Energy Efficiency Optimization of Task-Parallel Codes on Asymmetric and Heterogeneous Architectures

Luis Costero

lcostero@ucm.es

Dpto. de Arquitectura de Computadores y Automática
Universidad Complutense de Madrid

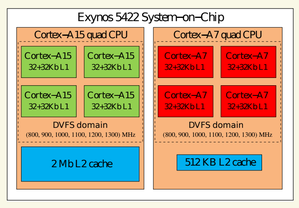
Abstract

We present a family of policies that, integrated within a runtime task scheduler (Nanox), pursue the goal of improving the energy efficiency of task-parallel executions with no intervention from the programmer. The proposed policies tackle the problem by modifying the core operating frequency via DVFS mechanisms, or by enabling/disabling the mapping of tasks to specific cores at selected execution points, depending on the internal status of the scheduler.

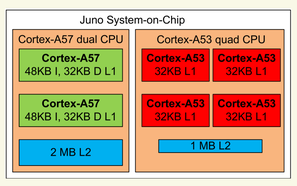
Experimental results on an asymmetric SoC (Exynos 5422) and for a specific operation (Cholesky factorization) reveal gains up to 29% in terms of energy efficiency and considerable reductions in average power.

Introduction

- New trends on HPC:
  - Energy efficiency on parallel applications.
  - Leveraging low-power architectures.
  - Interest on asymmetric architectures.
- ARM big.LITTLE:



Odroid-XU3 platform.

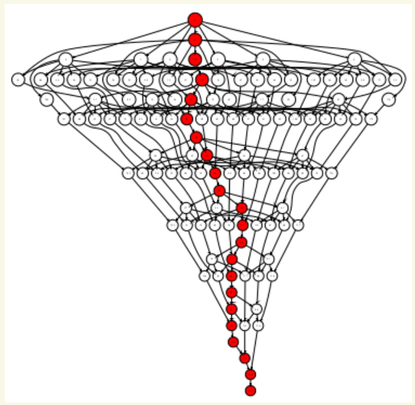


Juno platform.

- **Problem:** We need to adapt current programming models to the new architectural trends, combining:
  (a) High performance
  (b) Energy efficiency
- **Our solution:** Designing a set of policies that modify both task scheduling algorithms and frequency of operation of modern AMPs via DVFS depending on the internal status of the task scheduler.

BSC: Criticality-Aware Task Scheduling for Asymmetric Architectures *(CATS)*

- Dynamic scheduler for OmpSs.
- Focused on performance, not energy efficiency.
- Longest path detection at run time.
- Two kind of tasks:
  - critical tasks: tasks belonging to the longest path.
  - non-critical tasks.



Critical tasks → Mapped over BIG cores.
Non-critical tasks: → Mapped over LITTLE cores (and over BIG cores if they are idle).

Ref.: K. Chronaki et al. *"Criticality-aware dynamic task scheduling for heterogeneous architectures,"* ICS '15, pp. 329-338

- **Dates:** March, 27th — 31st.

ArTeCS — BSC Barcelona Supercomputing Center, Centro Nacional de Supercomputación

- **Funded** by:

SyeC — Supercomputación y eCiencia

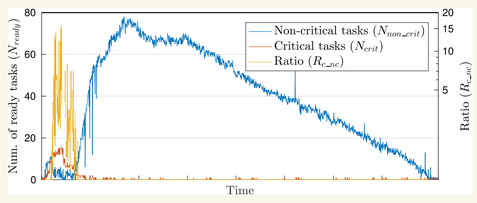
ArTeCS: Energy-Aware Policies

- Goal: To design a set policies integrated with an asymmetry-aware scheduler to increase energy efficiency.
- Policy behavior based on the number of critical and non-critical ready tasks at each moment of the execution.
- Each policy only affects at one of the clusters.

I.- Policies based on frequency scaling (FS)

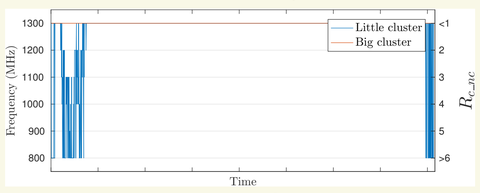
- Based on the internal state of the scheduler. No modifications on the scheduling algorithm are needed.

