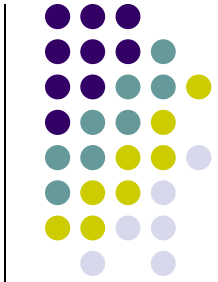
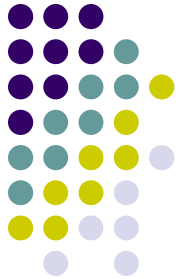


Towards Energy Proportional Cloud for Data Processing Frameworks

Hyeong S. Kim, Dong In Shin, Young Jin Yu,
Hyeonsang Eom, Heon Y. Yeom
Seoul National University

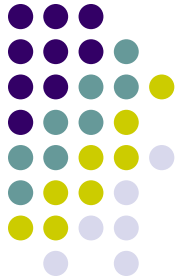
Introduction



- Recent advances in cloud computing is driving the heavy use of world-wide data centers.

- But, the cost of operating data center is rapidly increasing.
  - Environmental Protection Agent (EPA) recently reported that 1.5% of the total US energy use in 2006 was used to power data centers.
  - It is expected to nearly double by 2010.

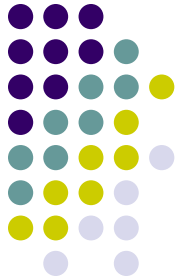
Introduction



- Amazon.com is facing highly increased power demand.
  - Hamilton(2009) reported that “the cost to power data centers” accounts for 59% of the total budget with three year amortizations.
  - He also says that power distribution is already fairly efficient.

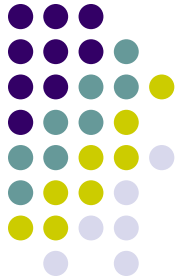
- Therefore, we should keep our attention on reducing the power delivered to the servers.

Introduction



- Fortunately, there are still much room to reduce the power consumption in various ways.
  - Barroso et al.(2007) proposed the concept of energy proportional computing.
    - Google's commodity servers lack the property.
  - DCEF(2007) reported that savings of the order of 20% can be achieved in server and network energy consumption.

From Energy Proportional Computer to Energy Proportional Cloud



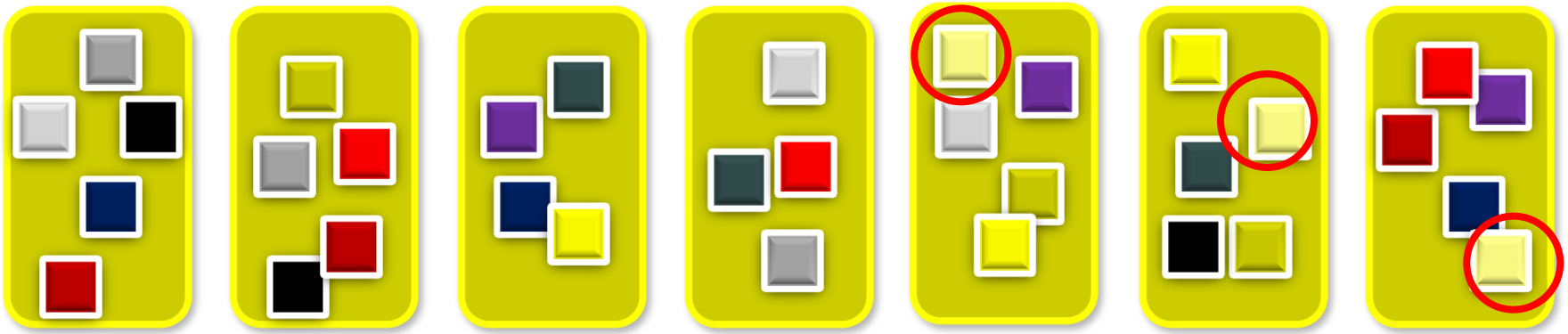
- Power save mode for cloud computing
  - We advocate **power down or suspending** method
- Service-level PSM
  - Each of the services provided by the data center has its own PSM
  - Advantages
    - Save the energy consumed by a single service by turning off some of the servers belonging to the service
    - Temporarily assign the suspended servers to the services which need more computing power

Motivating Example

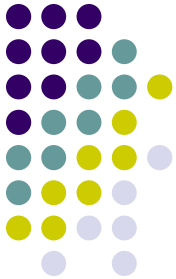
File chunks

Unavailable chunks + degraded performance (decreased data locality + reduced number of processing nodes)

Servers

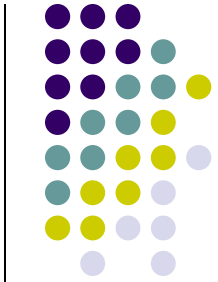


Motivating Example



- Two Problems
  - Data Unavailability
    - We may lose data during power save mode.
    - We have to consider the data placement policy before suspending some servers.
  - Performance Degradation
    - Suspended servers are not only used for the distributed storage, but also for the data processing.
    - But, the “very poor performance” can be problematic even if we want reduced power consumption at the cost of performance.

