GeoSched: Minimizing Cost of Cloud Workloads in Geo-Distributed Data Centers

Anirudh Jayakumar and Harshit Dokania
University of Illinois at Urbana-Champaign
{ajayaku2,hdokani2}@illinois.edu

ABSTRACT

Today, many of the leading cloud service providers have geographically distributed data centers to serve millions of users around the world. The need for a geo-distributed data center arising from the need to provide low-latency and high-availability of the different services. The workloads that run on these data centers are also highly varied. A typical data centers run jobs of different kinds including business-critical workloads—web servers, mail servers, instant messaging services etc, big-data processing, real-time data analytics and HPC jobs. As cloud services grow to serve more customers, there is an increasing need for a workload provisioning mechanism which can minimize the cost of operation of the data center while still obeying the SLA and also minimizing the user-perceived latency.

We present GeoSched, a job provisioning framework that is workload aware, energy-aware and cooling-aware. GeoSched considers the workload type (batch process, real-time processing, web-service), the energy source of the data center (brown and green energy), electricity pricing of the region, cooling techniques used in the data center (air economizer) and also the history of jobs to predict the load patterns and cluster utilization. We wish to evaluate GeoSched on industry workload traces and also on synthetic workloads that can mimic the real-world data center workload.

1. RELATED WORK

Cost minimization has been extensively studied both in the context of a single data center and a geo-distributed data center. Most of the work has been targeted towards minimizing the energy consumption of the data center to reduce the operational cost. For example, the work by Gu et al. [1] studies the influence of task assignment, data placement and data movement on the operational expenditure of large-scale geo-distributed data centers for big data applications. The authors jointly optimize these three factors by proposing a 2-D Markov chain to derive the average task completion time and solve the model as a MILP problem. On similar lines Agarwal et al. [2] propose an automated data placement mechanism Volley for geo-distributed cloud services with the consideration of WAN bandwidth cost, data center capacity limits, data inter-dependencies, etc.

Yu et all.[3] propose minimizing energy cost for distributed Internet data centers (IDCs) in smart microgrids by considering system dynamics like uncertainties in electricity price, workload, renewable energy generation, and power outage state. They model the problem as a stochastic program and solve it using Lyapunov optimization technique. The work by Ren and He [4] propose an online algorithm, called COCA to minimize data center operational cost while satisfying carbon neutrality. Unlike some of the existing research, COCA enables distributed server-level resource management: each server autonomously adjusts its processing speed and optimally decides the amount of workloads to process. This work is restricted to a single data center with heterogeneous resources.

Cooling energy is a substantial part of the total energy spent by a data center. According to some reports, this can be as high as 50% [5], [6],[7] of the total energy budget. There has been a lot of work both at the application/middleware level [8], [9] as well as provisioning techniques [10], [11] that try to minimize the cooling energy. Li et all. [12] proposes SmartCool, a power optimization scheme that effectively coordinates different cooling techniques and dynamically manages workload allocation for jointly optimized cooling and server power. Unlike existing work that addresses different cooling techniques in an isolated manner, SmartCool integrates different cooling systems as a constrained optimization problem. Also, since geo-distributed data centers have different ambient temperatures, SmartCool dynamically dispatches the incoming requests among a network of data centers with heterogeneous cooling systems to best leverage the high efficiency of free cooling.

2. CLOUD WORKLOAD

With the increase in the adoption of cloud computing by businesses around the world, more and more workloads are moving out of traditional data centers into the cloud. A cloud data center typically handles a range of IT workloads. These workloads include service-critical and latency sensitive interactive applications that run 24x7 such as instant messengers, e-mail applications and other internet services, and batch-style applications such as scientific HPC applications, BigData applications like financial analysis and MapReduce applications. A thoughtful management of these workloads are necessary not only to provide customers with low-latency services but also to reduce the energy consumption of the data centers that run these workloads. A good way to study these workloads are by analyzing real-world or synthetic workload traces. We look at workload traces in detail in section 2.1 and 2.3

2.1 Workload Traces

Unfortunately, there is a lack of real-world traces in the cloud computing research community. As a result, there are limited work done to understand the diversity in cloud workloads. The first large-scale workload trace was released by Google in 2011[13], featuring traces from over 12,000 servers over a period of 29 days. Various studies have been carried out to understand the characteristic of the Google trace data. Other real-world traces include Wikipedia request trace [14] and Hadoop logs from Yahoo! [15]There have also been attempts to generate synthetic workload traces using known real-world workload characteristics [16] [17]. For the sake of this work, we have decided to use the workload trace from Google.

