UNIVERSITY OF ILLINOIS
AT URBANA-CHAMPAIGN

MapReduce and Relevant Issues

Presented by Shen Li

illinois.edu

Paper List

- [1] ***MapReduce: Simplified Data Processing on Large Clusters***, J. Dean, and S. Ghemawat, OSDI’04.
- [2] ***MapReduce Online,*** T. Condie, N. Conway, P. Alvaro, J. Hellerstein, K. Elmeleegy, and R. Sears, NSDI’10.
- [3] ***Energy Efficiency of Map Reduce***, Y. Chen, and Tracy Xiaoxiao Wang, Tech. Rep. UC Berkeley, 2008.
- [4] ***Statistical Workloads for Energy Efficient MapReduce***, Y. Chen, A. Ganapathi, A. Fox, R. Katz, and D. Patterson, Tech. Rep. UC Berkeley, 2010.
- [5] ***On the Energy (In)efficiency of Hadoop Clusters***, J. Leverich, and C. Kozyrakis, HotPower'09.
- [6] ***GreenHDFS: Towards An Energy-Conserving, Storage-Efficient***, Hybrid Hadooop Compute Cluster, R. Kaushik, and M. Bhandarkar, HotPower'10.
- [7] ***Energy Management for MapReduce Clusters***, W. Lang, and J. Patel, VLDB'10.

illinois.edu

Outline

- MapReduce Overview
- MapReduce Acceleration
- Energy issues of MapReduce

Outline

- MapReduce Overview
- MapReduce Acceleration
- Energy issues of MapReduce

illinois.edu

Map and Reduce in Functional Programming

- Map: applies a given function element-wise to a list of elements and returns a list of results.
- A Simple Example:
  - L = (1, 2, 3, 4, 5);
  - f : Multiply an element by two;
  - Map(f, L) returns (2, 4, 6, 8, 10).

Map and Reduce in Functional Programming

- Reduce: deals with a combining function and a list of elements of some data structure. The Reduce then proceeds to combine elements of the data structure using the function in some systematic way.
- A Simple Example:
  - L = (1, 2, 3, 4, 5);
  - f : add two elements;
  - Reduce(f, L) apply f to L recursively, returns 15.

illinois.edu

Motivation: Large Scale Data Processing

- Want to process lots of data
- Want to parallelize across hundreds/thousands of CPUs
- Want to make it easy

illinois.edu

Basic Idea

- Split input file into pieces. (Generate a list)
- Instead of applying a function to the original data, apply it to each piece (or *split*). (Map a function to a list)
- Collect all the output from each piece to calculate the final result. (Reduce a list based on a function)
- Users can customize map and reduce functions.

Implementation

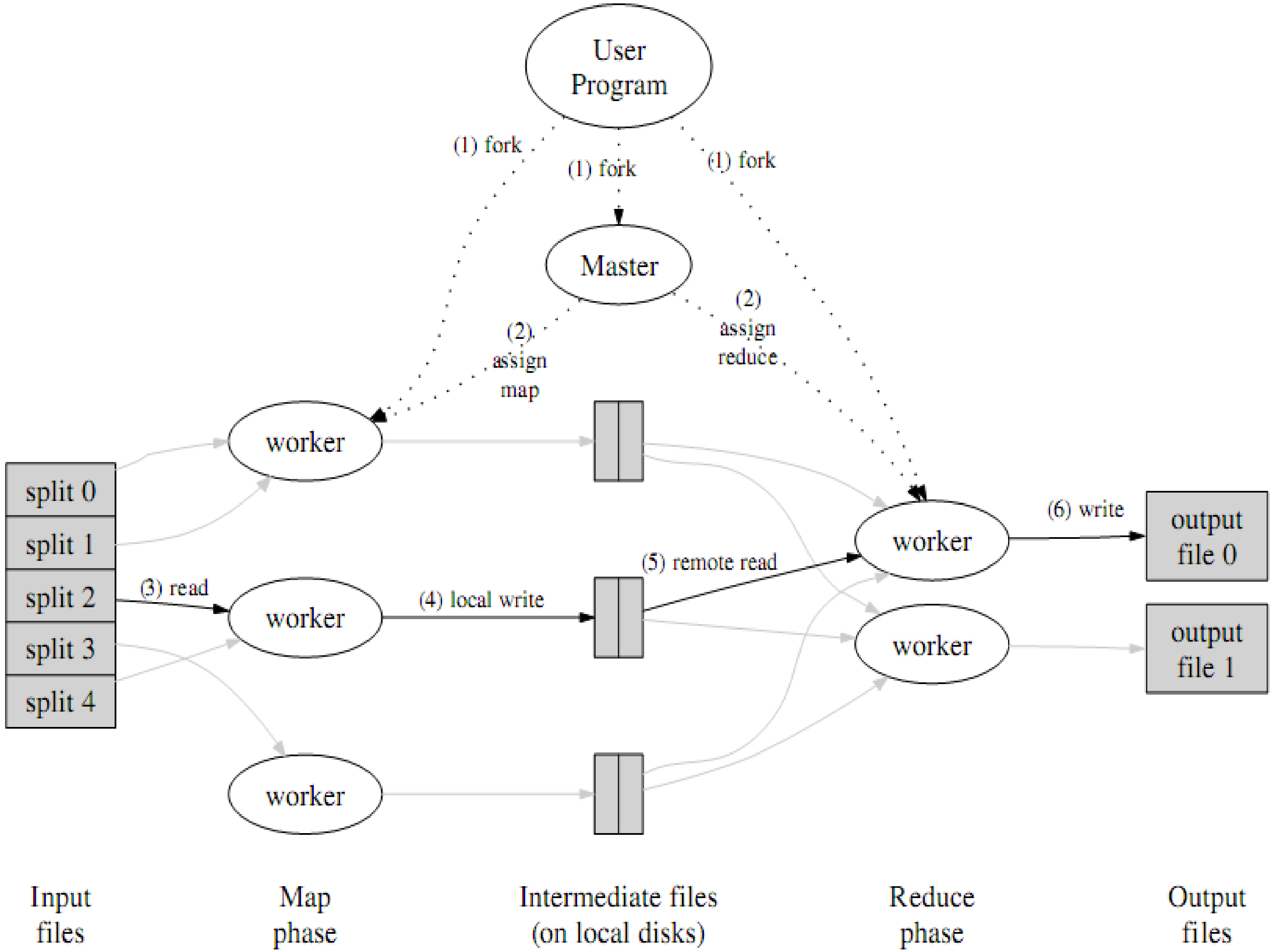
- Master node:
  - Slipt input data into pieces;
  - Assign tasks to map workers;
  - Inform reduce workers where the outputs of map phase are.
- Worker node:
  - Conduct mapping or Reducing;
  - Mappers run map function on splits;
  - Reducers run reduce function on partitions.

