Cutting Down the Energy Cost of Geographically Distributed Cloud Data Centers

Huseyin Guler[1], B. Barla Cambazoglu[2] and Oznur Ozkasap[1]

[1] Koc University, Istanbul, Turkey
[2] Yahoo! Research, Barcelona, Spain

Abstract. The energy costs constitute a significant portion of the total cost of cloud providers. The major cloud data centers are often geographically distributed, and this brings an opportunity to minimize their energy cost. In this work, we model a geographically distributed data center network that is specialized to run batch jobs. Taking into account the spatio-temporal variation in the electricity prices and the outside weather temperature, we model the problem of minimizing the energy cost as a linear programming problem. We propose various heuristic solutions for the problem. Our simulations using real-life workload traces and electricity prices demonstrate that the proposed heuristics can considerably decrease the total energy cost of geographically distributed cloud data centers, compared to a baseline technique.

1 Introduction

Cloud data centers provide massive computing power to serve a large amount of tasks generated by the Internet services as well as IT industries. While processing incoming computational tasks, cloud service providers also need to satisfy certain service level agreements (SLAs). In order to meet the requirements of their SLAs, cloud providers geographically distribute their data centers around the world. The data centers' high computing power requires significant energy for both running the computing resources and also cooling them. The cost of energy constitutes an important fraction of the total operational costs of distributed data centers [1]. Therefore, service providers constantly chase novel methods to reduce their energy cost.

Reducing the energy consumption of large-scale distributed systems has recently been a hot research topic [1]. Some studies [2, 4, 9] focus on server consolidation by moving virtual machines across data centers and also consider SLA penalties if the deadline of a certain job is not satisfied [2, 9]. Le et al. [8] and Gao et al. [3] concentrate on the greenness aspect of data centers and allow service providers to trade off between the electricity cost and the carbon footprint. Certain studies investigate the data center cooling problem and propose solutions based on workload placement [11, 17] and thermal storage [5, 18]. In recent years, researchers have also investigated the impact of spatio-temporal electricity price variations on financial cost savings [6, 7, 10, 12–16, 19, 20] All these works either do not consider the SLA requirements or ignore cooling related costs. In this

Table 1. System parameters

Description	Symbol
Total energy cost of servers in DC i (\$)	E_i^{IT}
Total energy cost of DC i (\$)	E_i^{total}
Total penalty paid in DC i	Pen_i
Length of a unit time slot	u
Electricity price of DC i in time slot $[t, t+u)$	$E_i(t)$
Number of DC	N
Number of different type of servers	K
Set of servers in DC i	s_i
Set of type k servers in DC i	$s_{i,k}$
Number of CPU cores that a type k server has	c_k
CPU frequency of a type k server	f_k
Power consumption of a type k server when idle (Watt/u)	P_k^{idle}
Power consumption of a type k server at peak (Watt/u)	P_k^{peak}
Power consumption of a type k server s in time slot $[t, t+u)$	$P_{k,s,t}$
Number of jobs	J
Number of CPUs job j requires	c_j
Total number of timeslots job j requires	l_j
CPU frequency job j requires	f_j
Submission time of job j	Ts_j
Deadline of job j	Td_j
Penalty of late delivery of job j	Pen_j
Percentage of time that SLA of job j is violated	$\sigma(j)$
Number of time slots required to process job j in server s	$T_{j,s}$

paper, we propose electricity-cost-aware request dispatching algorithms which also consider SLA related penalties.

2 Problem Specification

In this section, we formally state our linear optimization problem. Table 1 lists the parameters and system variables used in the formulation.

We calculate the performance coefficient (CoP) of a data center according to its outside weather temperature. We set the CoP of the hottest data center (26 °C) as 2.0 (this means 1W energy is needed to cool down the data center for each 1W of IT job), and for the coldest data center (−9 °C) as 1.2, which is similar to the CoP of the currently existing energy-efficient cloud data centers [1].

