Energy Cost-Effectiveness of Cloud Service Datacenters

Cheng-Jen Tang and Miau-Ru Dai

Graduate Institute of Communication Engineering,
Tatung University,
No. 40 Chung-shan N. Rd. Sec. 3, 104 Taipei, Taiwan
ctang@ttu.edu.tw, d9610002@ms2.ttu.edu.tw

Abstract. Cloud computing is a computation intensive service that clusters distributed computers providing applications as services and on-demand resources over Internet. Theoretically, such consolidated resource enhances the energy efficiency of both clients and servers. In reality, cloud computing is a panacea for enhancing energy efficiency under some certain conditions. For a user of cloud services, the computing resources are located at remote machines. Pioneers in exploring cloud computing, such as Google, AmazonWeb, Microsoft Azure, Yahoo, and IBM all use web pages as service interface via HTTP protocol. Through appropriated designs, sorting, one of the most frequently used algorithms, required by a web page can be executed and succeed by either clients or servers. As the model proposed in this paper, such client-server balanced computing allocation suggests a more energy-efficient and cost-effective web service.

Keywords: Energy Efficiency, Cloud Computing, Datacenter.

1 Introduction

Recently, datacenters have a significant growth in power density. Rapidly developing information and communication technologies (ICTs) and Internet elaborates establishment of datacenters. With the emergence of cloud computing, datacenters now also have to respond to the mounting computation requests. Modern datacenters usually equip with fully populated rack of blades to better use their limited spaces. Datacenters therefore become areas with the highest power density. The increasing power demand brings the rising energy costs together, thus improving energy efficiency becomes a key issue for the datacenter management. According to estimation from Amazons [10], expenses related to the cost and operation of their servers is responsible for 53% of the total budget, and energy-related costs for 42%, which consists of direct power consumption, about 19%, and the cooling infrastructure, 23%.

The energy demand of datacenters keeps growing due the emergence of cloud computing. Cloud computing is the newest Internet technology that evolves datacenters from information warehouses to information factories. The computation

R.-S. Chang, T.-h. Kim, and S.-L. Peng (Eds.): SUComS 2011, CCIS 223, pp. 94–103, 2011.

oriented nature of cloud computing differs the power consumption patterns of datacenters from the traditional storage oriented model. This computation oriented model is likely to widen the difference between the amount of energy required for peak periods and off-peak periods in a cloud computing datacenter due to:

1. CPU: Intensive computation enlarges CPU utilization rate. Power consumption of modern server CPUs is proportional to their utilization.
2. DRAM: Increased computation needs bring up the required memory sizes. DRAM has been identified as one of the main contributors to energy consumption in a computer [13].
3. Hard Disk: Frequent computation requests raise the frequency of random disk accesses. Random disk accesses consume much more energy than sequential disk accesses [16].

That is, cloud computing broaden the gap between the peak energy demand and off-peak energy demand because of increasing needs for computation.

The way of how computation performed greatly affects the power usage of datacenters. There are numerous algorithms that are applied in computer applications. A reasonable first step to establish the relationship between energy efficiency and computation algorithms is to find the most energy-influential algorithm as a starting point. Among all the frequently invoked algorithms in a computer, sorting is one of the most applied computing procedures. According to Knuth [11], one quarter of the entire 1960s computer running time was consumed by the computation of sorting processes. Sorting also has been considered as the foundation of many other algorithms [15]. In other words, computers spend a great portion of CPU cycles on sorting data. Therefore, this paper uses sorting to demonstrate how locating computing processes on clients or servers affects energy efficiency from the perspectives of energy demand side and energy supply side.

Sorting has to be performed by computer CPUs. Therefore, the energy consumption pattern of a CPU is critical. With the development of dynamic voltage and frequency scaling (DVFS), CPUs now consume power in proportion to their utilization. Studies [5] show that current desktop and server processors consume less power at low-activity modes than at busy modes. The average dynamic range, which is the difference between peak power and idle power of a CPU, of these CPUs is more than 70 percent of the peak power. For example, if a CPU has a 150-Watt peak power and a 30-Watt idle power, the dynamic range of this CPU is 120 Watts. Within the dynamic range of a CPU, its power consumption is close to proportional to its utilization rate.