illinois.edu

Implementation

- Job
  - Defined on input file;
  - To finish one job, master node will invoke multiple map Tasks and reduce Tasks.
- Task
  - Defined on a split;
  - Assigned by master node to worker node.

illinois.edu



illinois.edu

Cited from [1]

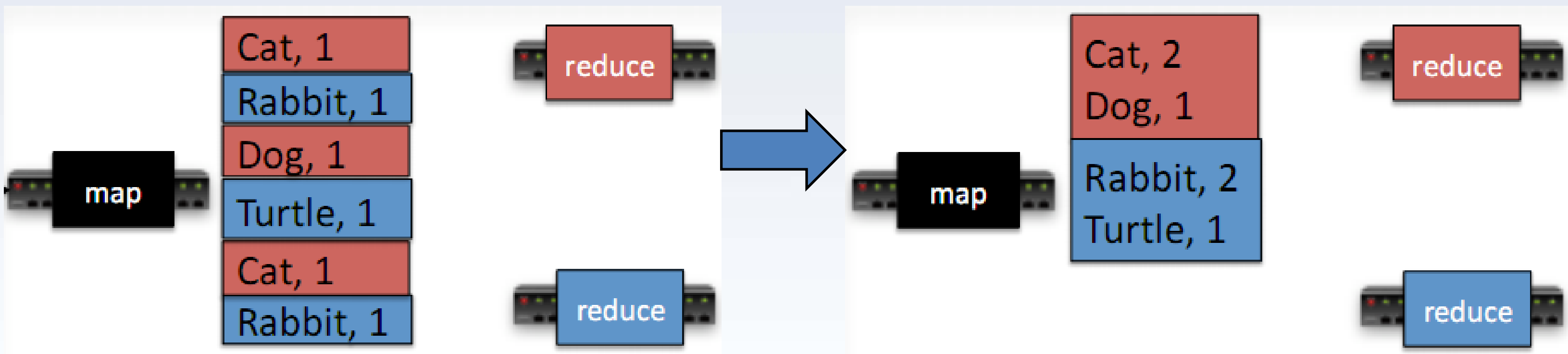
Outline

- MapReduce Overview
- MapReduce Acceleration
- Energy issues of MapReduce

illinois.edu

Acceleration

- Problem: Too much data to pass from Map worker to Reduce worker
  - Solution: Map nodes apply combiner functions to their local output.