The performance coefficient of DC i with temperature T_i:

$$CoP(T_i) = 1.2 + 0.128 \times \sqrt{T_i + 9} \tag{1}$$

The decision variable indicating whether server s is busy operating in time slot $[t, t+u)$:

$$x_{s,t} = \begin{cases} 1, & \text{if server } s \text{ is busy operating} \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

Power consumption of a type k server [14]:

$$P_{k,s,t} = P_k^{\text{idle}} + (P_k^{\text{peak}} - P_k^{\text{idle}}) \times x_{s,t} \tag{3}$$

$$E_i^{\text{IT}} = \sum_{k=1}^{K} \sum_{s \in s_{i,k}} \sum_{t=0}^{\lfloor T/u \rfloor} P_{k,s,t} \times E_i(t) \tag{4}$$

Total cost of data center i:

$$E_i^{\text{total}} = E_i^{\text{IT}} \times CoP(T_i) \tag{5}$$

Decision variable indicating whether job j is dispatched to server s:

$$r_{j,s} = \begin{cases} 1, & \text{if job } j \text{ is assigned to server } s \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

$$T_{j,s} = \frac{c_j \times f_j \times l_j}{c_k \times f_k}, \text{where } s \text{ is a type } k \text{ server} \tag{7}$$

Percentage of time that SLA of job j is violated [4]

$$\sigma(j,s,t) = (Ts + T_{j,s} - Td_j)/T_{j,s} \tag{8}$$

$$Pen_{j,s,t} = P_{k,s,t} \times E_i(t) \times \frac{Gom(\sigma(j,s,t)))}{100} \tag{9}$$

$$Pen_i = \sum_{j=1}^{J} \sum_{s \in s_i} \sum_{t=0}^{\lfloor T/u \rfloor} r_{j,s,t} \times Pen_{j,s,t} \tag{10}$$

Our objective is to minimize:

$$\sum_{i=1}^{N} E_i^{\text{total}} + Pen_i, \tag{11}$$

subject to

$$s_i = \sum_{k=1}^{K} s_{i,k} \tag{12}$$

In other words, our goal is to minimize the cost of a provider during the given time period while respecting QoS requirements (this includes financial penalties if SLA is violated).

3 Proposed Algorithms

We have a central scheduler that receives the incoming jobs and then forwards each job to one of the idle servers. Our proposed algorithms take advantage of three important factors:

- Spatial electricity price variation.
- Temporal electricity price variation.
- Reduced cooling cost in cooler places due to evaporation.

Considering these factors, we propose two types of request dispatching algorithms. In the first type of algorithms, each incoming job request is immediately scheduled to an available server. The algorithms of the second type can schedule jobs ahead of time within a time window if they "forecast" that the electricity price will be lower in the future.

3.1 Immediate Scheduling Algorithms

The jobs are immediately scheduled in FCFS order. The following two algorithms differ in the way they decide on which server to assign a job.

Cheapest data center (`CheapestDC`): The current electricity prices of all data centers are checked and a random server is selected from the cheapest data center. If all servers in the cheapest data center are busy, then a server from the second cheapest data center is selected. The procedure continues until the job is scheduled. If all servers are busy operating at that time then the job is put in the queue again to be scheduled in the next time slot. Only the spatial electricity price change is exploited in this simple greedy heuristic.

Cheapest server (`CheapestS`): Different than the `CheapestDC` algorithm, this algorithm takes advantage of the outside weather temperature associated with the data centers. The assumption is that the total cost of running a job in a server in cooler locations can be cheaper even if the server is not located in the cheapest data center. The algorithm runs the job on the server with the lowest expected total cost.

3.2 Delayed Scheduling Algorithms

`CheapestDC` and `CheapestS` aim at scheduling the jobs in the current time slot. However, it is possible to postpone the execution of a job to future time slots if we can somehow identify that the current electricity is expensive. To this end, we can use historical electricity prices to determine whether to schedule the job immediately or delay its execution to a later time slot. As in the `CheapestS` algorithm, we examine every server for all time slots within a predefined time window and select the best server and time slot combination in terms of the electricity cost. In this case, we utilize all three variations including spatial and temporal electricity price fluctuations as well as the reduced cooling cost of the servers located in colder climates.

