TeSLA: Thermal Service Level Agreement for Mobile Devices

HyunJong Lee*, Minsung Jang**, William Agnew*, Ada Gavrilovska*, Karsten Schwan*
Georgia Institute of Technology*, AT&T Labs - Research**
{josephlee, minsung, wagnew, ada, karsten}@cc.gatech.edu

Rapid advances in the mobile processors show a promise in running compute-intensive applications (apps) on resource-constrained mobile devices. The mobile apps, however, face significant challenges in obtaining the full potential of those processors, especially when performance sustainability [3] and computational sprinting [2] for the apps are absolutely crucial. The need for sustained performance sharply and inevidently increases temperature on the processors, often onto the throttling threshold temperature, which we refer as *thermal state*. In a fanless environment, a software throttling mechanism, called *ThermalEngine*, by vendors comes into the play to cool down the processors, by which apps severly suffer from unpredictable performance.

In this poster, we present our ongoing effort to develop an abstraction of *Thermal Service Level Agreement* (TESLA) between underlying mobile system and apps, to guarantee sustained and expectable performance from mobile apps' perspective and to deliver compuational sprinting with the peak peformance on deamnd. Having an enforcement of TESLA abstraction in system-wide level, the devices not only retrieve full processing performance for the necessary duration of time, but also avoid a degradation of core performance. TESLA and its thermal-state-aware resource management permit mobile devices to operate at the optimal thermal level by maintaining temperature of the cores below the thresholds to avoid thermal engine intrusion. We achieve this by using local priority-scheduler and cloud-backed offloading mechanisms.

Enforcing TESLA abstraction at system-wide level is non-trivial task. First, identifying thermal-related system performance issue at apps granularity does not exist; deternming which apps contribute to temperature changes (increase, most of time) in numerical measure is challenging. To determine 'weight' toward temperature changes by apps, we conducted preliminary study to show that performance-need from even a single app causes unfettered system-wide thermal problem, that results in performance dropoff and affects all apps running in the same device.

Second, providing a mechanism to relieve the thermal-oriented performance issue comes at an overhead. TESLA proposes a mechanism to stay the mobile processors in the Goldilocks temperature for sustained and predictable performance by suppresing temperature under throttling thresholds. TESLA manager periodically polls temperature of heterogenous cores and collects traces of apps to count their contribution toward temperature changes on the cores. Keeping track of temperature changes in 100 ms polling period imposes a small overhead and helps determining weight of the apps. This results the apps to run with long-lasting reasonable performance, that is substantially better than throttled one yet is equivalent to unthrottled one. Lastly, supporting unmodified apps straight from the market and running on real Android devices limit our flexibility on design choice. Off-the-shelf phones with powerful processors reveal more issues on thermal-oriented performance but in a lack of backward compatibility, only a few apps could have ran on the phones.

Our in-progress prototype naively keeps track of the cores' uptime for suspicious apps and the weights are indexed for future use. We expect to cope this weight-based method with priority scheduling mechanism to decide which apps to run at certain temperature, similar to that of MDTM [1]. If TESLA manager detects losing a control over the temperature and local scheduling has no effect, the 'culprit' apps with higher weight are offloaded to nearby/remote cloud clone.

We have tested three representative apps for light, average, and heavy computation tasks, on three different generation of phones. Enforcing TESLA abstraction at system-level showed user-perceived performance improvement by a factor of six.

References

[1] Y. G. Kim, M. Kim, J. M. Kim, and S. W. Chung. M-dtm: migration-based dynamic thermal management for heterogeneous mobile multi-core processors. In *DATE*, pages 1533–1538. EDA Consortium, 2015.

[2] A. Raghavan, Y. Luo, A. Chandawalla, M. Papaefthymiou, K. P. Pipe, T. F. Wenisch, and M. M. Martin. Computational sprinting. In *High Performance Computer Architecture (HPCA), 2012 IEEE 18th International Symposium on*, pages 1–12. IEEE, 2012.

[3] O. Sahin and A. Coskun. On the impacts of greedy thermal management in mobile devices.