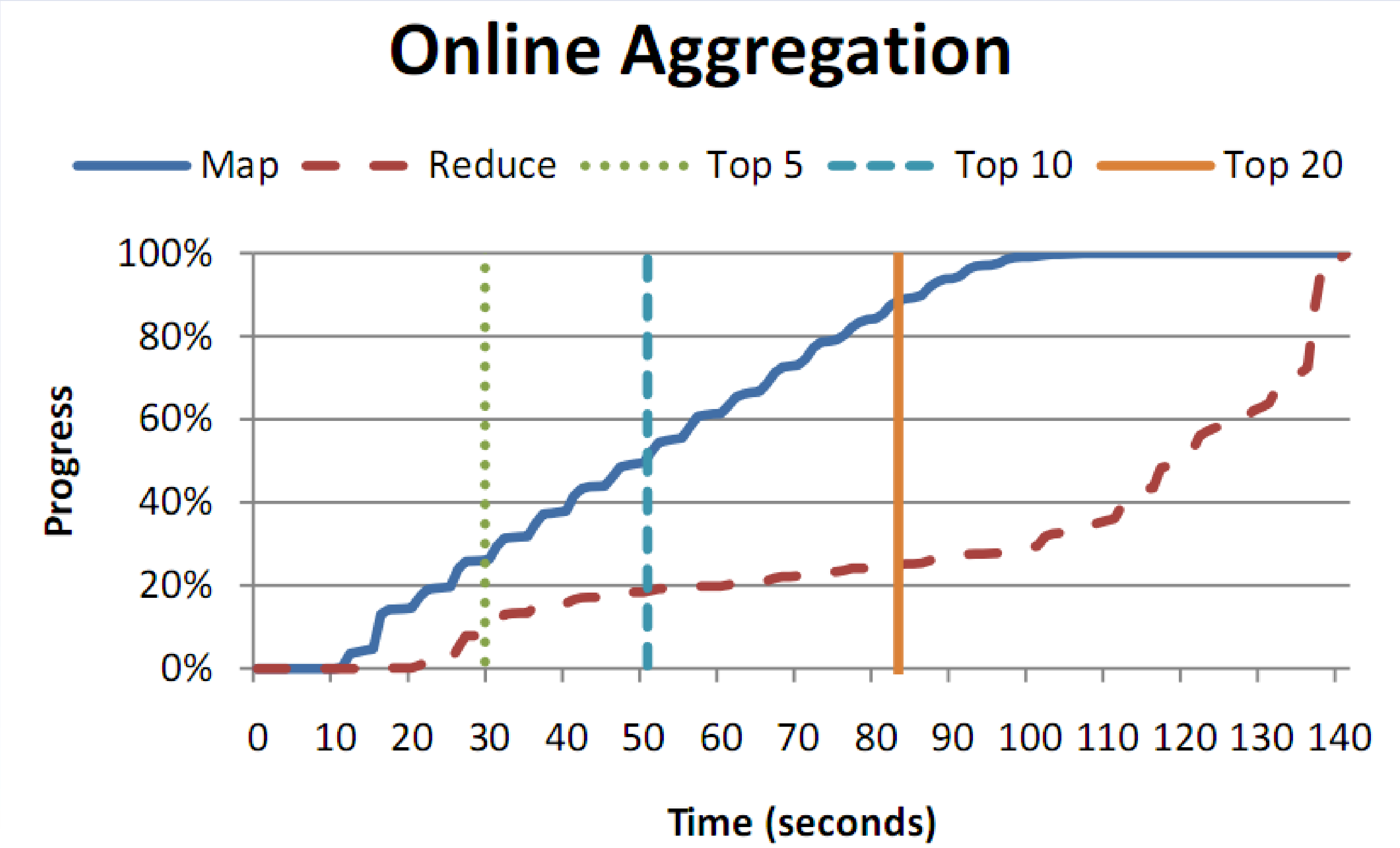
illinois.edu

Cited from [2]

Acceleration

- Problem: No reduce can start until map is complete.
  - Solution: Master redundantly executes “slow-moving” map tasks; uses result of first copy to finish.
  - Solution: Intermediate data is pipelined between mappers and reducers. Thus, reducers begin processing data as soon as it is produced by mappers.

Evaluation



Cited from [2]

Outline

- MapReduce Overview
- MapReduce Acceleration
- Energy issues of MapReduce

illinois.edu

Knobs

- Configuration [3][4]
- Machine On/Off [5][6][7]
- DVFS
- Temperature
- Workload [4][6]

illinois.edu

Cluster Configuration

- General cost metric [3]
  - All cost over all benefits
  - Minimize the metric

$$\frac{f_1(energy)f_2(latency)f_3(number\ of\ machines)}{f_4(workload\ size)f_5(degree\ of\ replication)}$$

$$=\frac{f_1(E)f_2(t)f_3(M)}{f_4(W)f_5(R)}$$

illinois.edu

Cluster Configuration

- Their model [3]
  -

$$\frac{E}{WR} = \frac{Pt}{WR}$$

E: Energy
t: Delay
M: Number of machines
W: Workload
R: Replication

illinois.edu

Arguments

- $\frac{E}{WR} = \frac{Pt}{WR}$ vs $\frac{Et}{WR} = \frac{Pt^2}{WR}$
  - Too heavily prioritize the latency.
  - A system could achieve better performance just by decreasing the workload.

E: Energy
t: Delay
M: Number of machines
W: Workload
R: Replication

illinois.edu

Arguments

- $$\frac{E}{WR} = \frac{Pt}{WR}$$ vs $$\frac{EM}{WR} = \frac{E_{per\ machine} M^2}{WR}$$
  - Too heavily prioritize the number of machines.
  - A system could achieve better performance just by using fewer machines.