People rarely work on standalone computers nowadays. Many computation works require the involvement of Internet services. Sustainability and availability becomes the major factors for choosing Internet service providers. The number and the frequency of the web requests are important factors to the energy consumption of a web server. A report published by Forrester Research [1] predicted that there would be over one billion PCs in use worldwide by the end of 2008,

and over 2 billion PCs in use by 2015. The projected number given by Forrester is probably a little underestimated. From another source [2], there were already 1,966,514,816 Internet users as of June 30, 2010. Based on the later number, if each Internet user spends one joule per second for a minute on computation tasks daily, the total consumed energy for one year is about 12 TWh. This amount of energy roughly equals to the energy generated by a nuclear reactor of Palo Verde, Arizona, which is the largest nuclear plant in U.S. [3] Obviously, how the computation tasks are handled is one of the important issues regarding the energy efficiency of datacenters. To effectively evaluate the energy efficiency of computation tasks need to consider the energy usage conditions on both client-side and server-side.

As mentioned above, cloud computing broaden the gap between the peak energy demand and off-peak energy demand because of increasing needs for computation. This paper discusses how computation task allocation, on clients or on servers, affects energy efficiency of datacenter from the perspectives of demand side and supply side. The findings help datacenter managers to improve the energy efficiency of computation-intensive web applications of cloud services.

2 Related Work

This section briefs some previous studies that address the energy efficiency issues, sorting performance issues, or related issues.

Besides designing a new energy-efficient hardware, existing researches mainly attack the energy efficiency issue regarding sorting algorithms from two different perspectives:

- Finds the most energy-efficient sorting algorithm by comparing different algorithms, such as Bunse et al.[8]
- Makes compilers to generate energy-efficient codes or use a energy-efficient library, such as Zhong et al.[17], Ayala et al.[4], and Segmund et al.[14]

Bunse et al. [8] define a set of trend functions that decides on which sorting algorithm to use under certain conditions. In their work, bubblesort, heapsort, insertionsort, mergesort, quicksort, selectionsort, shakersort, and shellsort are evaluated. Insertionsort is identified as the most energy-efficient sorting algorithm in this work, if the number of input items is large enough.

Zhong et al. [17] design a tool, AcovSA, Analysis of Compiler Options via Simulated Annealing that finds a good set of compiler options for a particular CPU and software. Although this tool is not particularly designed for optimizing the energy usage of a program, it is helpful for finding a set of compiler options that produces an executable image consuming less energy than other sets of options. Ayala et al. [4] tune the settings of register file with some code profiles. The main challenge of adopting such mechanism is the necessity of modifying ISA (Instruction Set Architecture). Segmund et al. [14] propose an energy feature library that is developed with many energy-saving techniques. These shared object libraries replace applications code with the code of the library or (de)activate the necessary hardware components.

Some existing approaches aim at building energy-efficient datacenters, such as Berl et al.[6], Rusu et al.[12], Bianchini, and Rajamony [7], and Elnozahy et al.[9]

Berl et al. [6] believe that Cloud Computing with Virtualization is a way to improve the energy efficiency of a datacenter. Rusu et al. [12] dynamically reconfigures a heterogeneous cluster to reduce energy consumption during off-peak hours. Bianchini, and Rajamony [7] identify the techniques for conserving energy in heterogeneous server clusters. Elnozahy et al. [9] show that using the dynamic voltage scaling (DVS) on each server node can achieve 29 percent energy saving. Moreover, by turning off certain nodes based on workload achieves 40 percent energy saving.

3 Energy Efficiency of a Datacenter

This paper uses the number of performed tasks, the power factor, and the cost of power generation to define the energy efficiency of a datacenter. The followings list the mathematical model of related factors.

