Using hardware counter data to model performance and energy usage in NUMA systems

Doctoral Meeting
February 2019

Miguel López Becoña

Centro Singular de Investigación en Tecnoloxías da Información

Summary

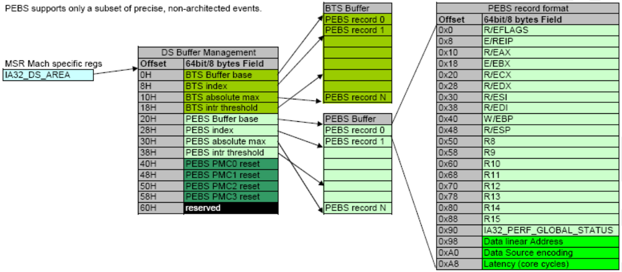
Summary

Motivation and goals

- Memory gap: data locality is a key matter in performance.
  - Specially in NUMA systems.
- Power consumption issues.
- Goals: runtime optimization, performance and energy usage modelling.
- How? Development of a tool that:
  - Reads and gathers hardware counter information about the activity of each thread.
  - Uses of Roofline Model as the basis to characterize the performance.
  - Performs migrations to improve locality.

Hardware Counters

- Microprocessor specific registers.
- Intel PEBS: sampling.
- Low overhead and high accuracy.
- Issues while measuring FLOPS: replaced by instructions. More general.
- `perf_events` Linux interface to access and extend its information.
- Each sample dumps the core state.
  - Generic fields: *timestamp*, CPU ID, PID, TID ...
  - Hardware events: what we actually measure.

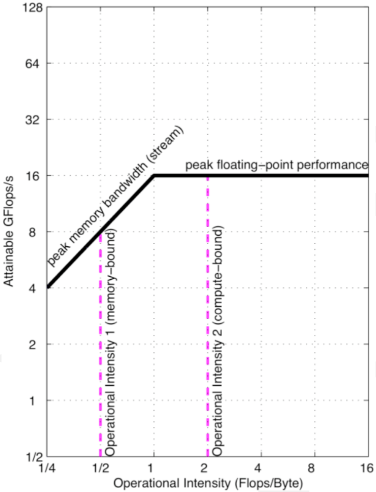


[1]

[1] *https://perf.wiki.kernel.org/index.php/Nehalem*

Berkeley Roofline Model

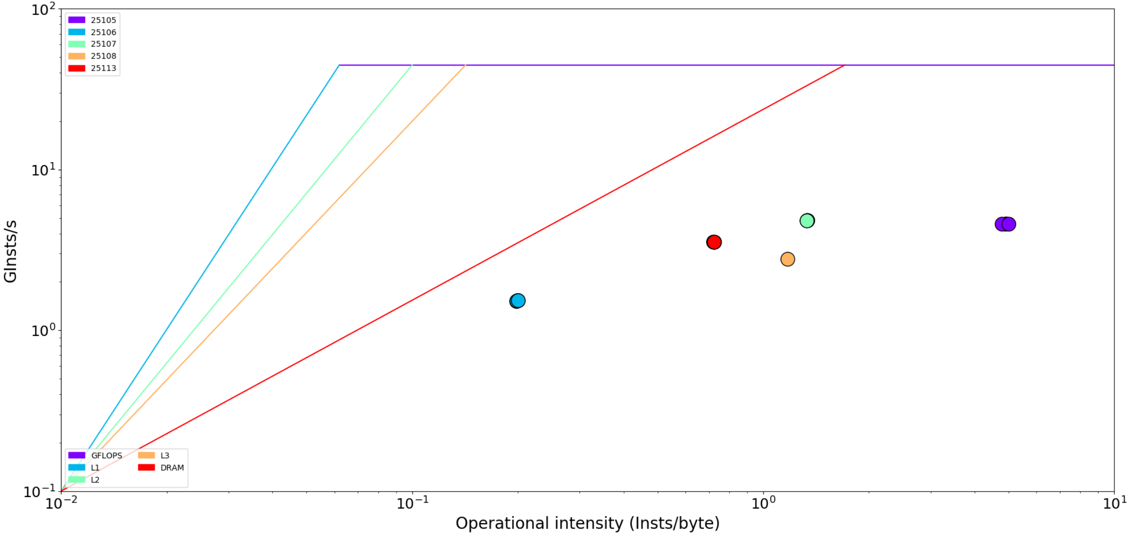
- Performance model.
- 2D plot: simple and easy to understand.
- Helps finding bottlenecks.
  - Clues to optimize our applications.
- X axis: operational intensity.
  - $\frac{FLOPS}{bytes\_read\_from\_memory}$
- Y axis: raw performance (GFLOPS/sec).
- Roof/s: hardware limits.
- Basic classification: side of the graphic.
- Extensions: Dynamic Roofline Model (DyRM), and 3DyRM (useful for NUMA)[2]