E: Energy
t: Delay
M: Number of machines
W: Workload
R: Replication

illinois.edu

Cluster Configuration[3]

Map task	Reduce task	Component stressed	Workloads
Compute π w/ random samples	--	Map compute	20, 80, 160 hundreds of millions of samples
Read from input	--	Map read	40, 160, 320 millions of pairs
Write interm. pairs	--	Map write	40, 160, 320 millions of pairs
Write interm. pairs	Sort	Reduce sort	40, 160, 320 millions of pairs
--	Write to output	Reduce write	40, 160, 320 millions of pairs
--	Compute π w/ random samples	Reduce compute	20, 80, 160 hundreds of millions of samples

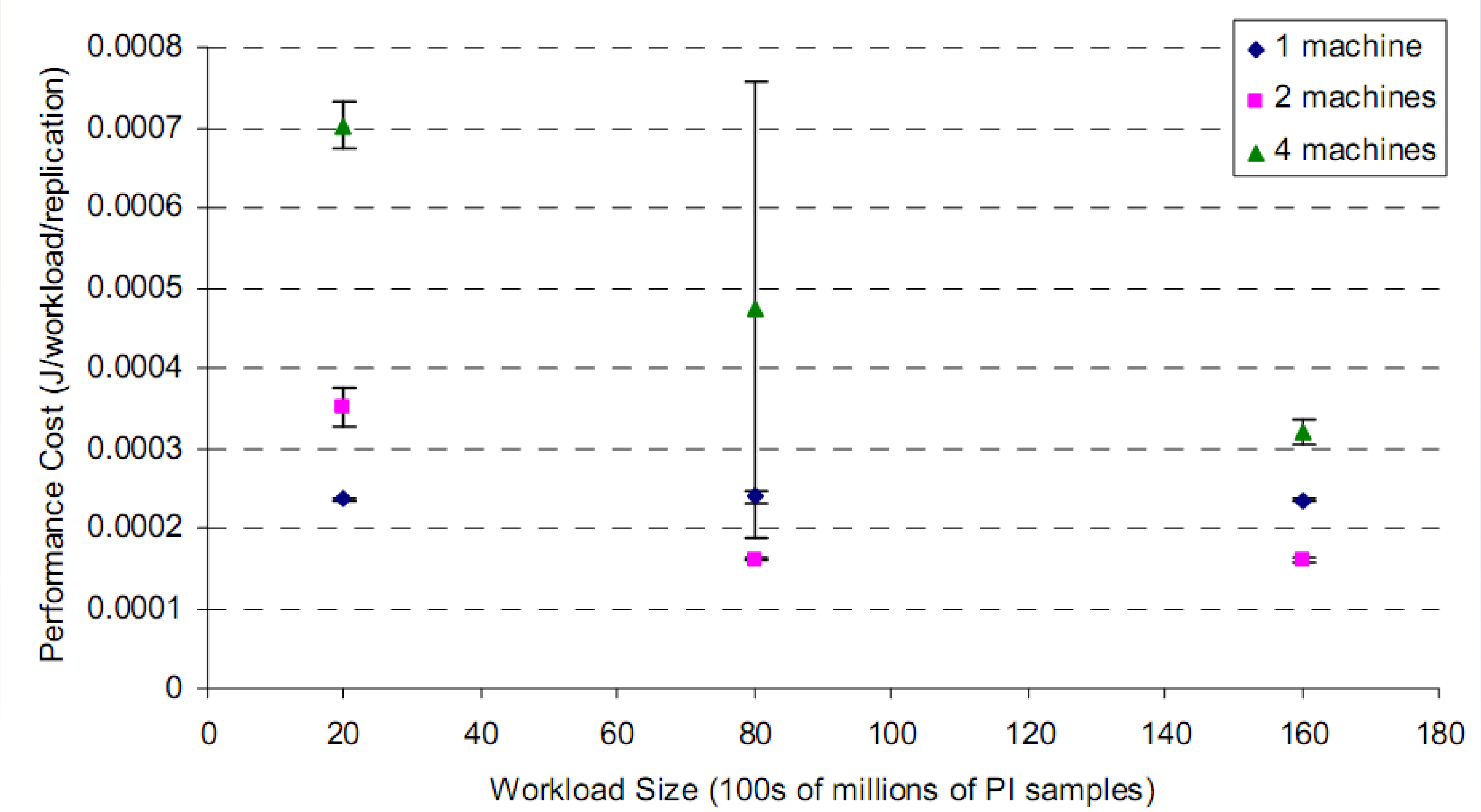
illinois.edu

Cluster Configuration[3]

Number of machines	Replication	Number of map jobs	Number of reduce jobs
1	1	1	1
2	2	7	3
4	3	13	3

illinois.edu

Result



Map Compute

Cluster Configuration[4]

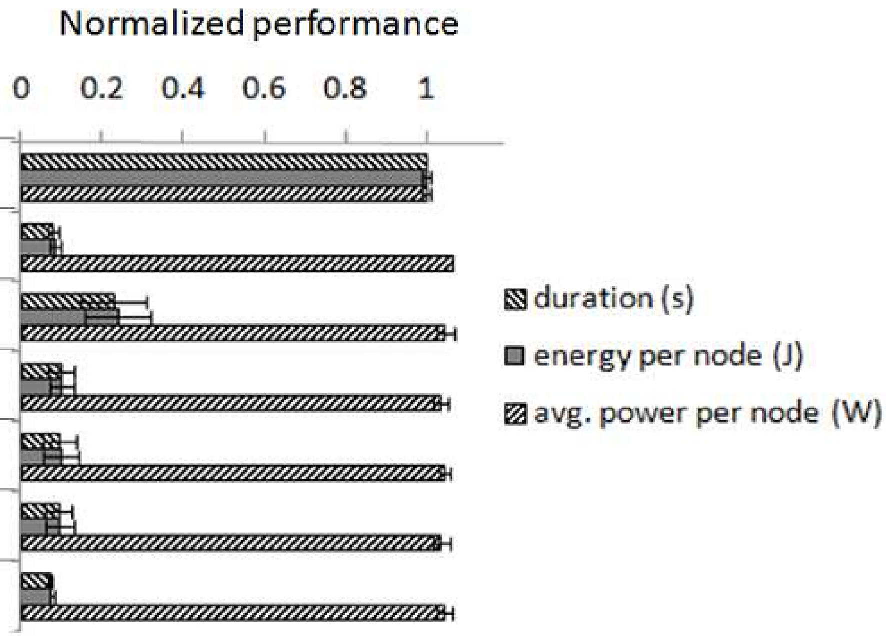
- 1. Launch jobs as they arrive vs. queue up the job and launch them in batches;
- 2. For batched execution, launch all jobs on the queue at the same time vs. in a staggered fashion;
- 3. Use the standard HDFS block size of 64MB vs. larger block sizes;
- 4. Assign the default 4 task trackers per node vs. more task trackers per node.