Power factor is a ratio of the amount of real power to the apparent power. Real power is the capacity of datacenter machines for performing tasks at a particular time. Apparent power equals to the product of measured RMS (root mean square) voltage and RMS current of a datacenter, which is the supplied power. Suppose the amount of supplied power is P_s (the apparent power); the real power is P_u, which is the power consumed by working machines in a datacenter. The power factor PF of a datacenter is:

$$PF = \frac{P_u}{P_s} \tag{1}$$

- Suppose $PF_d(t)$ is the power factor of a datacenter d at time t.
- $P_{ds}(t)$ denotes the supplied power to a datacenter d at time t.
- n_d is the number of server machines in a datacenter d.
- K_d is a constant that represents the line loss and power consumed by other active electrically-driven devices in a datacenter d.
- $P_{di}(t)$ is the power consumed by a server machine i in a datacenter d, where $i = 1...n_d$.

$PF_d(t)$ is defined as:

$$PF_d(t) = \frac{K_d + \sum_{i=1}^{n_d} P_{di}(t)}{P_{ds}(t)} \tag{2}$$

The amount of supplied power must be larger than or equal to the demand. Therefore, power suppliers have to generate power based on some load forecasting. Due to the fact that electricity demand is not constant, different types of power generators are required to meet this fluctuating demand. Generators are usually divided into three different types according to their missions:

- Base load generators,
- Intermediate load generators, and
- Peak load generators.

Peak load generators usually cost most, followed by intermediate load generators, and then base load generators.

- $P_{Bd}(t)$ denotes the power generated for a datacenter d by base load generators at time t.
- $P_{Id}(t)$ denotes the power generated for a datacenter d by intermediate load generators at time t.
- $P_{Pd}(t)$ denotes the power generated for a datacenter d by peak load generators at time t.
- C_B denotes the unit cost of the power generated by base load generators.
- C_I denotes the unit cost of the power generated by intermediate load generators.
- C_P denotes the unit cost of the power generated by peak load generators.

The power generation cost for the datacenter d at time t is:

$$C_d(t) = C_B P_{Bd}(t) + C_I P_{Id}(t) + C_P P_{Pd}(t) \tag{3}$$

- Suppose the energy required for finishing a client request k on a server i of a datacenter d is E_{kdi}.
- Suppose a server i needs a period of time L_{kdi} to finish the request k.

The power requirement of the request k performing on a server i of a datacenter d is:

$$P_{kdi} = \frac{E_{kdi}}{L_{kdi}} \tag{4}$$

- Suppose the power usage of a server i at the idle state is $P_{idle_{di}}$.
- Suppose the server i performs one request at a time. Other requests are queued until the working request has finished.

Therefore,

$$\begin{cases} P_{di}(t) = P_{idle_{di}}, \textit{if there is no job at time } t \\ P_{di}(t) = P_{idle_{di}} + P_{kdi}, \textit{if there is a request } k \textit{ at time } t \end{cases} \tag{5}$$

Energy Efficiency of a Datacenter

- Suppose the number of finished jobs in a datacenter d for a period t_0 to t_1 is J_d.
- Suppose the investigated energy suppliers have only one customer that is the datacenter d.

Therefore, the energy efficiency of the datacenter d for a period t_0 to t_1 is defined as:

$$Eff_d = \frac{J_d}{\int_{t_0}^{t_1} C_d(t)\mathrm{d}t} \tag{6}$$

Investigation on Energy Efficiency. Suppose C_B, C_I, and C_P are constants, and $C_P > C_I > C_B$. From Eq. (3) , to determine $C_d(t)$ needs to find $P_{Bd}(t)$, $P_{Id}(t)$, and $P_{Pd}(t)$.

- Suppose P_{MAX_B} is the maximum output of base load generators.
- Suppose P_{MAX_I} is the maximum output of intermediate load generators.
- Suppose P_{MAX_P} is the maximum output of peak load generators.

$$\begin{cases} P_{ds}(t) = P_{Bd},\ for\, P_{ds}(t) \leq P_{MAX_B} \\ P_{ds}(t) = P_{MAX_B} + P_{Id}(t), \\ \ for\, P_{ds}(t) \leq P_{MAX_I} + P_{MAX_B} \\ P_{ds}(t) = P_{MAX_B} + P_{MAX_I} + P_{Pd}(t), \\ \ for\, P_{ds}(t) \leq P_{MAX_P} + P_{MAX_I} + P_{MAX_B} \end{cases} \tag{7}$$

This paper does not consider the condition of $P_{ds}(t)$ exceeding P_{MAX_P}, which causes the circuit breaker of the datacenter to be tripped.