[3]

[2] Óscar García Lorenzo et al. *"3DyRM: A dynamic roofline model including memory latency information"*
[3] Samuel Williams et al. *"Roofline: an insightful visual performance model for multicore architectures"*

Example of generated Roofline Model

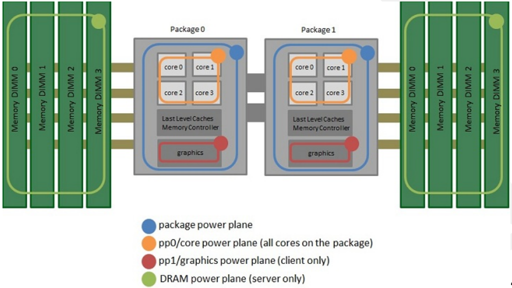


Which hardware events do we currently use?

- Mainly those which are associated to the Roofline Model.
- **Instruction count** (performance related).
- **Offcore requests**: cache lines read from memory (performance related).
- **Memory access latency** (data locality related).
- **Energy usage per node** (load balancing related), with the aid of Intel RAPL.

Intel RAPL (Running Average Power Limit)

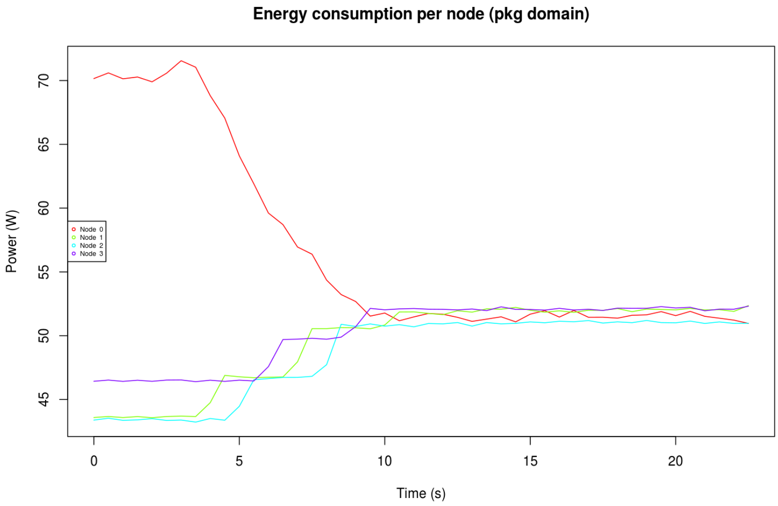
- Software interface to estimate energy usage.
- Divides consumption between logical domains:
  - *cores*
  - *pkg*: *cores* + LLC + memory controller + ...
  - *ram*
  - Variable support.
- Set of buffers for each NUMA node.