At run time, the number of ready critical (N_{crit}) and non-critical (N_{non_crit}) tasks is tracked.



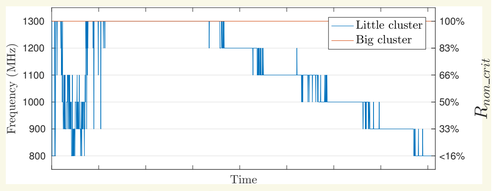
Evolution of the number of critical and non-critical ready tasks.

Policy FS1: Frequency of the LITTLE cluster is modified based on the relation between the number of tasks: $R_{c\_nc} = N_{crit}/N_{non\_crit}$



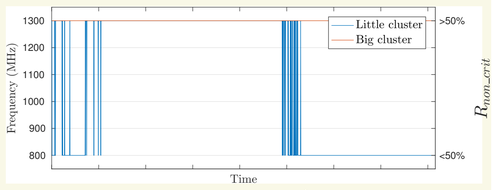
Policy FS1.

Policies FS2/FS3: Frequency of the LITTLE cluster (FS2) or the BIG cluster (FS3) is modified based on the maximum amount of ready tasks recorded previously.



Policy FS2. FS3 will have the same behavior, but applied to the BIG cluster.

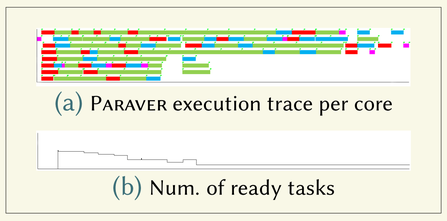
Policy FS2': Similar to policy FS2, but working only with the maximum and minimum frequencies available.



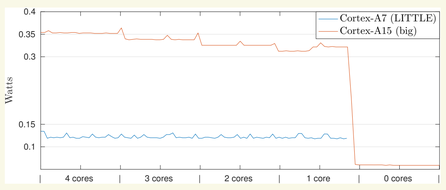
Policy FS2'.

II.- Policies based on task scheduling (TS)

- TS policies decide at runtime the phases in which both clusters are considered to execute tasks, or just one of them is used as a scheduling target.
  - Pros: Using only one of the clusters in specific moments means that power consumption is likely to decrease.
  - Cons: Performance will also be affected (Trade-off solution).



Policy TS2. Cores 0-4 belong to LITTLE cluster. Cores 5-7 belong to BIG cluster.

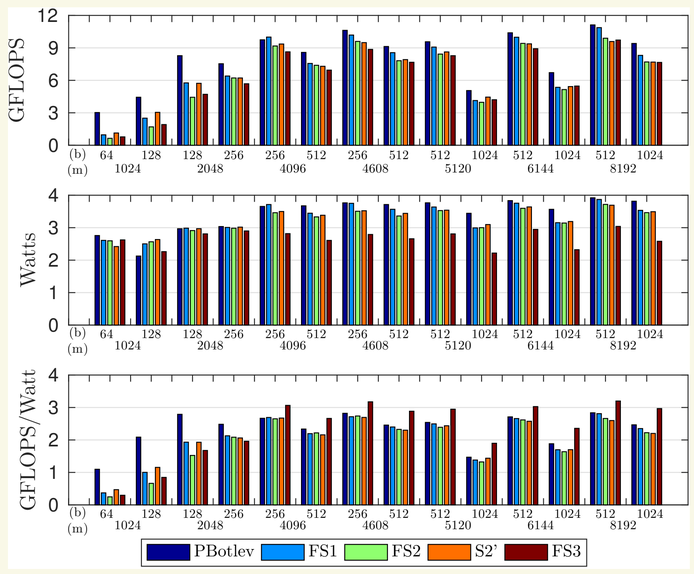


Power consumption of each cluster on idle state with different number of active cores on the ODROID platform.

- *Policies TS1 and TS2:* If the number of ready tasks is low enough, the policy will not assign any new task to the LITTLE cluster (policy TS1) or to the BIG cluster (policy TS2).
- Some platforms allow to switch off a whole cluster under demand via the OS, entailing an important decrease on power consumption.
- Policy TS3 is similar to policy TS2, but deactivating the whole cluster.