In order to simplify the calculations, this paper assumes the datacenter d uses homogeneous architecture, which means all server machines have the same hardware configuration. Furthermore, this paper also assumes all requests consume the same amount of energy, and P_j is the power requirement for performing a request. Therefore, for a period t_0 to t_1

$$\sum_{i=0}^{n_d} P_{di}(t) = n \times P_{idle_{di}}(t) + \frac{J}{t_1 - t_0} \times P_j \tag{8}$$

Suppose the datacenter d has a constant power factor PF_d. By substituting Eq. (8) in Eq. (2), $P_{ds}(t)$ is written as:

$$P_{ds}(t) = \frac{K_d + n \times P_{idle_{di}}(t) + \frac{J}{t_1 - t_0} \times P_j}{PF_d} \tag{9}$$

From Eq. (9), the P_{MAX_B} of a carefully designed power generation system of a data center d is:

$$P_{MAX_B} = \frac{C + n \times P_{idle_{di}}}{PF_d} \tag{10}$$

The unit cost of the power generated by peak load generators C_P is always the highest among C_B, C_I, and C_P. Peak load generators are operated under some critical conditions. For most of time, the $P_{ds}(t)$ is:

$$P_{ds}(t) = P_{MAX_B} + P_{Id}(t) \tag{11}$$

The $C_d(t)$ is therefore:

$$C_d(t) = C_B P_{MAX_B} + C_I P_{Id}(t) \tag{12}$$

From Eq. (9), Eq. (10), and Eq. (11), the following equation is obtained:

$$P_{Id}(t) = \frac{\frac{J}{t_1 - t_0} \times P_j}{PF_d} \tag{13}$$

By substituting Eq. (13) in Eq. (12), $C_d(t)$ is written as:

$$C_d(t) = C_B P_{MAX_B} + C_I \frac{\frac{J}{t_1 - t_0} \times P_j}{PF_d} \tag{14}$$

A well-known mathematical model for modeling request arrival is the queuing model. The probability of that there are exactly k arrivals of jobs is equal to:

$$f(k, \lambda) = \frac{\lambda^k e^{-\lambda}}{k!} \tag{15}$$

where k is the number of occurrences of an event, λ is a positive real number that equals to the expected number of arrival jobs during the given interval, which is $\frac{J}{t_1 - t_0}$. However, the number of arrival requests is not controllable in real world. From Eq. (6) and Eq. (14), the following equation represents the energy efficiency under normal condition, which is neither critically peaking nor completely idle.

$$Eff_d = \frac{J}{\int_{t_0}^{t_1} C_B P_{MAX_B} + C_I \frac{\frac{J}{t_1 - t_0} \times P_j}{PF_d} \mathrm{d}t} \tag{16}$$

$$Eff_d = \frac{J}{(C_B P_{MAX_B} + C_I \frac{\frac{J}{t_1 - t_0} \times P_j}{PF_d})(t_1 - t_0)} \tag{17}$$

Let a constant $C_0 = C_B P_{MAX_B}(t_1 - t_0)$, a constant$C_1 = \frac{C_I P_j}{PF_d}$, the Eq. (17) is written as:

$$Eff_d = \frac{J}{C_0 + C_1 J} \tag{18}$$

$$Eff_d = \frac{1}{C_1} - \frac{C_0}{C_1(C_0 + C_1 J)} \tag{19}$$

From Eq. (19), to maximize Eff_d needs to maximize J. The upper bound of J in a period t_0 to t_1 is $\frac{t_1 - t_0}{L_j}$. Therefore, an easy way to gain a better energy efficiency of a data center is to reduce the L_j, which is the period of time required for finishing a request.