Related Work



- "Full coverage" by Harnik et al.(2009)
  - A method to choose candidate nodes to be suspended for generic distributed file systems
  - The problem of minimizing the number of unavailable files is NP-Complete.
  - They use heuristic ➔ a greedy algorithm
  - We name this as *postPSM* since they deal with the replicas after the system enters PSM (Reactive approach)

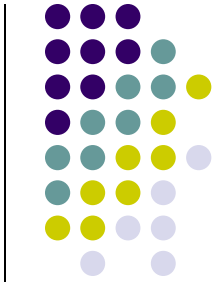
- "Covering subset" by Leverich et al.(2009)
  - At least one replica of a data-block must be stored in a subset of nodes.
  - We name this as *prePSM* since they construct a set of nodes a priori (Proactive approach)

Related Work

- Use low power machines in the data center
  - Cooperative Expendable Micro-Sliced Servers(CEMS)
    - Each server ➔ dual-core AMD, Mini-ITX board
    - Each sled ➔ 6 servers, 6 disks, 1 shared power supply
  - LinuxArmOrg
    - ARM-cpu servers running web servers
  - FAWN
    - A cluster of cost-effective components, e.g. low-power, efficient embedded CPUs and the flash storage

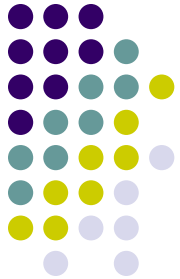
- They don't consider the hybrid design that utilizes both of high-end servers and low power ones.

Our Contribution



- We answer the following questions to enable PSM for the data processing frameworks
  - Is it reasonable to use low power computers instead of commodity servers during the power save mode?
    - We give a performance study of MapReduce with heterogeneous servers
  - Are there any practical challenges to enable power save mode for data processing frameworks?

1. Feasibility of Low Power Machines for D.P.F

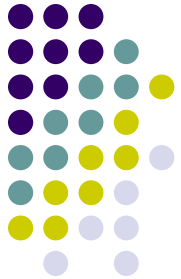


- Our primary concern is to augment high performance systems with low-power machines for D.P.F.

- The server class used in our evaluation

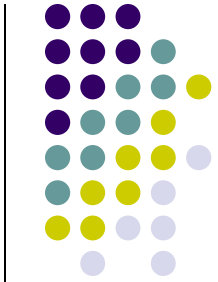
Name	*CPU*	*Cores*	*Memory*	*CPU TDP*	*Measured Power Consumption*	*Cost*	*Remarks*
Svr1	Intel Xeon X5450 3.00 GHz	2 x 4	16 GB DDR2	120 W	Peak/360 W, Idle/228 W	$3,200	pre-packaged server
Svr2	Intel Core2 Quad Q9550 2.83 GHz	4	8 GB DDR2	95 W	Peak/125 W, Idle/69 W	$1200	
Low1	Intel Atom 330 1.60 GHz	2	2 GB DDR2	8 W	Peak/33 W, Idle/25 W	$390	Zotac ION motherboard
Low2	Intel Atom Z530 1.60 GHz	1	1 GB DDR2	2 W	Peak/12 W, Idle/7 W	$360	fitPC2

1. Feasibility of Low Power Machines for D.P.F



- Simple Observations
  - Svr1 consumes more than 200W even if it is just sitting around.(→ bigger than Svr2's peak)
  - Low power nodes spend negligible amount of powers during idle time.
  - Low1 and Low2 contribute to space saving.
    - Low1: 215x210x55(mm), Low2: 101x115x27(mm)

1. Feasibility of Low Power Machines for D.P.F



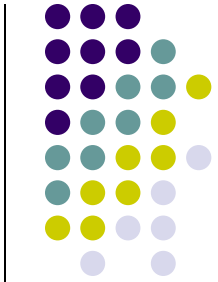
- MapReduce Performance
  - TeraSort (10GB), GridMix(streamSort, javaSort, dataScan, combiner, monsterQuery, webdataSort) for Small/Medium/Large dataset
  - Hadoop jobs on a single machine to study the performance of each server class
  - Calculate “running time” and “perf/watt”

1. Feasibility of Low Power Machines for D.P.F

	sort		gridmix	
	running time	Perf/Watt	running time	Perf/Watt
Svr1	1	1	1	1
Svr2	1.1	3.3	1.1	3.2
Low1	2.5	25.5	1.4	14.1
Low2	3.7	113.3	2.1	65.9

- Svr1 performs the best of all, but the difference bet'n Svr1 and Svr2 is very small.
- Although Low1 and Low2 increased the running time significantly, they are very power-efficient.