[4]

[4] *https://blog.chih.me/read-cpu-power-with-RAPL.html*

Example energy usage plot



Summary

Performance analysis and optimization tool

Main goal

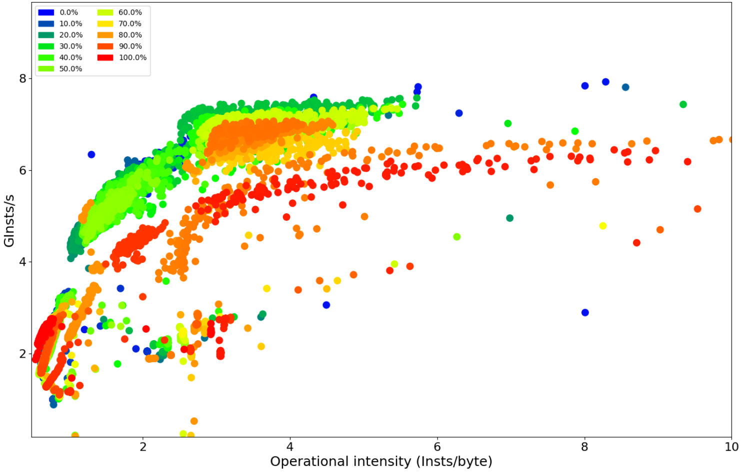
To characterize the performance of parallel programs and performs thread and page migrations accordingly using hardware counters information.

- Two main modes: `JUST_PROFILE` and `DO_MIGRATIONS`.
  - CSV dumping for a posteriori analysis.
- Requires Linux OS and a Intel microprocessor.
- Not very intrusive: low overhead and easy to install and use.
- No `root` permissions required*.
- Automatic detection of system topology (relation of CPUs per NUMA node, etc.).
- Language–independent solution (code is not analysed).

[0] *Paranoid file!

Example of Dynamic Roofline Model plot

- Additional R and Python code for analysing data and making neat plots.



Optimization strategies

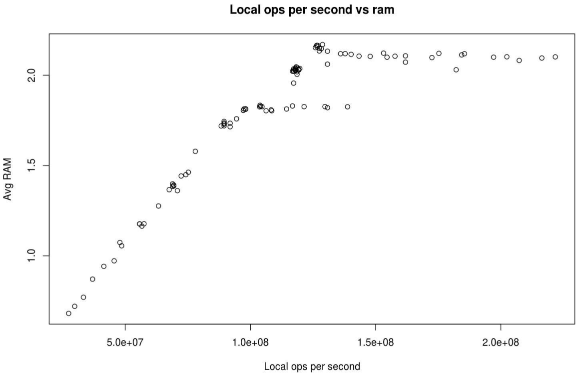
- Decisions about which thread/page to migrate and where.
- Selection of a set of migrations (that might be empty) per iteration.
- Modular implementation of the strategies.
- Usually based on classic optimization strategies and search problems.
  - Simulated annealing, genetic algorithm, random, energy-balancing...
- Most of them are in an early stage of development.

Summary

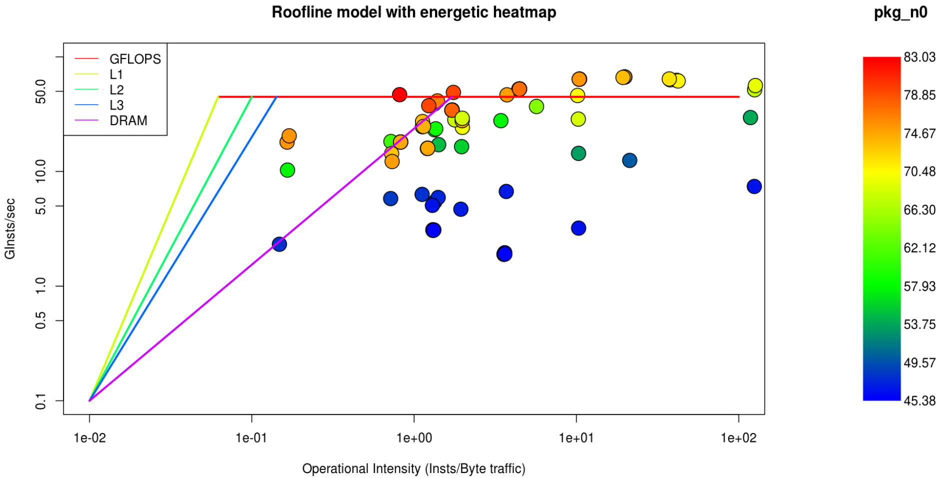
CiTIUS

Energy usage modelling with Roofline Model

- Let's model how energy usage is affected by performance metrics.
- RAPL domains!
- First approach: Roofline Model.
- pkg related to CPU activity (ginsts/s), ram related to memory operations (OI).
- First approach: energetic Roofline.