4 Experiment Result and Analysis

The mentioned work in section 2 for enhancing energy efficiency of datacenters have their strength on certain aspects. However, to implement any of them either requires taking some major changes on systems, or needs to recompile existing code. Such modification might not be acceptable for some system operators; since system-level and code-level modifications often accompany with uncertainties, which lead to unexpected system failures. To avoid such risk, this paper proposes a new approach that is to make the required computation tasks of web pages swappable. In other words, this is to identify computation tasks of interactive

web pages that can be performed by either the server or the client. If a server is more energy-efficient for performing the task than its client, let the server does the job. Otherwise, let its client does the job. This paper takes the sorting as a starting point to examine this concept. This section identifies some possible sorting-performing scenarios.

In considering the energy consumption scenarios of client computers, the power-saving CPUs and the ordinary CPUs are fundamentally different. Hence, there are two scenario groups: one is for the power-saving CPUs, and the other is for the ordinary CPUs.

Most of the content-sortable web pages perform sorting on servers. To make the sorting performed on a client, a sorting function is needed to be shipped with the web page. This sorting function is often written in Javascript or other client-side scripting languages. However, performing the sorting using an external application often offers a better performance than the browser-embedded scripting engine.

The workload of a server greatly affects the performance of a web server. In the surveyed scenarios, three workload levels are considered:

- light: 1 to 10 connections
- medium: 10 to 100 connections
- heavy: 100 to 1000 connections

For the power-saving CPUs, this experiment setup chooses Intel Atom processor N280 with 512KB Cache, 1.66 GHz, 667 MHz FSB. Intel uses clock gating to manage the power usage of Atom. This mechanism activates the clocks in a logic block only when they have work to do. Intel Atom N280 consumes 2.5 watts at the peak utilization, 100 mwatts at idle state. Acer Veriton N260G, which equips with Intel Atom N280, is used in the experiment. In the experiment, up to 4 Acer Veriton N260G PCs are used. The power supply unit (PSU) of Acer Veriton N260G provides up to 65 Watts. For the ordinary CPUs, Intel Pentium 4 2.4 GHz Northwood Processor with 533 MHz FSB, 512KB cache is used. 4 HP Compaq D330 Minitower PCs that equipped with Intel Pentium 4 2.4 GHz are chosen in this experiment. D330 uses a 240-Watt PSU. Intel Pentium 4 2.4 GHz Northwood has a 59.8-watt thermal design power (TDP).

The server used in this experiment equips with 4 Intel Xeon 2.4 GHz processors. Each processor in the server has a 65-watt TDP. Keep one thing in mind, this Intel Xeon 2.4 GHz processor is considered ancient, since it was first announced in March, 2002. The server has a CPU set of 260-watt TDP, and is equipped with two 500-Watt PSUs. This experiment assumes the CPU TDP is the power of a CPU at peak utilization, and PSU capacity is its maximum power.

In order to examine the finding of Eq. 19, there are three tests with f=8.33, 4.16, and 1.67 requests/sec in this experiment. Every run of each test is observed for 60 seconds. Each test uses two different web pages, sorted-by-client, and sortedVby-server. Each test runs either page 10 times. Table 1 shows the experiment result. From the result, there is an observable difference. This experiment is conducted on a single machine server. The process time has been reduced

Table 1. Experiment Result

	Sorted-by-client		Sorted-by-server		
	Ave. time(s)	Process time(s)	Ave. time(s)	Process time(s)	Improvement
$\lambda = 8.33$	0.026	13	0.029	14.5	10.34%
$\lambda = 4.16$	0.026	6.5	0.028	7	7.14%
$\lambda = 1.67$	0.02	2	0.025	2.5	20%

to some extent. For a datacenter that is with multiple machines and load balancing mechanism, the result implies a lessening of the difference between peak and off-peak demands.

