Studying the Impact of Power Capping on MapReduce-based, Data-intensive Mini-applications on Intel KNL and KNM Architectures

Tell me what your data looks like and I will tell you how much power you need

Student: Joshua Davis - University of Delaware

Advisors: Michela Taufer - University of Tennessee Knoxville, Sunita Chandrasekaran - University of Delaware, and Tao Gao - University of Delaware

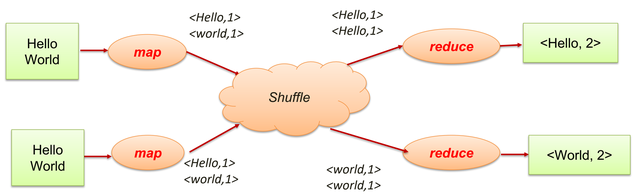
Abstract

In this poster, we quantitatively measure the impacts of data movement on performance in MapReduce-based applications when executed on HPC systems. We leverage the PAPI 'powercap' component to identify ideal conditions for execution of our applications in terms of (1) dataset characteristics (i.e., unique words); (2) HPC system (i.e., KNL and KNM); and (3) implementation of the MapReduce programming model (i.e., with or without combiner optimizations). Results confirm the high energy and runtime costs of data movement, and the benefits of the combiner optimization on these costs.

Key Questions

- Does power capping impact performance of MapReduce-based, data-intensive applications?
- Is there any trade off between power and performance for different datasets and applications?

MapReduce Programming Model



Users specify map and reduce functions. MapReduce handles data movement and communication. Data are treated as key-value (KV) pairs.

PAPI Power Cap Tool

PAPI's 'powercap' component allows users to set power limits and collect energy consumption, using Intel RAPL (Running Average Power Limit).

RAPL uses Dynamic Voltage and Frequency Scaling to limit power consumption of the processor package and memory system.

KNL and KNM Systems

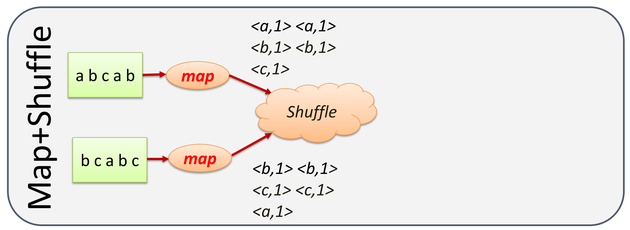
Knights Landing system is an Intel Xeon Phi 7250: one self-hosted node with 68 cores. Its Thermal Design Point (TDP) is 215 watts.

Knights Mill system is a Xeon Phi 7295: one self-hosted node with 72 cores. Its TDP is 320 watts.

Mini-applications in Mimir

We run three wordcount-based mini-apps on top of the Mimir framework, a fully in-memory MapReduce over MPI implementation [3]. They are:

- Map+Shuffle: wordcount up to shuffle stage
- GroupByKey: full wordcount (map+shuffle+reduce)
- ReduceByKey: map+shuffle + combiner optimization replacing reduce



Dataset Generation

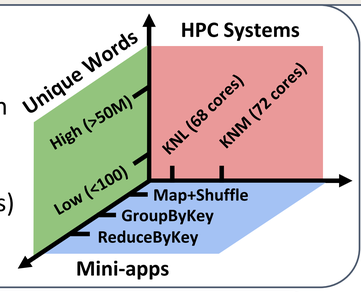
We generate different datasets and vary the **number of unique words** (i.e., XBY, where X is the total number of words and Y is the unique words). A larger number of unique words means less data can be compressed before shuffling in ReduceByKey. E.g.,

- 4 billion words/68 unique: > 99% combiner rate (4B68)
- 4 billion words/42 million unique: 25% combiner rate (4B42M)

