2012 IEEE Fifth International Conference on Cloud Computing

Analysis of the Power and Hardware Resource Consumption of Servers under Different Load Balancing Policies

Waltenegus Dargie and Alexander Schill
Chair for Computer Networks
Faculty of Computer Science
Technical University of Dresden
01062 Dresden, Germany
Email: {waltenegus.dargie, alexander.schill}@tu-dresden.de

***Abstract*—Most Internet applications employ some kind of load balancing policies in a cluster setting to achieve reliable service provision as well as to deal with a resource bottleneck. However, these policies may not ensure the utilization of *all* of the hardware resources in a server equally efficiently. This paper experimentally investigates the relationship between the power consumption and resource utilization of a multimedia server cluster when different load balancing policies are used to distribute a workload. Our observations are the following: (1) A bottleneck on a single hardware resource can lead to a significant amount of underutilization of the entire system. (2) A ten times increment in the network bandwidth of the entire cluster can double the throughput of individual servers. The associated increment in power consumption of the individual servers is 1.2% only. (3) For TCP-based applications, session information is more useful than other types of status information to utilize power more efficiently. (4) The use of dynamic frequency scaling does not affect the overall throughput of IO-bound applications but reduces the power consumption of the servers; but this reduction is only 12% of the overall power consumption. More power can be saved by avoiding a resource bottleneck or through service consolidation.**

***Index Terms*—Cluster computing, load balancing, power consumption, resource utilization, service consolidation**

I. Introduction

The problem of energy consumption in large-scale servers and data centers has become a prime concern due to a number of reasons. Firstly, the cost of energy is almost universally and steadily increasing and may affect the way Internet based services and contents are provided. Secondly, the rate at which data (contents) are generated and shared over the Internet requires a corresponding adjustment in the ICT energy budget. Thirdly, while preliminary research indicates that the ICT sector will play a positive role in reducing the carbon footprints of many sectors (for example, through smart motor systems, smart logistics, smart grid, smart buildings, etc.), its own carbon footprint has been showing a significant increment over the past few years [1].

As far as Internet application servers are concerned, the problem of energy consumption has been addressed in the following ways:

- Optimal placement of servers and data centers [2], [4], [10]. The aim is mainly to minimize the cost of cooling as well as to utilize renewable (clean) energy.
- Optimal placement of data (content) [11]. This refers to putting (multimedia) data near to their final consumers.
- Consolidation of services through live migration [4], [6] and dynamic resource pool resizing [8]. The aim is to consolidate services in an optimal number of servers in a cluster and selectively switch off underutilized resources.
- Employing frequency and voltage scaling [5]. This enables the operating system to optimally allocate hardware resources (CPU) for impending jobs and to avoid idle-state power consumption.

The existing approaches dealing with runtime consolidation of services employ some form of load balancing strategy, which has essentially three assignments: (1) it regularly estimates the workload arrival rate at the cluster of servers and estimates their resource cost; (2) it applies an optimal distribution strategy and assigns individual servers to process a portion of the workload (this step entails service migration as well as service rebinding); and (3) it monitors the resource consumption of the cluster and, if need be, migrates services as well as workloads from server to server to minimize the number of underutilized or overloaded resources. These steps are not trivial and introduce their own cost, both in terms of energy consumption and resource utilization.

One way of reducing this cost is to ensure that the load balancer achieves an optimal workload distribution in the first place (i.e., in step 1). For example, Chen et al. [5] employ a centralized approach to deliberately skew (instead of balance) workload distribution in a TCP-based, client-server architecture. By so doing, the "load balancer" creates "tail" servers with fewer live connections and then it gradually switches off underutilized servers. The load balancer exploits knowledge of the computational cost of establishing TCP connections.

Likewise, Elnozahy et al. [7] propose five different types of power management policies that can be applied by a load balancer. One of these, independent voltage scaling, can be

978-0-7695-4755-8/12 $26.00 © 2012 IEEE
DOI 10.1109/CLOUD.2012.30

employed in peer-to-peer setting while all the others are suitable for a centralized setting. The centralized policies enable a "load balancer" to estimate the optimal CPU clock frequency budget of the entire cluster and selectively switch on or off servers. The optimal number of servers in a cluster is computed by defining an upper and a lower limit to the cluster's clock frequency budget. The future operation clock frequency of the cluster depends on the anticipated workload. For example, if the load balancer determines that the anticipated workload requires an aggregate CPU frequency that is above the upper frequency budget, it decides to turn on additional servers, but if this frequency is below the lower limit, it decides to switch off some of the active servers. If the desired frequency is between the upper and the lower limit, a coordinated voltage scaling policy is applied. Similar energy-aware service (request) consolidation approaches are given in [13] and [4].