5 Conclusion

Cloud computing regards applications (or software) as a service. A cloud computing datacenter must conduct huge amount of computations in addition to traditional server tasks. The energy cost of cloud computing datacenters are therefore staggering high. Recently, researchers try to resolve the energy-hunger problem for environmental purpose, or economical purpose etc. Most of the existing proposals focus on reducing the use of energy. This paper presents a different approach that levels the peak and off-peak demands to improve energy efficiency. In this paper, the definition of energy efficiency involves different types of power generation cost, and computation time. An experiment result shows this approach improves the utilization of the generated power. The proposed mathematic model suggests that consolidating computing resources altogether into a datacenter is not always the most efficient approach.

Acknowledgements. This study is funded by the National Science Council of the Republic of China under grant NSC100-2623-E-036-004-ET.

References

1. Forrester research - marketing and strategy data (2008), `http://www.forrester.com/consumerdata/overview`
2. Internet world stats - world internet users and population stats (2010), `http://internetworldstats.com/stats.htm`
3. Nuclear energy institute - u.s. nuclear power plants (2011), `http://www.nei.org/resourcesandstats/nuclear_statistics`
4. Ayala, J.L., Veidenbaum, A., Lpez-Vallejo, M.: Power-aware compilation for register file energy reduction. International Journal of Parallel Programming 31, 451–467 (2003), `http://dx.doi.org/10.1023/B:IJPP.0000004510.66751.2e`, doi:10.1023/B:IJPP.0000004510.66751.2e

5. Barroso, L.A., Hölzle, U.: The case for energy-proportional computing. IEEE Computer 40(12), 33–37 (2007), http://doi.ieeecomputersociety.org/10.1109/MC.2007.443
6. Berl, A., Gelenbe, E., Di Girolamo, M., Giuliani, G., De Meer, H., Dang, M., Pentikousis, K.: Energy-efficient cloud computing. The Computer Journal 53(7), 1045 (2010)
7. Bianchini, R., Rajamony, R.: Power and energy management for server systems. Computer 37(11), 68–76 (2004)
8. Bunse, C., Höpfner, H., Roychoudhury, S., Mansour, E.: Choosing the best sorting algorithm for optimal energy consumption? In: Proceedings of the International Conference on Software and Data Technologies (ICSOFT), pp. 199–206 (2009)
9. Elnozahy, E., Kistler, M., Rajamony, R.: Energy-efficient server clusters. Power-Aware Computer Systems, 179–197 (2003)
10. Hamilton, J.: Cooperative expendable micro-slice servers (CEMS): low cost, low power servers for internet-scale services. In: Conference on Innovative Data Systems Research (CIDR 2009), Citeseer (January 2009)
11. Knuth, D.E.: The Art of Computer Programming, Sorting and Searching, 2nd edn., vol. 3. Addison-Wesley, Reading (1998)
12. Rusu, C., Ferreira, A., Scordino, C., Watson, A.: Energy-efficient real-time heterogeneous server clusters. In: Proceedings of the 12th IEEE Real-Time and Embedded Technology and Applications Symposium, pp. 418–428. IEEE, Los Alamitos (2006)
13. Schmidt, D., Wehn, N.: Dram power management and energy consumption: a critical assessment. In: Proceedings of the 22nd Annual Symposium on Integrated Circuits and System Design: Chip on the Dunes, SBCCI 2009, pp. 32:1–32:5. ACM, New York (2009), http://doi.acm.org/10.1145/1601896.1601937
14. Siegmund, N., Rosenmüller, M., Apel, S.: Automating energy optimization with features. In: Proceedings of the 2nd International Workshop on Feature-Oriented Software Development, FOSD 2010, pp. 2–9. ACM, New York (2010), http://doi.acm.org/10.1145/1868688.1868690
15. Skiena, S.S.: The Algorithm Design Manual, 2nd edn. Springer, Heidelberg (2008)
16. Zedlewski, J., Sobti, S., Garg, N., Zheng, F., Krishnamurthy, A., Wang, R.: Modeling hard-disk power consumption. In: Proceedings of the 2nd USENIX Conference on File and Storage Technologies, pp. 217–230. USENIX Association (2003)
17. Zhong, S., Shen, Y., Hao, F.: Tuning compiler optimization options via simulated annealing. In: Second International Conference on Future Information Technology and Management Engineering, FITME 2009, pp. 305–308 (2009)