2.2 Custom Trace Format

To have a common trace format to represent different real-world and synthetic workload traces, we have formulated a trace format similar to standard workload format (swf) [18] defined for grid and supercomputing traces. Table 1 provides information about each field int the trace format.

Table 1: Trace format description

Field Name	**Type**	**Description**
JobID	INTEGER	Unique job identifier for each job
Arrival Time	INTEGER	Arrival time in microseconds
Scheduled Time	INTEGER	Arrival time in microseconds
Finish Time	INTEGER	Arrival time in microseconds
Schedule class	INTEGER	Latency class of the job.
Tasks	INTEGER	Total number of tasks in the job.
Total CPU	INTEGER	Aggregate sum of CPU used by all the tasks in the job
Total Memory	INTEGER	Aggregate sum of memory used by all the tasks in the job
User	INTEGER	Unique user identifier
SLA Param1	FLOAT	SLA parameter placeholder
SLA Param2	FLOAT	SLA parameter placeholder
SLA Param3	FLOAT	SLA parameter placeholder

2.3 Google Workload Trace

Google trace has job and task information in different files. We consolidated all the required information and generated a single work trace file with the format described in the previous section. For our work, we model five different data centers located at different parts of the globe. A detailed explanation about the data center profile are given in section 3. To simulate workload execution in these five data centers we have extracted five set of traces each worth five days of incoming workload from the original google trace. So, this covers 25 days of traces out of the 29 available days. We label these 5 set of traces from A to E. Additionally, in our experiments we wish to only consider those jobs that end without any failures. Therefore, we clean the original traces by removing all jobs that is either failed or get killed.

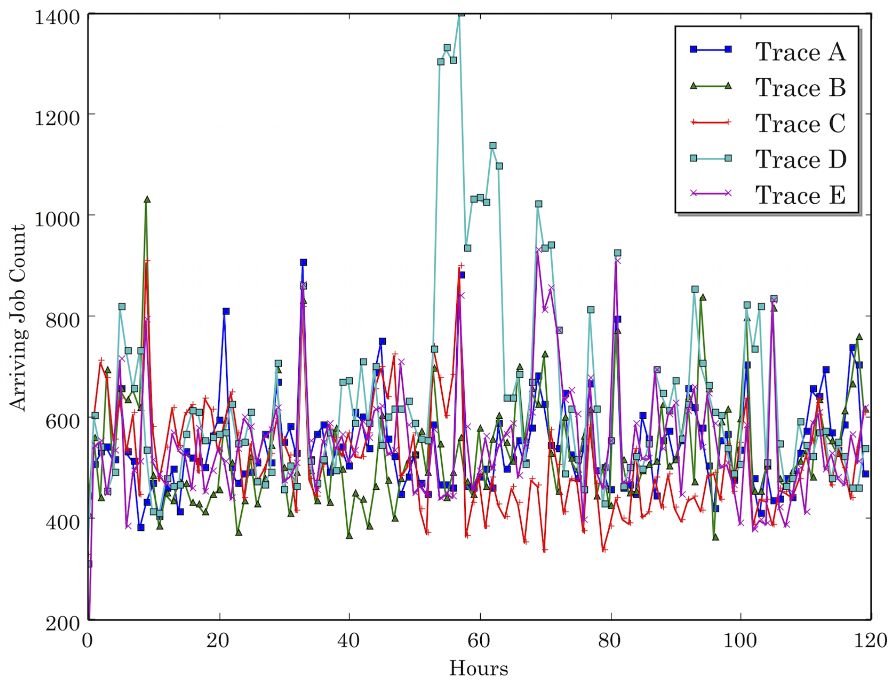


Figure 1: Arrival rate of jobs

Now we will look at some characteristics of the traces. Table 2 describes the statistics for each of the five traces grouped by the scheduling class of the job. In google traces, CPU and memory usage are normalized relative to the largest capacity of the resource on any machine in the trace (which is 1.0). Most of the class 0 and class 2 jobs are batch jobs with less latency requirements. Generally, these are the kind of jobs that can be provisioned on other data centers as their deadlines are not strict. You will find that as the sensitivity increases the running time of these jobs also increase. High-sensitive jobs are generally service jobs and run of longer periods of time. Figure 2 shows a pie chart of the job counts in different schedule classes for all the 5 traces. You will notice that the job count for class 3 (high latency-sensitivity) jobs are low. This is because these jobs generally run for long period of time, typically months. Since we only consider jobs that start and end within the 29 day range such jobs are missed out in our traces. For our work, we will consider class-0 and class-1 jobs as candidates for optimizing the provisioning. We leave, class-2 and class-3 jobs to run on the data center where the job arrived since these are high-sensitive and their SLA demands will be strict.