In addition to the novel time window approach, we also implemented an internal ordering of the jobs in this algorithm and observed the effect of the ordering on the final performance. These orderings are first come first serve (FCFS), longest job first (LJF), and shortest job first (SJF). Variations of these algorithm are named as `WindowFCFS`, `WindowLJF`, and `WindowSJF`.

Note that, in this algorithm, the scheduled time slots are fixed, i.e., we do not reschedule any job even if it is assigned to a future time slot. As a future work, we will also implement an algorithm with periodic rescheduling and compare its performance with the other solutions.

4 Simulation Setup

We simulated a geographically distributed data center network to evaluate the performance of the proposed algorithms. The simulator and the algorithms are implemented in Java. We only consider delay-tolerant batch jobs and use the Grid5000 logs for incoming job requests.[3] The job requests contain job specifications including submission time, runtime, required number of CPU cores, and similar information. Our problem formulation is generalizable to heterogeneous data centers. However, for the initial results we present here, we consider only the homogeneous data center scenario. We simulated six homogeneous data centers that are located in San Diego, California; Chicago, Illinois; Santiago, Chile; Helsinki, Finland; Dublin, Ireland and Singapore, Singapore. Each data center is given 100 servers with Xeon architecture and four core CPUs running at 2.66 GHZ [4]. We used real electricity price traces for San Diego and Chicago from FERC (Federal Energy Regulatory Commission of the USA),[4] and scaled the prices for other countries according to the country-wide average prices.[5] Average temperatures values are gathered from Wunderground.[6]

We simulated the network under three different loads: `light`, `medium`, and `heavy`. The Grid5000 data spans three years of job requests; however, we run our simulations on a weekly basis. The original log corresponds to the `light` workload. Thus, we scaled down the submission times of the jobs that belong to future weeks to construct the `medium` and `heavy` workloads. Moreover, our delayed scheduling algorithm is run under different time window values, that are 6, 12, and 24 hours to determine the best fit of the time window.

At this stage of our work, we have not yet introduced the penalty concept for the jobs and the results presented in Section 5 do not include any penalty-related cost. As a baseline, we also implemented a random request dispatcher

---

3 Grid5000, `http://gwa.ewi.tudelft.nl/pmwiki/pmwiki.php?n=Workloads.Gwa-t-2`.

4 FERC: Electric Power Markets, `http://www.ferc.gov/market-oversight/mkt-electric/overview.asp`.

5 Wikipedia-Electricity Pricing, `http://en.wikipedia.org/wiki/Electricity_pricing`.

6 Wunderground, `http://www.wunderground.com/history`.

Table 2. Performance of immediate scheduling techniques ($I\%$ denotes the relative percent improvement w.r.t the random scheduling baseline while C_{P}, C_{C}, and C_{T} denote the processing, cooling, and total financial cost per job, respectively)

	`Random`				`CheapestDC`				`CheapestS`			
Load	C_{P}	C_{C}	C_{T}	$I\%$	C_{P}	C_{C}	C_{T}	$I\%$	C_{P}	C_{C}	C_{T}	$I\%$
`light`	18.9	14.0	32.9	–	18.3	13.4	31.8	3.4	18.4	13.3	31.6	3.8
`medium`	12.1	8.9	21.0	–	11.9	8.7	20.7	1.7	11.9	8.7	20.6	2.1
`heavy`	8.0	5.9	14.0	–	8.0	5.9	14.0	0.0	8.0	5.9	14.0	0.0

Table 3. Performance of delayed scheduling techniques (W denotes the window size)