Most of the energy-aware load balancing strategies focus on the CPU, as it is the one which consumes a significant portion of the overall power consumption of a server. But this is true only when the workload of the CPU varies according to the workload of the server. If, on the other hand, the workload of the server does not create a corresponding load on the CPU because of (1) an intermediate resource bottleneck or (2) the nature of the workload (for example, if it is not a CPU-bound workload), then putting the main focus on the CPU alone may not result in saving a significant energy.

This paper experimentally investigates how hardware resources are utilized and power is consumed in a cluster environment when servers are managed by a load balancer. The load balancer employs different load balancing policies to determine how the workload should be distributed. Our aim is to show that service consolidation may not function well unless the services require complementary resources. Secondly, the experiment is useful to identify the parameters which enable a load balancer to significantly reduce the energy-consumption of a cluster without affecting its performance. We will use a multimedia server cluster for our experiment and the throughput of the servers as a performance metric.

The rest of this paper is organized as follows. In Section II, a brief summary of some of the existing and commercially employed load balancing policies are given. In Section III our experiment set up and the methodology of data analysis and interpretation is presented in detail. In Section IV, the evaluation of five load-balancing policies and the results of the experiment are discussed. Finally, in Section V some of the open research issues in this area are outlined and concluding remarks are given.

II. Load Balancing

A large number of Internet applications employ some kind of load balancing mechanisms. In a centralized load dispatcher architecture, a centralized load balancer decides the way a workload is distributed, whereas in a peer-to-peer environment, individual nodes decide how much load they can accommodate and share this knowledge with their peers. In a cluster environment, the former architecture is often employed, serving as a gateway to users and distributing workload among the servers.

One of the factors that necessitate the use of load balancing is that a specific workload may saturate some of the critical resources (for example, the communication bandwidth) of a server and distributing the workload between two or more servers becomes necessary. But this also means that the same workload may not saturate all the resources at the same time, and hence, the use of a load balancer alone may not ensure the utilization of all of the resources equally efficiently. This is particularly true if the servers host a single application that serves a large number of Internet users.

There are a large number of load balancing policies, but we will consider only those which are frequently used. These policies can be classified as none-state-based and state-based policies. The none-state-based policies do not require update information (pertaining to their current resource utilization) from individual servers whereas the state-based policies require periodic updates regarding resource utilization. In the first category, we have the round robin and random algorithm (policy) [3].

The simplest of these is the round robin algorithm in which incoming requests are distributed to all the active servers in a round robin fashion, as the name implies. The algorithm does not require knowledge of how resources are distributed in the active servers and, therefore, it is fast. But this also means that it does not prioritize tasks or direct them to preferred servers. The weighted round-robin algorithm assigns weights to the active servers according to their computational capability and uses this knowledge to determine how user requests should be dispatched.

The random scheduling algorithm forwards incoming requests randomly, but the randomness is governed by an underlying probability distribution function that takes knowledge of the computational capacity of the active servers into account. Where this knowledge is not available, the uniform distribution is assumed.

The second category of policies can take several types of context information into account in order to distribute workload [14]. For example, the least session algorithm forwards the next request to the server that has the least number of active sessions or connections; the shortest expected delay algorithm forwards the next request to a server which will have the shortest expected delay – the expected delay is given as $\frac{C_{(i+1)}}{U_i}$, where C is the active number of connections and U is the fixed service rate of the i_{th} server; the never queue algorithm sends the next request to any idle server, but if there is no idle server, then it employs the shortest-expected-delay approach to select a server.

Likewise, the least utilized resource algorithm forwards the next request to a server which is utilizing the least resource, for example, least CPU, memory, disk space, or network bandwidth.