Figure 1 shows the variation in job arrival for the traces. As you can see there is both intra-trace variation as well as inter-trace variation. This is representative of the real-world cloud workload pattern where bursty arrivals are common. There are opportunities to provision jobs to other data centers if the traffic at a particular data center is high. What is also important to notice is that the scheduling decision should be made quickly when the arrival rates of jobs are high, else there are chances for these jobs to miss the SLA. Hence, even the scheduler should be scalable during high volume traffic.

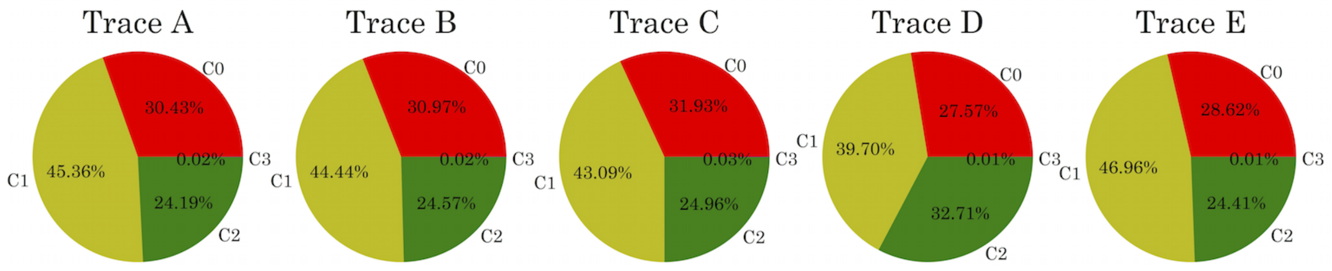


Figure 2: Distribution of job count in various schedule classes

Table 2: Trace set statistics

	Trace A			**Trace B**			**Trace C**			**Trace D**			**Trace E**		
Sched Class	**Avg CPU**	**Avg Mem**	**RT (secs)**	**Avg CPU**	**Avg Mem**	**RT (secs)**	**Avg CPU**	**Avg Mem**	**RT (secs)**	**Avg CPU**	**Avg Mem**	**RT (secs)**	**Avg CPU**	**Avg Mem**	**RT (secs)**
0	0.075	0.051	415	0.123	0.096	345	0.083	0.065	385	0.093	0.092	373	0.093	0.074	333
1	0.087	0.029	729	0.086	0.032	760	0.082	0.028	711	0.073	0.030	743	0.093	0.043	693
2	0.067	0.043	1028	0.096	0.058	872	0.088	0.057	918	0.073	0.045	707	0.084	0.049	832
3	0.088	0.12	1105	0.099	0.12	82	0.101	0.012	7961	0.081	0.051	17135	0.089	0.061	9021

Figure 3 shows the CDF of the job runtime. As you can see most of the jobs have runtime less than 1024 seconds. This pattern is consistent for all the trace sets. There are a few jobs that have higher run-times. As explained earlier, these jobs are primarily internet-service jobs that run for months.

In google workload trace, it is tough to study the resource usage since these value are scaled down and therefore and accurate picture of the resource utilization is tough to generate. In figure 4 we can see that