Results (Cholesky on ODROID XU3)

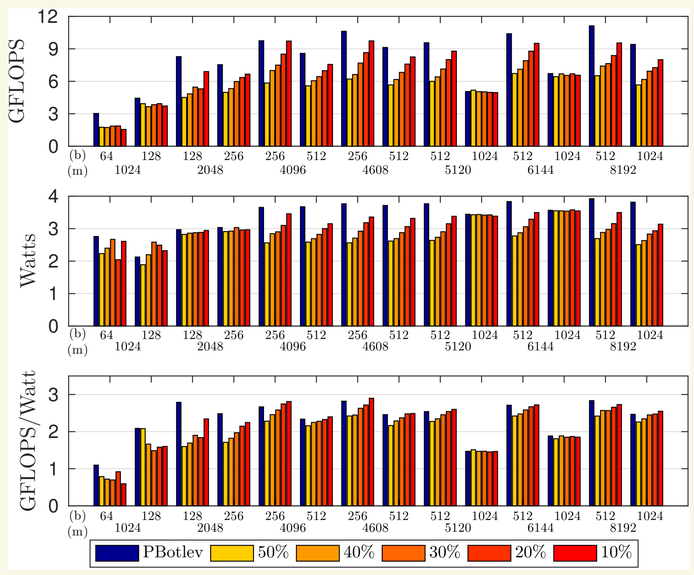
I.- Policies based on frequency scaling (FS)



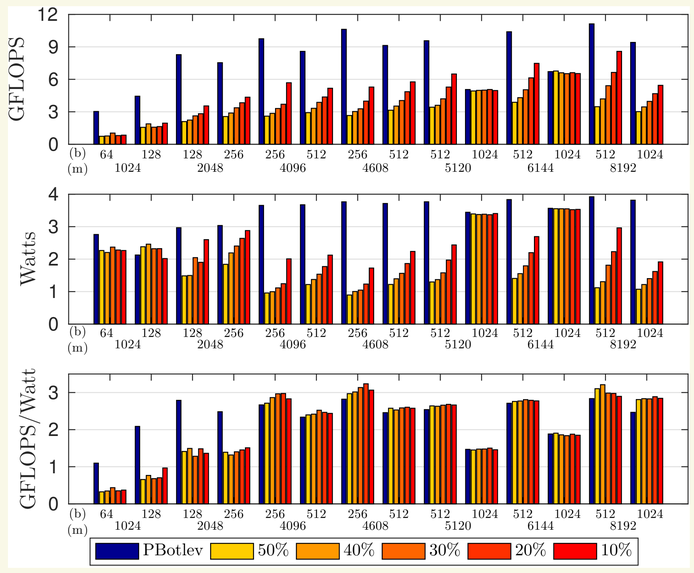
Experimental measures for policies from FS1 to FS3 . PBOTLEV stands for a normal execution using CATS without any policy.

- Two different behaviors: $m \leq 2048$ and $m \geq 4096$.
- Decrease up to 35% in energy consumption (Watts).
- FS3 achieves improvements from 12% up to 29% in Energy Efficiency.

II.- Policies based on task scheduling (TS)



Experimental results for different TS2 configurations. Policy TS1 has a similar behavior.



Experimental results for different TS3 configurations applied to multiple matrix sizes.

- The same two different behaviors: $m \leq 2048$ and $m \geq 4096$.
- Policy TS3 achieves improvements in most of the tested configurations.
- Improvements up to 17% in energy efficiency.

Conclusions and Future Work

- Adaptation of programming models to new architectures necessary: performance/energy efficiency trade-off.
- Applying DVFS techniques to an asymmetric architecture obtains improvements up to 29% on energy efficiency.
- Energy-aware policies are transparent for the programmer.
- Future work and further ArTeCS/BSC collaborations:
  - Extend this idea to other multicore asymmetric architectures.
  - Include other processing elements like GPUs to the architecture.
  - Leveraging BSC benchmaks to further improve and test our policies.