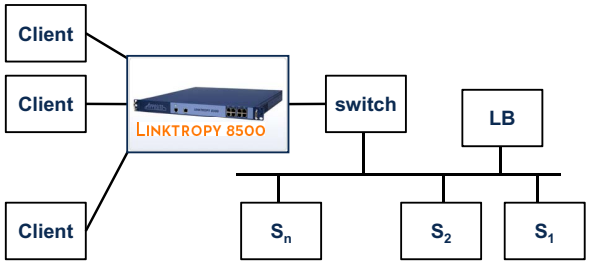

Fig. 1. The setup of the multimedia server cluster

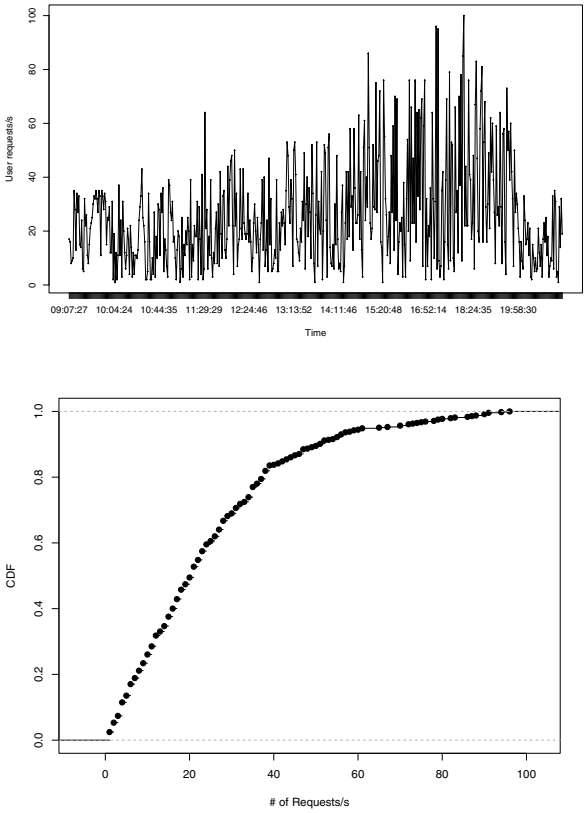

Fig. 2. A sample of the request arrival rate for a single server during a single day: Time series (top) and CDF (bottom).

III. Experiment

A. Cluster Set up

The server cluster we set up for our experiment consists of two multimedia servers and a load balancer connected to the multimedia servers via a switch (Figure 1). The servers run the Ubuntu Server Edition (v. 10.04)[1] on top of which the Apache Server[2] is installed to handle HTTP requests. The load balancer monitors the workload as well as the resource consumption of the servers. The clients and the server cluster are isolated from each other by a network emulator (Linktropy 8500) which introduces various constraints (delay, bandwidth, congestion, jitter, and packet loss) into the network. The servers employ AMD Athlon Dual Core 2 GHz processors, 4 Gbit DDR2 SDRAM memory, and 1 Gbit/s network interface card . They host more than 100 videos having different sizes. The length of the videos is between 3 MB and 100 MB and they have an exponential popularity distribution, the most popular videos having a mean size of 5 MB. The two servers host identical videos.

The client applications generate user requests for downloading the videos. The user request rate has a time varying Poisson distribution. The request rate starts slowly at 8 AM and gradually reaches peak between 6 and 8 PM. The pattern is similar to the one observed among IPTV application users [9]. Figure 2 displays a sample of the request rate observed by a single server on a single day.

We use a commercially available load balancer called BalanceNG[3] to carry out a realistic experiment.

B. Measurement

We employed Yokogawa WT210 digital power analyzers to measure and analyze the energy and power consumptions of the servers. The devices can measure DC as well as AC power consumption at a rate of 10 Hz and a DC current between 15 μA and 26 A with an accuracy of 0.1%.

The load balancer as well as the two servers were built on a D2641 Siemens/Fujitsu Motherboard architecture. The motherboard is supplied with power through two Molex connectors (one 4-pole and the other 25-pole). The 4-pole connector provides a 12 V while the 24-pole provides 3.3 V, 5 V, and 12 V. To understand the DC power consumption of the servers, it necessary to examine how the different operation voltages of the processor and the memory as well as the IO controllers are generated. In fact, the quality of a motherboard is mainly determined by this specific aspect.

The CPU core voltage is generated by a three-phase voltage regulator controlled by an ISL 6312 Pulse Width Modulator (PWM) controller. The PWM controller takes its core voltage from the 5 V line, but the main voltage of the voltage regulator comes from the 12 V line of the 4-pole connector (we denote this voltage by 12V2)). Hence, this line (12 V) is exclusively used by the processor. Likewise, the motherboard provides the memory unit with a single phase voltage regulator controlled by an ISL 6545 PWM controller. This voltage regulator draws much of its current from the 5V line. The motherboard also provides additional voltage regulators to the Southbridge and the various IO controllers. The Southbridge voltage regulators predominantly use the 12 V line of the 24-pole (denoted as 12V1) while all the other IO controllers predominantly draw current from the 3.3 V.

