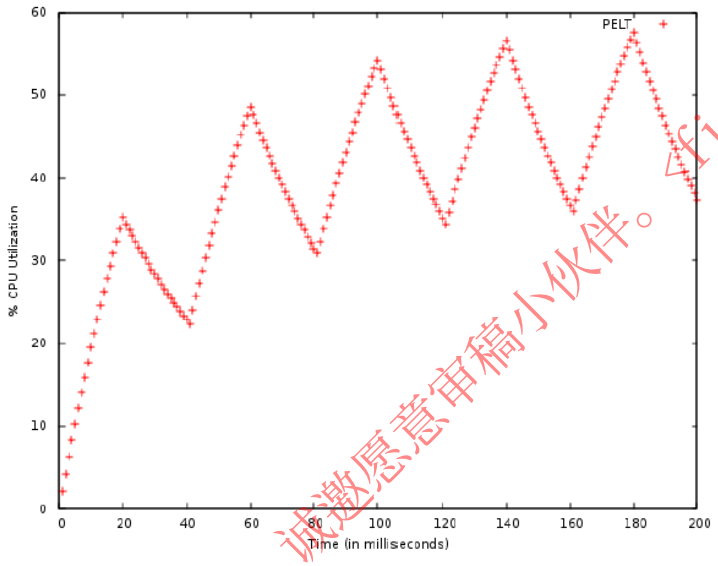
3.6 EAS 绿色节能调度器

在上一章 HMP 调度器的结尾对 HMP 调度器的实现进行一下批评，HMP 调度器的实现是在 2015 年，之后 HMP 调度器就没有再更新了。原来 Linaro 和高通等 ARM 厂商不满足 HMP 调度器，这两年憋出了个大招：WALT&EAS 调度器。

在 2012 年谷歌工程师 Paul Turner 针对 CFS 调度器在计算负载的不合理提出了“Pre-entity load tracking”的进程负载计算方法，简称 PELT，该方法之前已经详细介绍过。但是在手持移动设备特别是手机等发现 PELT 有很多不如意的地方。

举个简单的例子，比如一个进程工作 20ms 然后休息 20ms，绘出其 CPU 使用率的曲线图。



3.x 使用 PELT 的 CPU 使用率

如图中所示 PELT 就像一台老式发动机没法给你像法拉利那种突然提速强烈地推背的快感。在手机使用中经常会产生一些突发的大活（负载重的大任务，heavy task）比如滑屏或者浏览网页，能快速的识别这些大活并且能快速的迁移到计算能力强的 CPU 核上（大核或者最大频率比较高的核）可以有效的提高手机的流畅性。重新识别原本是小活（light task）突然变成大活了，比如渲染线程突然要在屏幕上渲染更多内容。PELT 在辨别进程负载的变化上显得有点迟钝，对于一个突然 100%持续运行的进程它大概需要花 74 毫秒才能最大负载的 80%左右，需要大约 139 毫秒才能达到最大负载的 95%[①]。

PELT 使用一个 32 毫秒的衰减时间，也就是大约 213 毫秒才能把之前的负载忘记掉，也就是历史负载清零了，这个在之前已经讲述过。这个特性对于一些周期性的进程不是很友好，因此有些进程需要存储 CPU 以及频率等信息来提高性能吞吐量，比如一个进程由于网络延迟等原因睡眠 300ms。

对于睡眠或阻塞进程，PELT 还会继续计算其衰减负载也就是继续为就绪队列贡献着平均负载，但是这些继续贡献的负载其实对于下一次唤醒没有什么用处，因此会推延减低 CPU 频率的速度从而增加 CPU 功耗。

针对上述在手持移动设备上种种问题，提出了新的计算进程负载的方法，Window-Assisted Load Tracking（简称 WALT），该计算方法已经被 Android 社区采纳，但是官方的 Linux 内核依然

① https://lwn.net/Articles/704903/