Energy Efficient MapReduce

Motivation:

Energy consumption is an important aspect of datacenters efficiency, the total power consumption in the united states has doubled from 2000 to 2005, representing 1.5% of national power consumption in 2005, power consumption of datacenters increases by 15% yearly.

Datacenters with their long history of innovation took the responsibly to lead the nations to use new ideas to reduce energy consumption, they have developed industry power efficiency standards, such as the power utilization efficiency (PUE) metric.

PUE metric is the ratio the total facility power and total IT equipments power, in 2004 the value was 2 and above, which means that most of the datacenters powers is wasted in the physical equipments, such as the cooling and power distribution systems, recently the value of PUE improved approaching 1, which means that power of datacenters is wasted in the IT equipments.

From the PUE metric, it is possible to improve energy efficiency of IT equipments through hardware or software. In this paper they focused in the software, and they selected the MapReduce from all existing paradigms, and they chose the Hadoop implementation of MapReduce.

Contributions:

They started to give a brief introduction about MapReduce, then they presented some analytical framework for energy efficiency, after that they presented some experiments to characterize the energy consumption of Hadoop MapReduce, at the end the provided some quantitative models that can help hadoop operators and developers in identifying the opportunities to improve energy consumption.

Summary:

MapReduce

It was developed at Google as a platform for parallel processing of large datasets, it's attractive qualities made it very popular, MapReduce can be run on clusters of cheap commodity machines, it is highly scalable, makes processing on petabytes of data on thousands and even millions of machines, it's jobs generate mixture of CPU, disk, network activity, it takes care of retrieving data, partitioning input data , outputting data, scheduling parallel execution, coordinating network communication, handling machine failures, it is very simple has two user defined functions Map and Reduce

Analytical Framework

Energy efficiency should be guided by the following considerations; the desired performance metrics, the available system parameters, the appropriate workload, and the method of measuring energy.

1) Metrics:

First of all it is necessary to discriminate between the energy and power, power is the rate of energy consumption, and energy is the result of multiplying power and time. Another crucial metric is the energy per useful work, by using this metric we are able to talk about energy consumption independent of workload size, the challenge for this metric is identifying the unit of useful work, however it is obvious to identify the appropriate unit of work for workloads of same types, such as joules per sorted records for sort workload. Third metric is work rate per watt , measures power efficiency independent of workload size.

In the paper they suggested to use bundle of metrics, and they started with the metrics that are easiest to collect, such as job duration, energy, power, and energy per unit of work like sorted records per joule.

2) System parameters:

For Hadoop/ MapReduce there are three types of parameters that could be optimized.

1) Static parameters: such number of workers to use in the hadoop cluster, number of HDFS data nodes, HDFS block size, and HDFS replication factor

2) Dynamic parameters: such as number of map reduce jobs, HDFS block placement schemes, and scheduling algorithm

3) Hardware parameters: understanding software-hardware interaction is vital to understand software energy efficiency, most important parameters are CPU, speed of various IO paths, including memory, network, and the size of memory

In this paper the focused only on the static parameters.

3) Appropriate Workloads:

Good workload should be representative of actual production workload, should exercise all parts of the system; like major parts of IO paths, common aspects of various scheduling/placement algorithms, and combination of IO/CPU biases. it is not possible to find common workload that meet all these criteria, but MapReduce is a good starting point

4) Energy measurement:

In measuring energy software consumption, it is important to identify system boundary, the convenient boundary is at wall power socket, in this paper they measured energy for all MapReduce master and workers.

Experiments and Results

The goal of the experiments that have been done in this paper is to characterize the Hadoop energy performance from different perspectives, we can group the experiments into the following categories

1) measuring scalability behaviors of sort and Nutch web crawler: The goal of these experiments is investigate the effect of increasing the number of workers on the performance of MapReduce job. From the results of sort experiment, we can say that workload size has been designed to scale to more workers, but

Nutch is not scalable to increasing the number of workers, large number of workers gives no benefit in either job duration or energy or power , in the paper the explained the results because the input data set is not large enough to take advantage of the full parallelism offer by Nutch

2) Artificial workloads to stress different data paths: For experiments in part one, they treated MapReduce job as one part, but the goal of second experiment group is to isolate each part of Hadoop MapReduce job. The three major parts of MapReduce job are: HDFS read, Shuffle, and write. The most surprising result is that increasing the number of workers would not decrease the job duration of HDFS write and consumes more energy, but read and shuffle run faster with increasing number of workers and energy consumed is relatively constant, so we can say that HDFS write doesn't scale at the same rate as read and shuffle and this because of the replication process used by HDFS, so that they made experiment to investigate the relation between the performance and replication factor.

3) Examine the effect of changing static hadoop configuration parameters: these configuration parameters are: HDFS replication, HDFS block size, and Input size, we will present the results of these experiments briefly.

   HDFS Replication: measure the performance of the MapReduce job while increasing the replication level, we conclude from the results that any replication level above 3 is not used, because this imposes storage overhead with negligible additional benefit in failure tolerance.

   HDFS block size: investigate the relation between MapReduce job and size of HDFS block, they suggested according to results they got to increase the block size appropriately with work load size.