We used a PCI Express and a raiser board to directly measure the power consumption of the network interface card (shown in Figure 3).

[1]http://www.ubuntu.com
[2]http://www.apache.org/.
[3]http://www.inlab.de/balanceng/. Last accessed November 21, 2011: 4:00 PM CET.

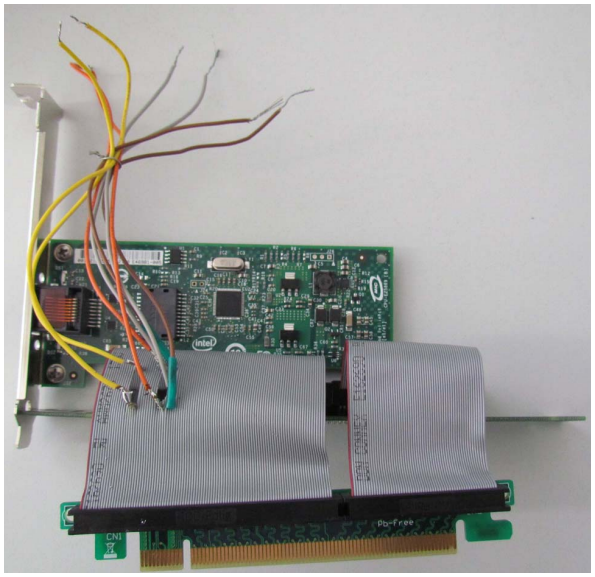

Fig. 3. Intercepting the power supply line of the network interface card using a PCI express and a Raiser board.

IV. Evaluation

The resource utilization and the power consumption of the servers under a specific load balancing policy are evaluated in three different ways, namely, in terms of:

1) The resources consumed,
2) The distribution of the resource consumed; and,
3) The throughput of the servers.

We run each algorithm twice, each time for 12 hours. We modeled the power consumption and resource utilization of the servers as random variables, P and C, respectively. Therefore, the cumulative distribution function of each random variable gives us the knowledge of the probability that it has a value below a certain real number *p or c*.

A. DC Power

Figure 4 shows the DC power consumption of the different subsystems (memory, processor, disk drive, NIC, and other IO controllers) of one of the servers when the Least Bandwidth policy was applied. The server's average AC power consumption was around 55 W. The corresponding DC power measured at the two Molex connectors was 36 W. this means the power supply unit has an efficiency of 65%[4]. This observation is consistent with the ATX specification[5].

As can be seen, the processor consumed much of the power. Moreover, this power consumption is dynamic, since it varied according to the workload of the server. A significant amount of the power consumed via the 5 V line is due to the memory subsystem and the memory termination circuit, since the power transistors of the voltage regulator of the memory subsystem are connected to this line. The power consumption has two aspects, a static and a dynamic aspect (see Figure 4, top-middle). The 5 W power consumption is the minimum power required to set the memory subsystem in operation. It does not vary as the amount of data stored in memory changes. The additional power consumption (approx. 3 W), was due to the transfer of data between memory and the CPU (VTT) and it is a dynamic cost.

All the other subsystems, including the network interface card, which has a consumption of 2 W, can be considered as a constant cost, as the current drawn from the 3.3 V, 12V1, 5 V (disk drive) and 12 V (disk drive) under different load conditions showed essentially a small variation[6].

B. Power consumption vs. Resource utilization

Figure 5 displays the CDFs of P and C of the two multimedia servers when the five load balancing policies were employed. The load balancing policy that resulted in the highest CPU utilization was the one based on the least CPU utilization, followed by the round robin policy. The third was the policy based on the least bandwidth. The load balancing policies that consumed the lowest amount of power with the highest probability were the random policy and the least session policy. The third was the round robin policy.

The load balancing policy that produced the highest throughput was the least session policy followed by the round robin policy. The third was the random policy. A summary of the comparison between the five load balancing policies is given in Table I.

The policies which resulted in almost identical patterns of resource consumption in both servers were the least CPU, the random, and the round robin policies. But even though it is reasonable to assume that much of the power consumption of a server is often due to the cost of processing, a similar pattern in CPU utilization by itself does not necessarily mean a corresponding similarity in power consumption. This is clearly the case when we consider the power consumption pattern of the two servers under the least CPU policy, which shows that the CDF of server one (blue) is not the same as that of the CDF of server two (red). On the contrary, the CDF of the power consumption of the least session policy shows a distinct difference in the way the CPUs are utilized under this policy, but the CDF of the power consumptions of the two servers were remarkably similar.

