DYSCALE: A MAPREDUCE JOB SCHEDULER FOR HETEROGENEOUS MULTICORE PROCESSORS

ABSTRACT:

The functionality of modern multi-core processors is often driven by a given power budget that requires designers to evaluate different decision trade-offs, e.g., to choose between many slow, power-efficient cores, or fewer faster, power-hungry cores, or a combination of them. Here, we prototype and evaluate a new Hadoop scheduler, called DyScale, that exploits capabilities offered by heterogeneous cores within a single multi-core processor for achieving a variety of performance objectives. A typical MapReduce workload contains jobs with different performance goals: large, batch jobs that are throughput oriented, and smaller interactive jobs that are response time sensitive. Heterogeneous multi-core processors enable creating virtual resource pools based on "slow" and "fast" cores for multi-class priority scheduling. Since the same data can be accessed with either "slow" or "fast" slots, spare resources (slots) can be shared between different resource pools. Using measurements on an actual experimental setting and via simulation, we argue in favor of heterogeneous multi-core processors as they achieve "faster" (up to 40%) processing of small, interactive MapReduce jobs, while offering improved throughput (up to 40%) for large, batch jobs. We evaluate the performance benefits of DyScale versus the FIFO and Capacity job schedulers that are broadly used in the Hadoop community.

EXISTING SYSTEM:

- In the MapReduce model computation is expressed as two functions: map and reduce. MapReduce jobs are executed across multiple machines: the map stage is partitioned into map tasks and the reduce stage is partitioned into reduce tasks. The map and reduce tasks are executed by map slots and reduce slots.
- Daniel et al. propose using architecture signatures to guide thread scheduling decisions.
- Lee et al. propose to divide the resources into two dynamically adjustable pools and use the new metric "progress share" to define the share of a job in a heterogeneous environment so that better performance and fairness can be achieved.
- Polo et al. modify the MapReduce scheduler to enable it to use special hardware like GPUs to accelerate the MapReduce jobs in the heterogeneous MapReduce cluster.
- Jiang et al. developed a MapReduce-like system in heterogeneous CPU and GPU clusters.