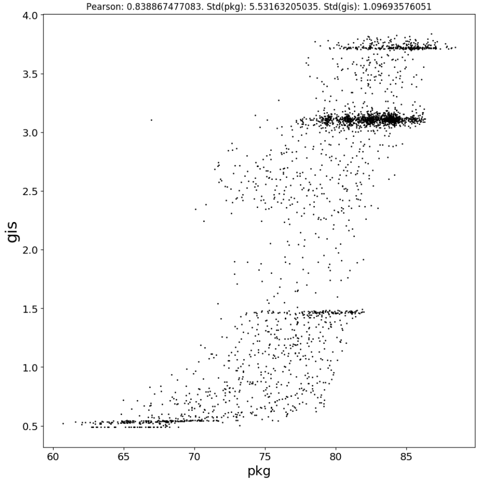


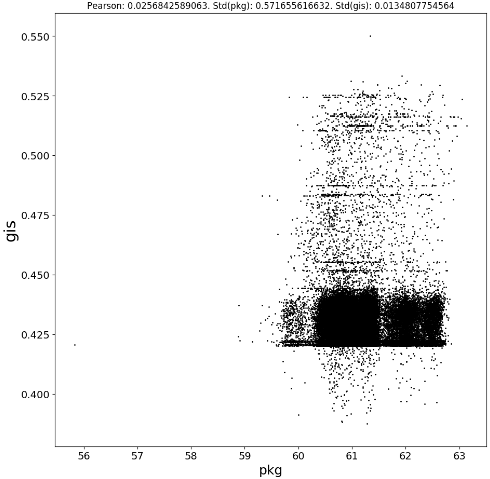
Energetic Roofline



Relation with DyRM

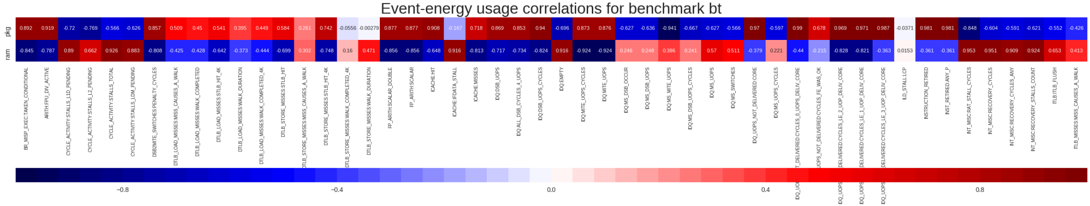
- Good results in some cases, but not in all of them...





Other hardware events

- Maybe we should take into account other/more metrics?
- Search for more correlations...



Current stage: machine learning

- Join data for lots of events: too many columns in CSV.
- Dimensionality Reduction: PCA (Principal Component Analysis).
- Machine learning model with the most N relevant events.

Summary

Conclusions

Development of a new tool

- To characterize the performance of parallel applications in NUMA systems.
- To optimize memory accesses through page and thread migrations.
- To lower power consumption.

Energy usage research work

- To explain it based on hardware counter information.

Future work

- Progressing in the energy study.
- Algorithm refinement for current strategies.
- Implement new strategies.

miguel.becona@usc.es

XUNTA DE GALICIA

UNIÓN EUROPEA

galicia

Cofinanciado pola Unión Europea

Fondo Europeo de Desenvolvemento Rexional “Unha maneira de facer Europa“