In all the evaluation criteria we defined, the least session policy produced the best result. This is because in TCP-based applications, the CPU's significant power consumption occurs during the creation of sessions (and then, the CPU remained idle much of the time during data transfer). This observation is in accord with the observation made by Chen

[4]This is without considering the power dissipation of the measurement devices, which is around 1.8 W.

[5]ATX Specification, version 2.2 (2003 – 2005), Intel Corporation.

[6]This is true for the power consumption of the peripheral controller as well as the IO devices (NIC and the graphic card). The power consumption of the disk drive can also be considered as a constant cost as long as the request rate is below 100/s and the average file size is small. For larger request rates and larger video data, the overall power consumption of the disk drive fluctuates between 7 and 14 W.

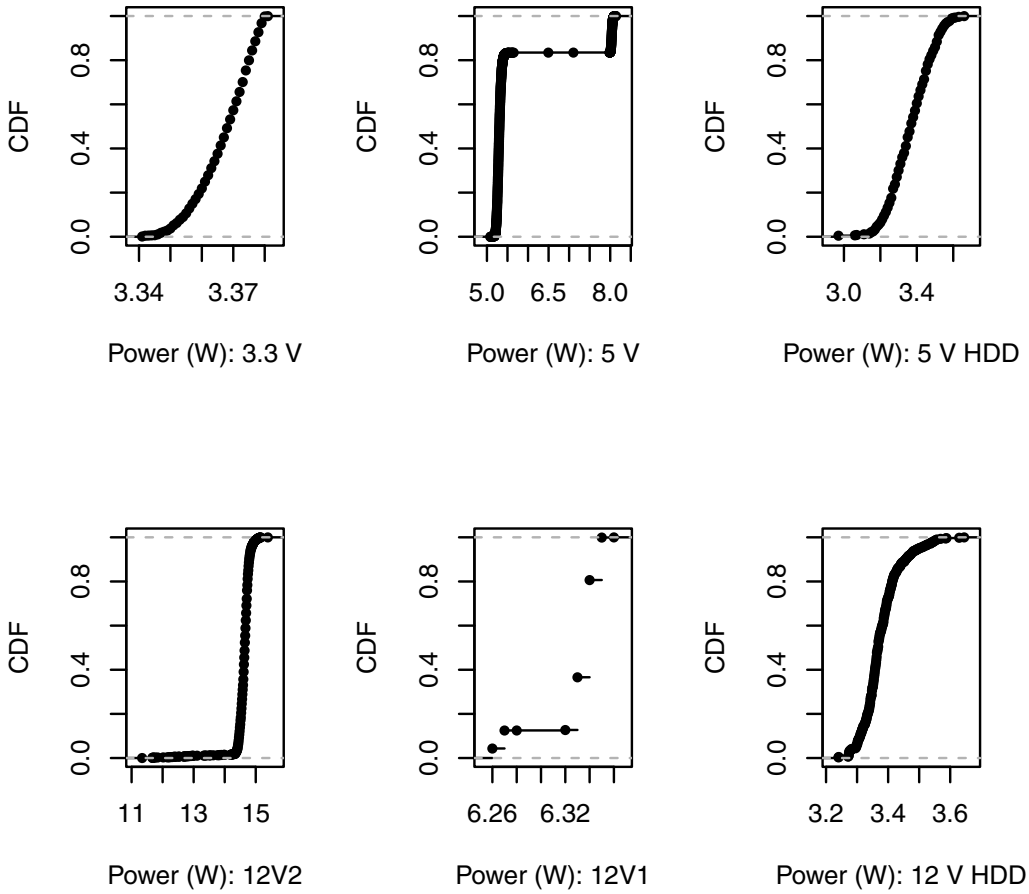

Fig. 4. Breakdown of the DC power consumption of one of the servers under the Least Bandwidth Policy. 12V1 refers to the 12 V line in the 24-pole connector whereas 12V2 refers to the processor voltage obtained from the 4-pole connector. HDD refers to the voltage of the disk drive.