illinois.edu

Result

Configuration number	Batch?	Staggered launch?	HDFS block size	Task trackers per node
1	No	N/A	64MB	4
2	Yes	No	64MB	4
3	Yes	No	64MB	8
4	Yes	No	80MB	4
5	Yes	No	80MB	6
6	Yes	No	80MB	8
7	Yes	Yes	80MB	8

Config. #	Duration (s)	Energy (MJ)	Avg. power (W)
1	86524±13	15.6±0.14	180±1.5
2	6830±971	1.31±0.19	192±0.2
3	19756±6930	3.71±1.2	188±3.8
4	8496±2662	1.58±0.47	186±3.3
5	8180±3664	1.54±0.69	188±2.0
6	7981±2735	1.49±0.55	186±4.1
7	6266±399	1.18±0.09	188±3.6



Knobs

- Cluster Configuration [3][4]
- Machine On/Off [5][6][7]
- DVFS
- Temperature
- Workload [4][6]

Machine On/Off

- Main Challenge
  - DFS: Files are stored in Distributed File Systems, each machine holds a subset of whole data. Turning off a set of machine can make some data unavailable.
  - SLA: The cluster should satisfy Service Level Agreement even after turning off some machines.

illinois.edu

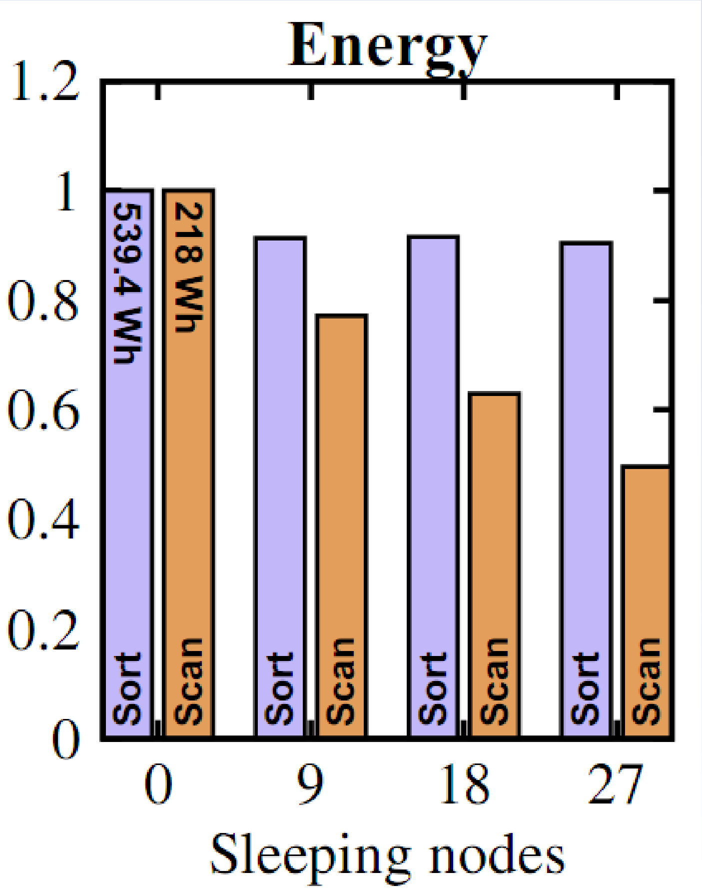
Machine On/Off

- Covering Set (CS) [5] ***“On the Energy (In)efficiency of Hadoop Clusters”***
- Hot Zone/Cold Zone (HC) [6] ***“GreenHDFS: Towards An Energy-Conserving, Storage-Efficient”***
- All-In Strategy(AIS) [7] “***Energy Management for MapReduce Clusters”***

Covering Set[5]

- At least one replica of a data-block must be stored in a subset of nodes referred as covering set.
- Large numbers of nodes can be gracefully turned off without affecting the availability of data.