   Input Size: the result they got from Nutch experiment made them to suspect that the workload size may impact the energy performance, so that the did experiments to investigate the relation between job performance and the input size by repeating same MapReduce jobs, sort, read, shuffle, and write while changing the input size, the interesting results were for experiment the did to measure records sorted per joule, from this experiment we conclude that, for small input size we can increase the sorted records per joule by decreasing the number of workers, because overhead of coordinating workers and distributing data on workers is a bottleneck, for large input size records sorted per joule remains constant relative to the number of workers, this means that overhead associated with coordinating many workers is balanced by increasing parallelism.

The most inserting result of these experiments especially the HDFS replication level is that power remains identical across different replication levels for all jobs, which means that any energy saving would come from decreasing job duration. Indeed there is a strong correlation in the shape of duration and energy graphs

Quantitative models

1) predicting energy consumption: good way to predict the performance of sort job is to add the performance of the HDFS read, write, and shuffle of the same workload and cluster size, so that they used the following model to predict job duration, energy and power.

$$\mathbf{t_{sort} \approx t_{read} + t_{shuffle} + t_{write}}$$

$$\mathbf{E_{sort} \approx E_{read} + E_{shuffle} + E_{write}}$$

$$\mathbf{P_{sort} \approx average\ [\ P_{read}\ ,\ P_{shuffle}\ ,\ P_{write}\ ]}$$

This model is the same for the IO MapReduce job. The model is not exact in some cases, there are a significant difference between predicted and measured, we have sometimes over prediction, since the model assumes no overlapping between IO tasks, and this is not real, under prediction because of coordination overhead in MapReduce

2) Provisioning machines: It is important to have a model which help Hadoop operators to select how many machines to use in the cluster, and also this kind of model help the Hadoop task scheduler in assigning machines to a particular job. For that they proposed the following expected energy function

$$E(N) = D(N) * P_a(N) + (T - D(N))P_i$$

where:
N: number of workers
D(N): job duration, function of N

$P_a(N)$: power when active, function of N
P_i: power when idle, near constant
T: average job arrival interval
E(N): energy consumed per job, function of N

with this equation we can optimize E(N) for range of N such that D(N) less than or equal T. select the number of workers which gives less energy consumption

3) Replication and reliability: For this model the authors distinguished between two scenarios, the first is the Hadoop deployments on top of reliable storage system, in this case the input data is located in the reliable storage, for computation data is read and stored in HDFS and the is written into reliable storage once the job finishes, so for this scenario failure probability is low, so it is always make sense operate at 1-fold HDFS replication .

   Second scenario, HDFS is the only storage system, in this case durability has priority over performance, when failure probabilities are constant, we need orders of magnitude decreases in cluster size such that 2-fold replication has the same fault resilience as 3-fold replication
   in this paper they presented equations which can help the Hadoop operators to select the replication factor, taking into account the failure probabilities, and the number of machines working in the cluster.

4) Relation between energy efficiency and time efficiency: they came up with simple quantitative model to answer the question about whether reducing job duration is equipment to reducing the energy consumed, they thought about this model, because of the correlation they found between job duration and energy in the HDFS replication experiments. The key result from the analysis they made is that if the work rate is proportional to system utilization, we should run our workloads as fast as possible.

Recommendations and Conclusions

Recommendations for Hadoop Operators:

Measure the duration, energy, and power for MapReduce task, Select the appropriate number of workers, good selection of HDFS block size, selection of HDFS replication depends on whether deployed on reliable storage, or HDFS is the primary

Recommendations for Hadoop/MapReduce Designers

Improve the HDFS replication process, data locality, Investigate a way to find the slow node in the cluster

Recommendations for Hadoop/ MapReduce energy efficiency Researchers

Run experiments on clusters of different machines, use different workloads, Effect of other parameters such as parallel reduce tasks, sort memory limit, and file buffer size

Conclusions:

1) Consider Energy as another performance metric in the design space for computer systems and datacenters

2) Develop Hadoop/MapReduce benchmark that uses bundle of performance metrics

3) They concludes their work with the thought energy efficiency and traditional time efficiency are fundamentally different performance metrics for the same thing

Pros & Cons

Pros:

1) They shed the light on many aspects that are needed for any one who wants to run a MapReduce job.

2) At the end of the paper they gave some recommendations for Hadoop operators according to the experiments they had in this paper

3) The work in this paper differs from other, by focusing on improving on software energy efficiency of MapReduce

Cons:

1) At the beginning of the paper they claimed that they will offer a framework to analyze energy efficiency in general, and MapReduce energy efficiency in particular, but they just talked about energy efficiency of MapReduce

2) It is better to focus on dynamic parameters of the Hadoop/MapReduce framework to improve energy efficiency, rather than concentrating on static parameters

3) Result in some experiments is misleading, such as the experiments on the effect of the HDFS replication factor, for example the behavior of the 4-fold replication is not understandable

4) Results of some experiments is not convenient and not helpful, such as the HDFS block size, they suggested to select the block size appropriately with work load size, but every one who works on the Hadoop knows this, but what it is the exact relation between input size and HDFS block size is not clear

5) We need some experiments to show the relation between different parameters, such as the relation between block size and number of Map workers or reducers

Presenter: Thaer Samar
Opponent: Souza N. Windiartono