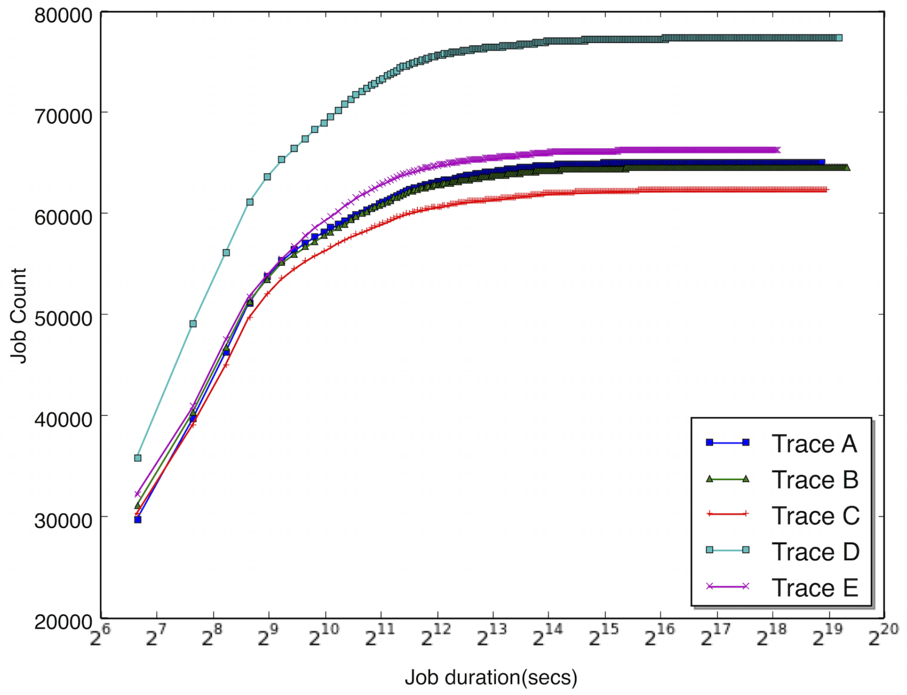
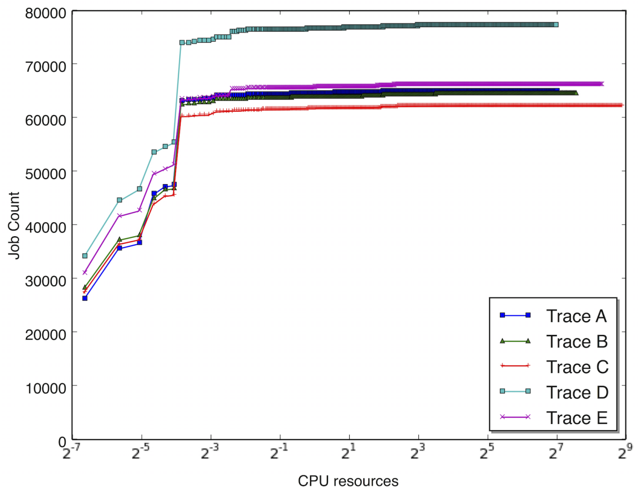
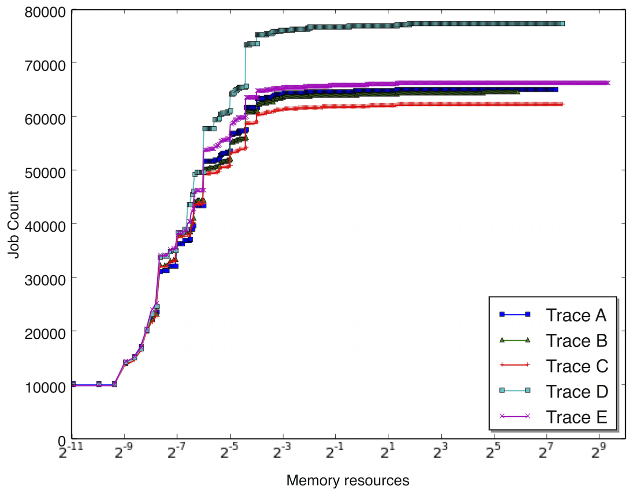


Figure 3: CDF of the job duration



(a) CPU resource



(b) RAM resource

Figure 4: CDF of resource requested by jobs

most jobs request a small portion of the cluster resources. Most of the resources in the google cluster are used by the few service jobs that are present. The large number of batch jobs only use a small percentage of the resources [19] [20]. From the above statistics, we believe that google workload trace provides an accurate representation of the real-world cloud datacenter workload.

3. DATA CENTER PROFILING

Provide some introduction about out profiling stratergy

3.1 Geo-Distributed Data Center

3.2 Profile Parameters

3.2.1 Temperature

3.2.2 Electricity Pricing

3.2.3 Energy Source

3.3 Air Economizer

4. SCHEDULER SIMULATOR

Provide basic idea about our strategy

4.1 Architecture

Introduce cloudsim

4.1.1 Components

4.1.2 Integration of traces and data center profile

4.1.3 Design

5. SCHEDULING ALGORITHM

Provide basic idea about our strategy

5.1 Features considered

5.1.1 Energy Cost Model

6. MILESTONES

Item	Deadline	Description
Study real-world and synthetic workloads	10th March	We have started studying workloads traces for google cluster-data which provides traces from a 12k-machine cell for one month duration in May 2011.
Data Center profile	20th March	In this process we will note the data center specific parameters such as electricity cost, available energy, economizer, cooling energy that is useful for simulating geo-distributed data centers.
Simulator design and implementation	12th April	Implement the simulator infrastructure for multi-datacenter environment.
Feature Extraction and Optimization	25th April	Here we identifying all the necessary features that contribute to the energy cost and build an optimization technique which is adaptive to workload characteristics and data center's resource capacity and electricity costs
Evaluation	7th May	This phase we will be simulating data centers with previously gathered data center profile and using the optimization technique mentioned above

References

[1] Gu, L., Zeng, D., Li, P., and Guo, S., "Cost minimization for big data processing in geo-distributed data centers," (2014).