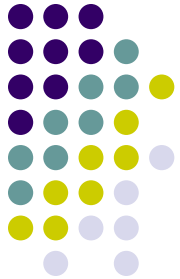
1. Feasibility of Low Power Machines for D.P.F



Environment	Normalized Running time	Normalized Perf/Watt
3 * Svr2 + 1 * Svr1	1	1
3 * Svr2 + 1 * Svr2	1.1	3.4
3 * Svr2 + 1 * Low1	1.3	18.2
3 * Svr2 + 1 * Low2	1.3	40.6

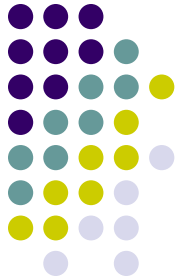
- gridmix benchmark
- The difference of running time is not significant and low power computers use power more effectively.
- We can indirectly show that replacing high end servers with low power ones does not incur significant performance degradation.

2. Practical Challenges



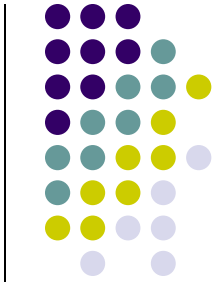
- Data Unavailability
  - Unavailable chunks lead to unavailable files
- Therefore, replica redistribution is needed to meet “replication factor” during PSM.
  - In our simulation, when we suspended 30% of the nodes, about 30% of the total chunks remain intact
  - This means 70% of the total chunks should be redistributed

2. Practical Challenges



- Simulation study
  - We simulated the data placement algorithm of HDFS (rack-aware replica placement)
  - We setup 16 nodes of two clusters (8 nodes per cluster)
  - In the simulation, we generated a fileset of 318GB and placed the file chunks according to the rack-aware replica placement
  - After that, we randomly suspended 30% of the nodes (4 nodes) and measured the number of remaining replicas of all the file chunks
  - On average,
    - 3-replicas : about 32% of chunks
    - 2-replicas : about 47% of chunks
    - 1-replica : about 19% of chunks
    - 0-replica : about 2% of chunks

2. Practical Challenges



- We also varied
  - The number of files of the fileset
  - The number of chunks of each file
- The results are similar
- Can we exploit this in replica redistribution?

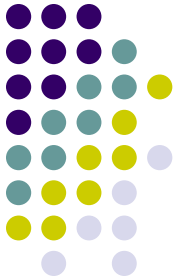
Table 2 The ratio of each chunk state for various numbers of input files. We randomly suspended 30% of nodes.

	30 files	*50 files*	*70 files*
0-replica	1.8 %	1.8 %	1.8 %
1-replica	19.8 %	19.2 %	18.3 %
2-replicas	46.8 %	47.1 %	48.5 %
3-replicas	31.6 %	31.9 %	31.4 %
unavailable /total files	7.2/30 (24.0%)	13.2/50 (26.4%)	18.1/70 (25.9%)

Table 3 The ratio of each chunk state for various numbers of chunks per file. We randomly suspended 30% of nodes.

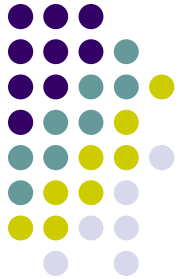
	16 chunks	*64 chunks*	*256 chunks*
0-replica	2.1 %	1.7 %	1.7 %
1-replica	18.0 %	18.2 %	18.2 %
2-replicas	47.9 %	48.6 %	48.7 %
3-replicas	32.0 %	31.5 %	31.5 %
unavailable /total files	14.8/50 (29.6%)	33.2/50 (66.4%)	49.1/50 (98.2%)

Efficient Replica Redistribution



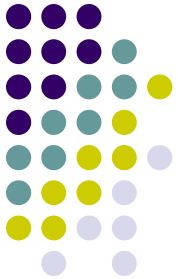
- We can allow decreased replication factor for some chunks
  - Chunks in 3-replicas state are complete
  - Chunks in 2-replicas state are relatively safe
  - Chunks in 1-replica are in potential danger
  - Chunks in 0-replica are in instant danger
- So, chunks in 0-replica and 1-replica had better be replicated instantly to reach the 2-replicas state
- When the state of a chunk reaches 2-replicas, we may force the chunk to stay in 2-replicas state
- Chunks already in 2-replicas state also maintain its state

Efficient Replica Redistribution



- In this way we can improve
  - The efficiency of replica redistribution
- Further optimization
  - The chunks in 2-replicas state can be replicated when the chunk is actually used by the MapReduce

Conclusion & Future Work



- We propose a Service-level PSM
- PSM for data processing frameworks is a challenging problem
- Future work
  - Candidate node set selection
  - We are implementing the power save mode for Hadoop