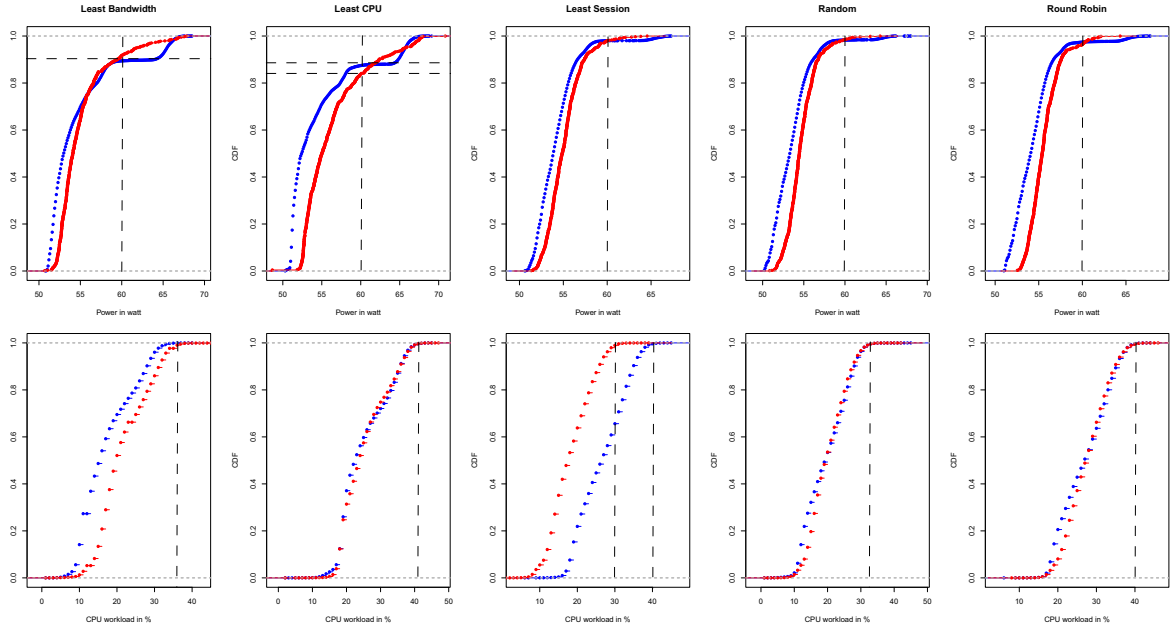

Fig. 5. The Power consumption (above) and Resource consumption (below) profiles of the five load balancing policies (Red: Server 1. Blue: Server 2.)

Policy	$(P \leq 60W)$	$(C \leq c) \approx 1.0$	Throughput (GB)
Least Bandwidth	0.9	36%	308.61
Least CPU	0.85 (S1) 0.87 (S2)	41%	330
Least Session	0.99	30% (S1) 40% (S2)	341.04
Random	0.99	32%	324.55
Round Robin	0.97	40%	337.63

TABLE I
COMPARISON BETWEEN DIFFERENT LOAD BALANCING POLICIES

et al. [5]. Moreover, unlike all the parameters, based on which the load balancing policies made decisions to forward a request, the session information was the one which did not stall quickly. Interestingly, the policies which did not require state information – namely, the random and the round robin policies – out perform the policies which needed state information pertaining to bandwidth and CPU utilization.

C. Resource Bottleneck

As can be seen in Figure 5, the dual-core processors of the multimedia servers were not fully utilized (the CPU utilization never exceeded 50%). On the other hand, the bandwidth utilization of the cluster was almost always near saturation. Increasing the network simulator's bandwidth by ten times (from 1 Gbit/s to 10 Gbit/s) increased the throughput of each servers almost by double. The corresponding increment in power consumption for each server was 1.25% only. The power efficiency ($\eta = (P_{active} - \overline{P}_{idle})/\overline{P}_{idle}$) achieved this way is displayed in Figure 6.

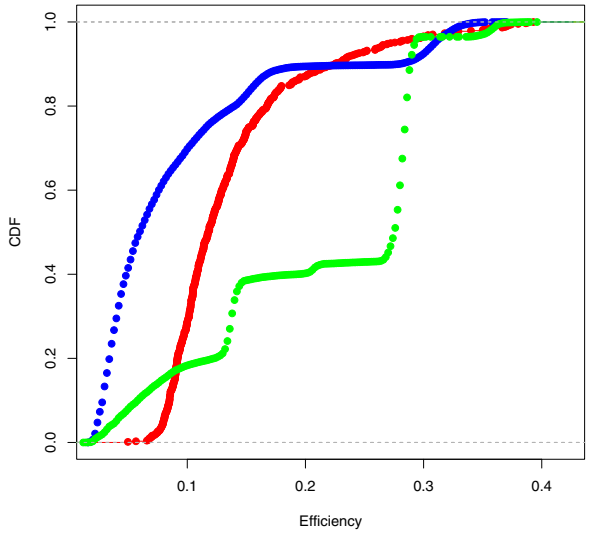

Fig. 6. A comparison of the power efficiency of the two servers when there was a resource bottle neck (blue and red) and when the bottleneck was removed (green).