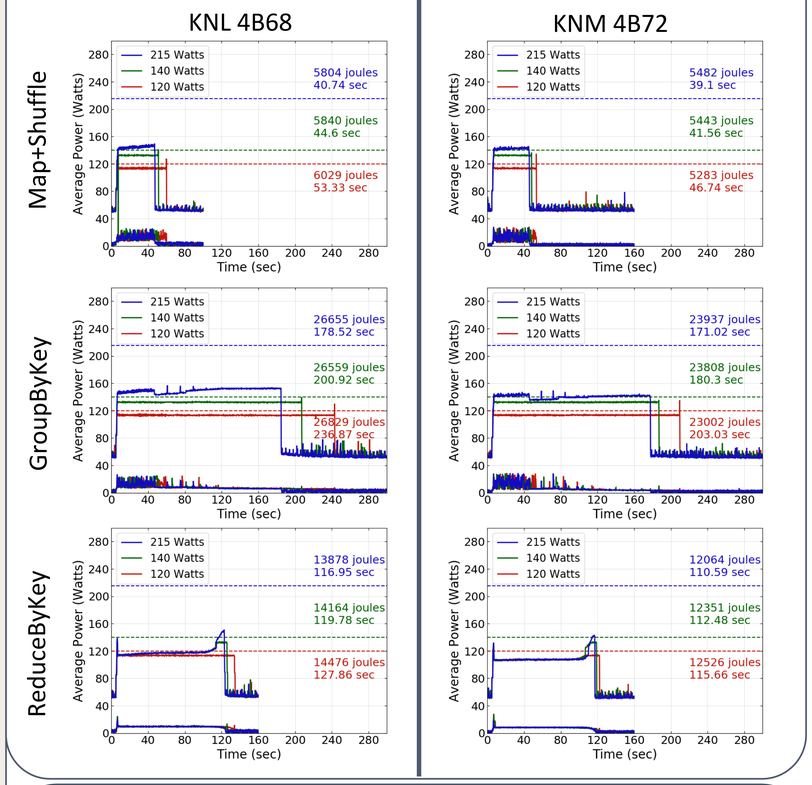
Space of Interest

We study the impact of architecture, unique words, and type of benchmark on performance measured by:

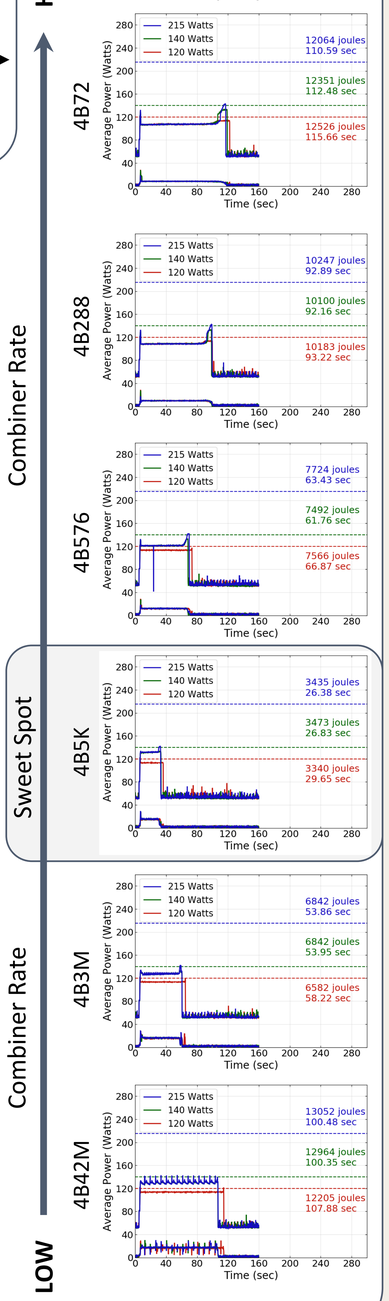
- Runtime (sec)
- Processor energy (joules, upper curves)
- DRAM energy (joules, lower curves)
- Power over time (watts)



Impact of Mini-app vs. HPC System



Impact of Unique Words



Lessons Learned

We quantitatively measure the impact of data movement across cores on both runtime and energy consumption for a set of mini-apps on two manycore systems (KNL and KNM).

We observe (1) the high cost of moving data across processes in terms of runtime and energy when adding the reduce operation to Map+Shuffle; (2) a larger number of cores (KNM) results in better runtime and energy performance; (3) the combiner in ReduceByKey substantially reduces data movement, and ultimately, the runtime and energy consumption (i.e., up to a 46% reduction in runtime and a 50% reduction in energy consumption), without the need for a power cap; and (4) the runtime and energy consumption vary around a 'sweet spot', a region of minimum runtime and energy.

THE UNIVERSITY OF TENNESSEE KNOXVILLE

[1] J. Dean and S. Ghemawat. "MapReduce: simplified data processing on large clusters," in *Commun. ACM* vol .51, no. 1, pp. 107-113, January 2008. doi: 10.1145/1327452.1327492

[2] T. Gao et al., "Mimir: Memory-Efficient and Scalable MapReduce for Large Supercomputing Systems," *IPDPS*, Orlando, FL, 2017, pp. 1098-1108. doi: 10.1109/IPDPS.2017.31

[3] A. Haidar et al., "Investigating power capping toward energy-efficient scientific applications," *Concurrency Computat Pract Exper.* 2018; e4485. doi: 10.1002/cpe.4485

We would like to gratefully acknowledge the assistance of Heike Jagode and Anthony Danalis (Innovative Computing Lab, UT Knoxville), and Pavan Balaji (Argonne National Lab) in preparing this work. Supported by NSF CCF 1841552.