[2] Agarwal, S., Dunagan, J., Jain, N., Saroiu, S., Wolman, A., and Bhogan, H., "Volley: Automated data placement for geo-distributed cloud services.," in [*NSDI*], 17–32 (2010).

[3] Yu, L., Jiang, T., and Cao, Y., "Energy cost minimization for distributed internet data centers in smart microgrids considering power outages," (2015).

[4] Ren, S. and He, Y., "Coca: Online distributed resource management for cost minimization and carbon neutrality in data centers," in [*Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*], 39, ACM (2013).

[5] Sullivan, R., "Alternating cold and hot aisles provides more reliable cooling for server farms, a white paper from the uptime institute," *Incorporated, Santa Fe, NM* (2002).

[6] Patel, C. D., Bash, C. E., Sharma, R., Beitelmal, M., and Friedrich, R., "Smart cooling of data centers," in [*ASME 2003 International Electronic Packaging Technical Conference and Exhibition*], 129–137, American Society of Mechanical Engineers (2003).

[7] Sawyer, R., "Calculating total power requirements for data centers," *White Paper, American Power Conversion* (2004).

[8] Sarood, O. and Kalé, L. V., "A 'cool' load balancer for parallel applications," in [*Proceedings of the 2011 ACM/IEEE conference on Supercomputing*], (November 2011).

[9] Leverich, J. and Kozyrakis, C., "On the energy (in) efficiency of hadoop clusters," *ACM SIGOPS Operating Systems Review* **44**(1), 61–65 (2010).

[10] Tang, Q., Gupta, S. K., and Varsamopoulos, G., "Thermal-aware task scheduling for data centers through minimizing heat recirculation," in [*Cluster Computing, 2007 IEEE International Conference on*], 129–138, IEEE (2007).

[11] Chen, Y., Gmach, D., Hyser, C., Wang, Z., Bash, C., Hoover, C., and Singhal, S., "Integrated management of application performance, power and cooling in data centers," in [*Network Operations and Management Symposium (NOMS), 2010 IEEE*], 615–622, IEEE (2010).

[12] Li, L., "Coordinating liquid and free air cooling with workload allocation for data center power minimization," in [*11th International Conference on Autonomic Computing (ICAC 14)*], USENIX Association (2014).

[13] "Google cluster data." `https://code.google.com/p/googleclusterdata/`. Accessed: 2015-3-30.

[14] Urdaneta, G., Pierre, G., and Van Steen, M., "Wikipedia workload analysis for decentralized hosting," *Computer Networks* **53**(11), 1830–1845 (2009).

[15] "Computing system data." `http://webscope.sandbox.yahoo.com/catalog.php?datatype=s`. Accessed: 2015-3-30.

[16] Beitch, A., Liu, B., Yung, T., Griffith, R., Fox, A., and Patterson, D. A., "Rain: A workload generation toolkit for cloud computing applications," *Electrical Engineering and Computer Sciences University of California at Berkeley, White paper UCB/EECS-2010-14* (2010).

[17] Wang, G., Butt, A. R., Monti, H., and Gupta, K., "Towards synthesizing realistic workload traces for studying the hadoop ecosystem," in [*Modeling, Analysis & Simulation of Computer and Telecommunication Systems (MASCOTS), 2011 IEEE 19th International Symposium on*], 400–408, IEEE (2011).

[18] Chapin, S. J., Cirne, W., Feitelson, D. G., Jones, J. P., Leutenegger, S. T., Schwiegelshohn, U., Smith, W., and Talby, D., "Benchmarks and standards for the evaluation of parallel job schedulers," in [*Job Scheduling Strategies for Parallel Processing*], 67–90, Springer (1999).

[19] Mishra, A. K., Hellerstein, J. L., Cirne, W., and Das, C. R., "Towards characterizing cloud backend workloads: insights from google compute clusters," *ACM SIGMETRICS Performance Evaluation Review* **37**(4), 34–41 (2010).

[20] Reiss, C., Tumanov, A., Ganger, G. R., Katz, R. H., and Kozuch, M. A., "Heterogeneity and dynamicity of clouds at scale: Google trace analysis," in [*Proceedings of the Third ACM Symposium on Cloud Computing*], 7, ACM (2012).