D. Dynamic frequency scaling

Whereas removing the network constraint doubled the throughput of the cluster without a significant amount of energy consumption, the CPUs of both servers were still much of the time not fully utilized. Therefore, we applied dynamic frequency scaling to examine the extent to which we could reduce the power consumption of the servers without a significant reduction in the overall throughput.

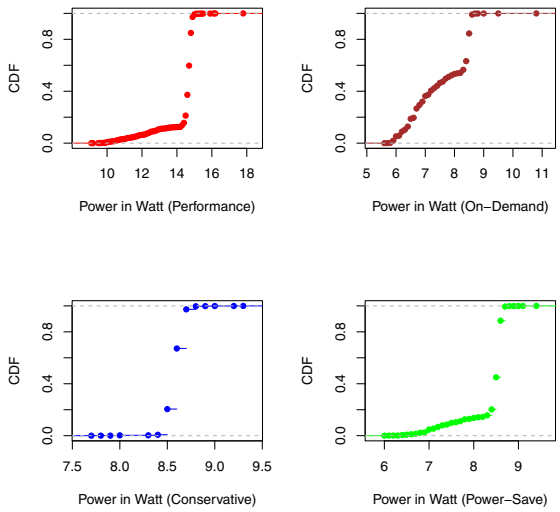

Fig. 7. CPU power consumption with different power management policies.

We integrated the **cpufrequtils** utilities[7] into the Ubuntu kernel infrastructure for supporting dynamic CPU frequency scaling. The utilities provide us with four different types of policies: Performance, On-Demand, Conservative, and Power-Save[8].

The Performance policy runs the CPU at the maximum clock speed. The On-Demand policy switches the CPU between the available frequencies based on the present workload. The Conservative policy is similar to the On-Demand policy, but it scales the clock frequency gradually (typically, in steps of 10 μs [12]). The Power-Save policy runs the CPU at the lowest clock speed regardless of the workload.

Figure 7 displays the power consumption of the CPU of one of the servers (measured from the 12V2 supply line (the 4-pin connector) under the four frequency scaling policies. Because the video application was an IO-bound application, we observed no noticeable reduction in the throughput. The power consumption of the server via the 3.3 V, 5 V, and 12V1 remained nearly unchanged, as expected, but the power consumption via the 12V2 shows different levels of changes. The optimal gain was observed with the "power save" mode (an average gain of 12%). Figure 8 displays the power consumption of the server via the 3.3 V when the different power management policies were employed.

[7]https://wiki.archlinux.org/index.php/Official_Repositories: Last accessed on November 14, 2011: 22:38 CET.

[8]The utilities also enable users to manually configure clock speed, but this was not interesting for our experiment.

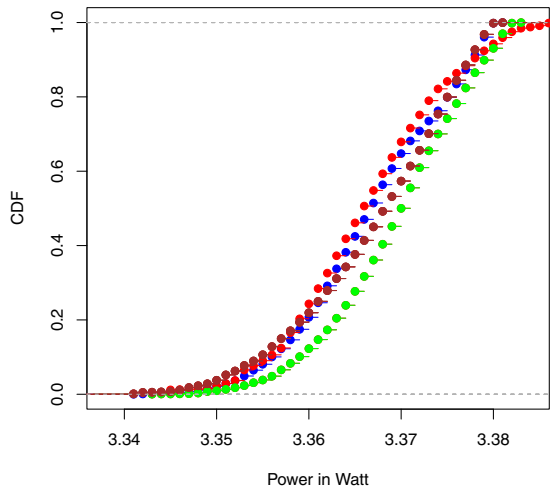

Fig. 8. The power consumption of one of the servers via the 3.3 V when different power management strategies were applied to the CPU.

Having said this, compared to the overall power consumption of the server, the 12% gain should not be considered significant. This implies that dynamic frequency and voltage scaling alone is not an optimal strategy in a cluster environment. The co-scheduling of complementary services (for example, a CPU-bound service) will certainly result in a better power efficiency.