		`WindowFCFS`				`WindowLJF`				`WindowSJF`			
Load	W	C_{P}	C_{C}	C_{T}	$I\%$	C_{P}	C_{C}	C_{T}	$I\%$	C_{P}	C_{C}	C_{T}	$I\%$
`light`	6	18.3	13.1	31.3	4.7	18.2	13.0	31.3	4.9	18.3	13.1	31.4	4.5
	12	18.3	13.0	31.2	5.0	18.2	13.0	31.2	5.2	18.3	13.0	31.3	4.7
	24	18.2	13.0	31.2	5.1	18.2	13.0	31.1	5.4	18.2	13.1	31.3	4.9
`medium`	6	11.8	8.6	20.3	3.3	11.8	8.5	20.3	3.5	11.8	8.6	20.4	3.1
	12	11.7	8.5	20.2	3.9	11.7	8.5	20.1	4.2	11.7	8.5	20.2	3.7
	24	11.7	8.4	20.1	4.5	11.6	8.4	20.0	4.7	11.7	8.4	20.1	4.3
`heavy`	6	8.0	5.9	13.8	0.9	8.0	5.9	13.8	1.0	7.9	5.8	13.8	1.4
	12	9.1	6.7	15.8	-13.1	9.8	7.2	17.1	-22.3	8.3	6.1	14.3	-2.8
	24	9.1	6.7	15.8	-13.0	9.4	6.9	16.2	-16.4	8.9	6.5	15.4	-10.2

(`Random`) that tries to balance the workload of the data centers. We compared the performance of the proposed algorithms against this baseline.

5 Results

As explained in Section 3, there are three factors that we consider: spatial and temporal electricity price change and outside weather temperature. By comparing the performance of the algorithms, we can determine the contribution of these factors to the cost saving. Tables 2 and 3 summarize our simulation results and the amount of improvement achieved by each algorithm. Most of the gain is achieved by exploiting the spatial electricity price variation as seen by the improvement of the `CheapestDC` algorithm. Next, the biggest improvement is achieved by algorithms with a time window that exploit the temporal electricity price change. The least improvement, but still significant, is by taking advantage of the reduced cooling cost of the servers located in colder climates.

In the `light` workload case, we have the flexibility to postpone the execution of a job since there are few jobs. This is also true for the `medium` workload. The current state-of-the-art data center systems work under medium workloads, where the system tries to keep the total utilization of the servers under a certain percentage, i.e., mostly around 35%-40%. When the simulation is run under the `heavy` workload, the number of jobs executed by each algorithm differs. Therefore, we presented these results in terms of cost per job. The number of jobs executed by delayed scheduling algorithms drops in the `heavy` workload case because some of the early time slots are not utilized by the algorithm, which forecasts some cheaper future time slots. In the end, we were left with many non-executed jobs in the job queue. In order to overcome this drawback, we are planning to propose a workload adaptive delayed scheduling algorithm that also utilizes the current time slots when the workload is heavy.

6 Conclusion

We proposed request dispatching algorithms that exploit spatio-temporal electricity price variations and reduced cooling cost opportunities in colder climates to minimize the energy cost of geographically distributed data centers. Our simulation results show that significant electricity cost reduction can be achieved by the proposed algorithms, compared to a random scheduler that aims to achieve only load balancing. As a future work, we plan to conduct experiments including the SLA penalty concept.

Acknowledgments. This work was partially supported by the COST (European Cooperation in Science and Technology) framework, under Action IC0804: Energy efficiency in large scale distributed systems, and by TUBITAK (The Scientific and Technical Research Council of Turkey) under Grant 109M761.