Covering Set[5]



"On the Energy (In)efficiency of Hadoop Clusters"

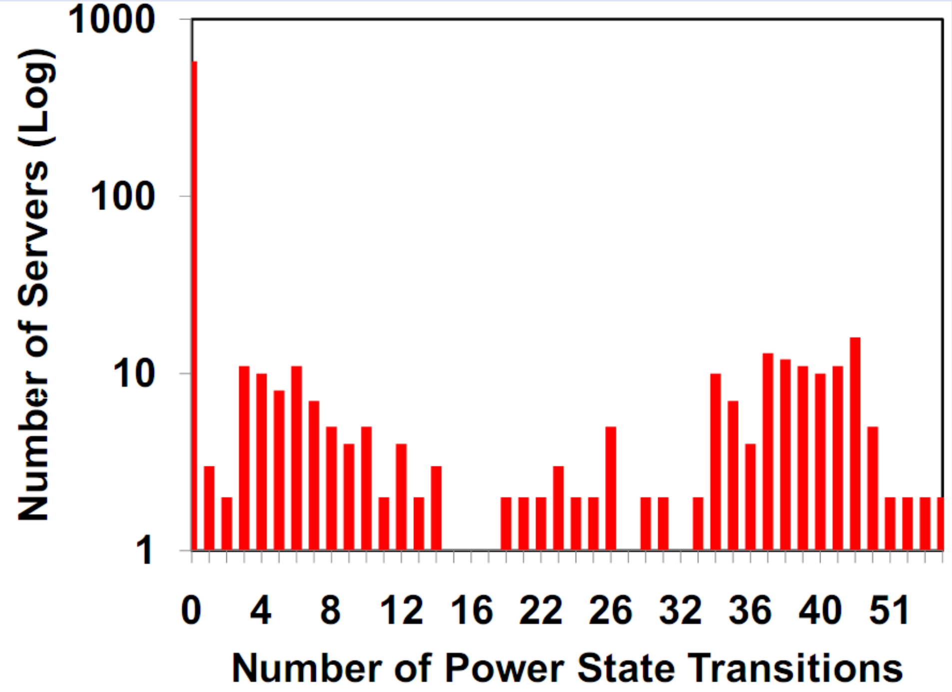
Hot Zone/Cold Zone[6]

- Classify data by its "temperature"
  - They use age of file, as defined by the last access to the file, as the measure of temperature of the file.
  - Hot files stored on high performance hot zone servers.
  - Cold files stored on cold zone servers with large storage space.

"GreenHDFS: Towards An Energy-Conserving, Storage-Efficient"

Hot Zone/Cold Zone[6]

- 26% energy savings within a three-month trace simulation.



illinois.edu

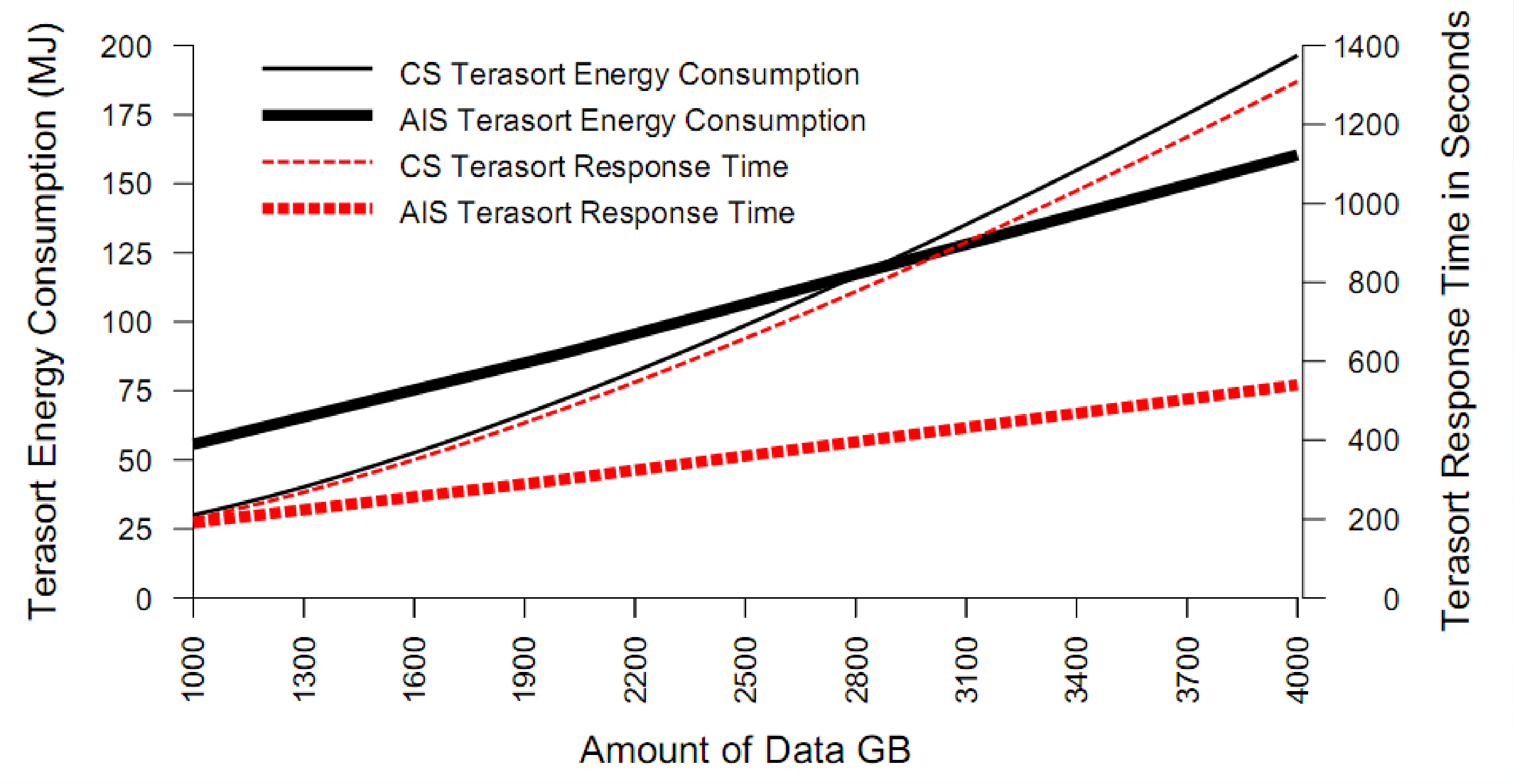
"GreenHDFS: Towards An Energy-Conserving, Storage-Efficient"