V. Conclusion

In this paper we investigated the relationship between power consumption, resource utilization, and throughput under different load balancing policies. We considered five load balancing policies, namely, least CPU utilization, least bandwidth utilization, least current Sessions, random, and round robin. These policies can be classified into state-based (least CPU utilization, least bandwidth utilization, and least current Sessions) and non-state-based (random and round robin) policies. The first class of policies dispatch user requests based on their knowledge of how resources are being utilized by the individual servers in a cluster. The second class of policies do not require this knowledge and dispatch requests either randomly or in a round robin fashion. Except the least session policies, we have not observed a remarkable difference between these policies concerning the performance or power consumption or resource utilization of the servers.

The least session policy, however, produced a high throughput and a low power consumption, but resources (CPUs) were not utilized fairly. One reason for this is that, the least session policy avoids a surge in connection requests, reducing the cost of session establishment. Another reasons is that of all the context information used to forward requests, the session information stalled less quickly.

The main reason for the inefficient resource consumption of the multimedia servers was the bottleneck created by the network bandwidth. We investigated two approaches and recommend another to deal with this problem. The first approach was to increase the bandwidth from 1 Gbit/s to 10 Gbit/s which doubled the throughput while increasing the power consumption by 1.25% only. The second approach was to scale the CPU frequency while leaving the bandwidth as it was. This reduced the overall power consumption of the servers by 12% without affecting the expected throughput. The third approach, which we have not investigated in this paper but will do so in the future is to schedule services which have complementary resource requirements.

Acknowledgment

The authors would like to acknowledge the German Research Foundation (DFG) for partially funding this work under agreement SFB 912/1 2011.

Mr. Syed Kewaan Ejaz was partially responsible for setting up the experiment environment and for undertaking a part of the experiment.

References

[1] Smart 2020: Enabling the low carbon economy in the information age. Technical report, 2008.

[2] L. A. Barroso and U. Hölzle. The case for energy-proportional computing. *Computer*, 40:33–37, December 2007.

[3] V. Cardellini, M. Colajanni, and P. S. Yu. Dynamic load balancing on web-server systems. *IEEE Internet Computing*, 3:28–39, May 1999.

[4] J. S. Chase, D. C. Anderson, P. N. Thakar, A. M. Vahdat, and R. P. Doyle. Managing energy and server resources in hosting centers. In *Proceedings of the eighteenth ACM symposium on Operating systems principles*, SOSP '01, pages 103–116, New York, NY, USA, 2001. ACM.

[5] G. Chen, W. He, J. Liu, S. Nath, L. Rigas, L. Xiao, and F. Zhao. Energy-aware server provisioning and load dispatching for connection-intensive internet services. In *Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation*, NSDI'08, pages 337–350, Berkeley, CA, USA, 2008. USENIX Association.

[6] W. Dargie, A. Strunk, and A. Schill. Energy-aware service execution. In *The 36th Annual IEEE Conference on Local Computer Networks*, 2011.

[7] E. N. Elnozahy, M. Kistler, and R. Rajamony. Energy-efficient server clusters. In *Proceedings of the 2nd international conference on Power-aware computer systems*, PACS'02, pages 179–197, Berlin, Heidelberg, 2003. Springer-Verlag.

[8] D. Gmach, J. Rolia, L. Cherkasova, and A. Kemper. Resource pool management: Reactive versus proactive or let's be friends. *Comput. Netw.*, 53:2905–2922, December 2009.

[9] X. Hei, C. Liang, J. Liang, Y. Liu, and K. Ross. A measurement study of a large-scale p2p iptv system. *IEEE Transactions on Multimedia*, 9(7):1672–1687, 2007.

[10] S. Huang and W. Feng. Energy-efficient cluster computing via accurate workload characterization. In *Proceedings of the 2009 9th IEEE/ACM International Symposium on Cluster Computing and the Grid*, CCGRID '09, pages 68–75, Washington, DC, USA, 2009. IEEE Computer Society.

[11] J. Moore, J. Chase, P. Ranganathan, and R. Sharma. Making scheduling "cool": Temperature-aware workload placement in data centers. In *USENIX'05*, 2005.

[12] V. Pallipadi and A. Starikovisky. The ondemand governer. Proceedings of the Linux Symposium (volume two), 2006.

[13] S. Srikantaiah, A. Kansal, and F. Zhao. Energy aware consolidation for cloud computing. In *Proceedings of the 2008 conference on Power aware computing and systems*, HotPower'08, pages 10–10, Berkeley, CA, USA, 2008. USENIX Association.

[14] B. Yagoubi and Y. Slimani. Dynamic load balancing strategy for grid computing. *Trans. Engin., Comput. Technol. (World Acad. Science)*, 13:260–265, 2006.