References

1. L. A. Barroso and U. Hölzle. *The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines.* Morgan and Claypool Publishers, 1st edition, 2009.
2. N. Buchbinder, N. Jain, and I. Menache. Online job-migration for reducing the electricity bill in the cloud. In *Proceedings of the 10th International IFIP TC 6 Conference on Networking - Volume Part I*, pages 172–185, 2011.
3. P. X. Gao, A. R. Curtis, B. Wong, and S. Keshav. It's not easy being green. In *Proceedings of the ACM SIGCOMM 2012 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, pages 211–222, 2012.
4. I. Goiri, J. L. Berral, J. O. Fitó, F. Julií, R. Nou, J. Guitart, R. Gavaldí, and J. Torres. Energy-efficient and multifaceted resource management for profit-driven virtualized data centers. *Future Gener. Comput. Syst.*, 28(5):718–731, 2012.
5. Y. Guo, Z. Ding, Y. Fang, and D. Wu. Cutting down electricity cost in internet data centers by using energy storage. In *Global Telecommunications Conference*, pages 1–5, 2011.
6. E. Kayaaslan, B. B. Cambazoglu, R. Blanco, F. P. Junqueira, and C. Aykanat. Energy-price-driven query processing in multi-center web search engines. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 983–992, 2011.
7. K. Le, R. Bianchini, M. Martonosi, and T. D. Nguyen. Cost- and energy-aware load distribution across data centers. In *Workshop on Power Aware Computing and Systems*, 2009.

8. K. Le, O. Bilgir, R. Bianchini, M. Martonosi, and T. D. Nguyen. Managing the cost, energy consumption, and carbon footprint of internet services. In *Proceedings of the ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, pages 357–358, 2010.
9. J. Li, Z. Li, K. Ren, and X. Liu. Towards optimal electric demand management for internet data centers. *IEEE Trans. Smart Grid*, 3(1):183–192, 2012.
10. S. Mani and S. Rao. Operating cost aware scheduling model for distributed servers based on global power pricing policies. In *Proceedings of the 4th Annual ACM Bangalore Conference*, 2011.
11. J. Moore, J. Chase, P. Ranganathan, and R. Sharma. Making scheduling "cool": temperature-aware workload placement in data centers. In *Proceedings of the Annual Conference on USENIX Annual Technical Conference*, pages 5–5, 2005.
12. A. Qureshi, R. Weber, H. Balakrishnan, J. Guttag, and B. Maggs. Cutting the electric bill for Internet-scale systems. In *Proceedings of the ACM SIGCOMM 2009 Conference on Data Communication*, pages 123–134, 2009.
13. L. Rao, X. Liu, M. Ilic, and J. Liu. MEC-IDC: joint load balancing and power control for distributed Internet data centers. In *Proceedings of the 1st ACM/IEEE International Conference on Cyber-Physical Systems*, pages 188–197, 2010.
14. S. Ren, Y. He, and F. Xu. Provably-efficient job scheduling for energy and fairness in geographically distributed data centers. In *Proceedings of the 32nd International Conference on Distributed Computing Systems*, pages 22–31, 2012.
15. T. Sakamoto, H. Yamada, H. Horie, and K. Kono. Energy-price-driven request dispatching for cloud data centers. In *Proceedings of the 5th International Conference on Cloud Computing*, pages 974–976, 2012.
16. A. N. Sankaranarayanan, S. Sharangi, and A. Fedorova. Global cost diversity aware dispatch algorithm for heterogeneous data centers. In *Proceedings of the 2nd Joint WOSP/SIPEW International Conference on Performance Engineering*, pages 289–294, 2011.
17. Q. Tang, S. K. S. Gupta, and G. Varsamopoulos. Energy-efficient thermal-aware task scheduling for homogeneous high-performance computing data centers: A cyber-physical approach. *IEEE Trans. Parallel Distrib. Syst.*, 19(11):1458–1472, 2008.
18. Y. Wang, X. Wang, and Y. Zhang. Leveraging thermal storage to cut the electricity bill for datacenter cooling. In *Proceedings of the 4th Workshop on Power-Aware Computing and Systems*, pages 8:1–8:5, 2011.
19. Y. Yao, L. Huang, A. Sharma, L. Golubchik, and M. Neely. Data centers power reduction: A two time scale approach for delay tolerant workloads. In *Proceedings of the 31st Annual IEEE International Conference on Computer Communications*, pages 1431–1439, 2012.
20. Y. Zhang, Y. Wang, and X. Wang. Electricity bill capping for cloud-scale data centers that impact the power markets. In *Proceedings of the 41st International Conference on Parallel Processing*, pages 440–449, 2012.