All In Service[7]

- In cases where there is a consistent low utilization period, AIS would batch the MR jobs in a queue.
- Periodically power up the entire system and run the entire batch of jobs as fast as they can and power off again.

illinois.edu

All In Service[7]



"Energy Management for MapReduce Clusters"

illinois.edu

Workload Ditribution

- Explicitly
  - Modify task assignment algorithm.
- Inexplicitly
  - Change frequency;
  - Modify data storage strategy;

illinois.edu

Hot Zone/Cold Zone[6]

- HC stores "hot" data on running and powerful machines.
- MapReduce will try to run a task locally first.
- Most task will be run on hot servers.

illinois.edu

Hot Zone/Cold Zone[6]

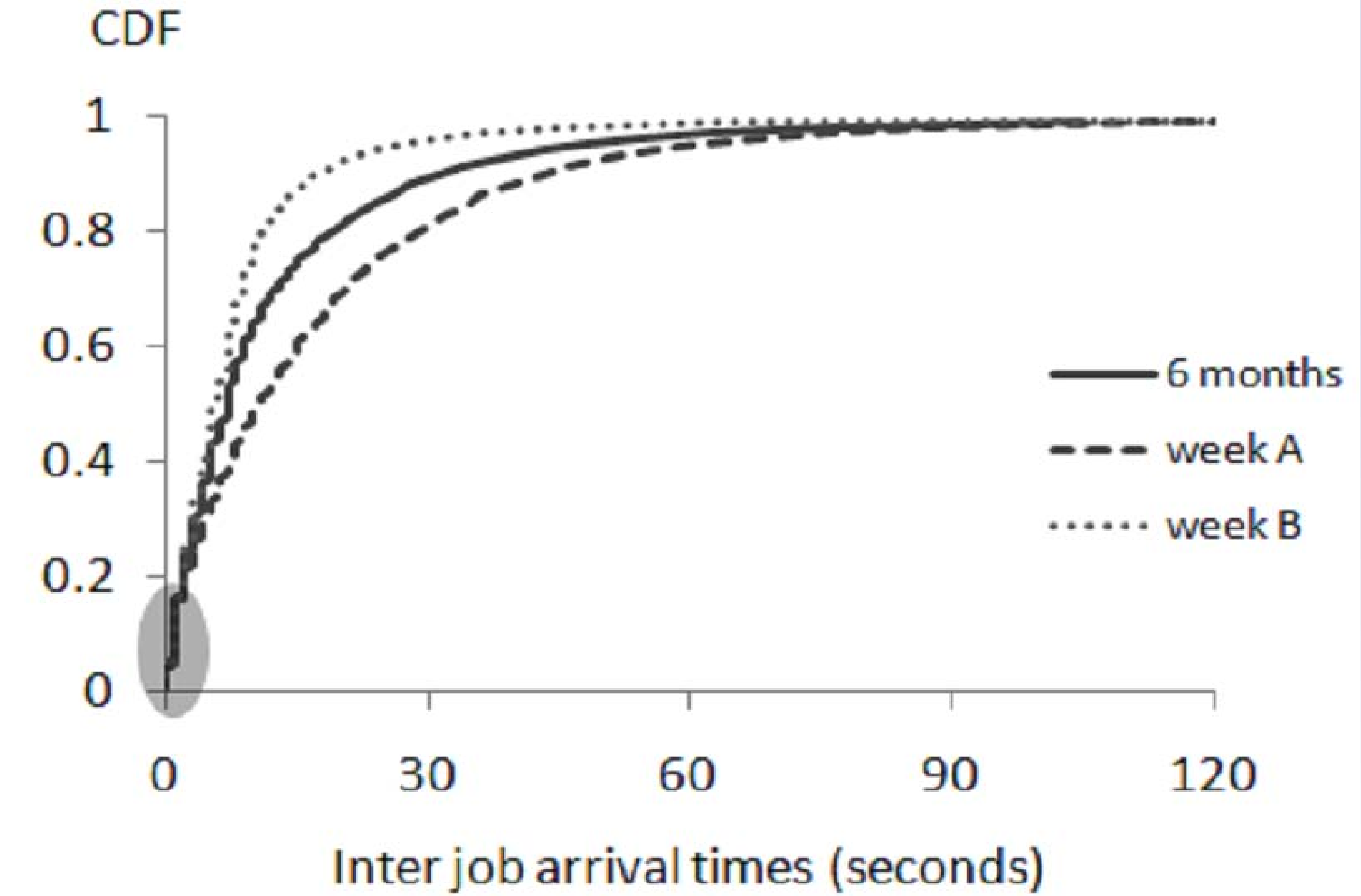
- HC stores “hot” data on running and powerful machines.
- MapReduce will try to run a task locally first.
- Most task will be run on hot servers.

Workload Generation

- Motivation
  - Company Competitive concerns
  - Better evaluation

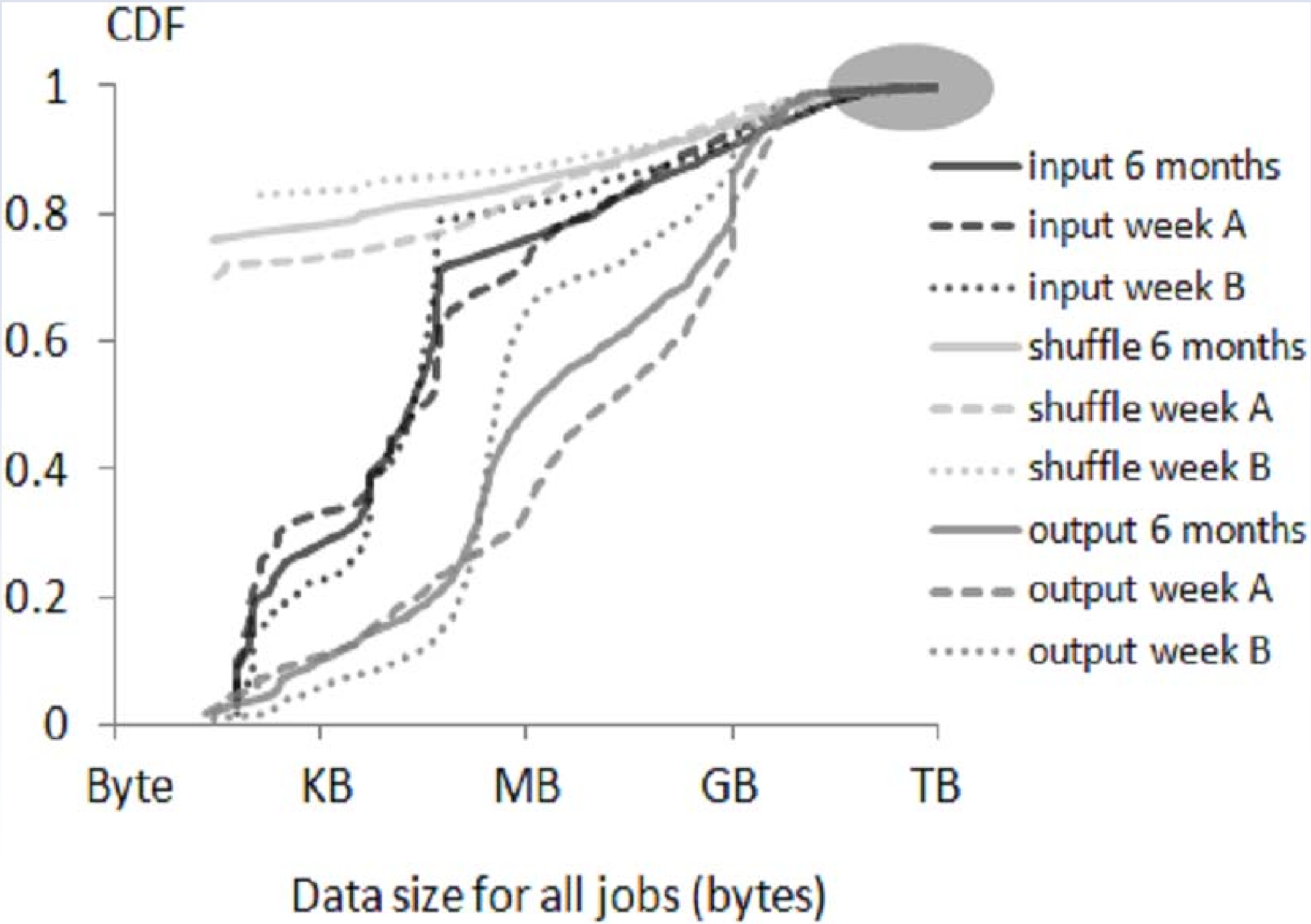
illinois.edu

Trace Analysis



illinois.edu

Trace Analysis



illinois.edu

Trace Statistics to Synthetic Workloads

- Simple non-parametric statistics are average and standard deviation.
  - Problem: Distributions are irregular, skewed, and asymmetric. Hence, averages and standard deviations are insufficient.
- Solution： use percentiles.
  - The authors choose percentiles based on Gaussian model;
  - Five-number summary: 1^{st}, 25^{th}, 50^{th}, 75^{th}, 99^{th},
  - Seven-number summary and so on.

illinois.edu

Thank you !

illinois.edu