ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)
Vol 3, Issue 10, October 2016

Energy Efficient Utilization of Resources in Cloud Computing Systems

[1]Saurabh Kumar, [2]Bharti Vidhury, [3]Neha Bhadana
M .Tech Scholar, Department of Computer Science and Engineering,
Dr. A.P.J Abdul Kalam Technical University, Lucknow (U.P),

Abstract: **-- Abstract the energy consumption of under-utilized resources, particularly in a cloud environment, accounts for a substantial amount of the actual energy use. Inherently, a resource allocation strategy that takes into account resource utilization would lead to better energy efficiency; this, in clouds, extends further with virtualization technologies in that tasks can be easily consolidated. Task consolidation is an effective method to increase resource utilization and in turn reduces energy consumption. Recent studies identified that server energy consumption scales linearly with (processor) resource utilization. This encouraging fact further highlights the significant contribution of task consolidation to the reduction in energy consumption. However, task consolidation can also lead to the freeing up of resources that can sit idling yet still drawing power. There have been some notable efforts to reduce idle power draw, typically by putting computer resources into some form of sleep/power-saving mode. In this paper, we present two energy-conscious task consolidation heuristics, which aim to maximize resource utilization and explicitly take into account both active and idle energy consumption. Our heuristics assign each task to the resource on which the energy consumption for executing the task is explicitly or implicitly minimized without the performance degradation of that task. Based on our experimental results, our heuristics demonstrate their promising energy-saving capability.**

Keywords: **-- Cloud computing · Energy aware computing · Load balancing · Scheduling**

I. INTRODUCTION

Cloud computing has become a very promising paradigm for both consumers and providers in various fields of endeavor, such as science, engineering and business. A cloud typically consists of multiple resources possibly distributed and heterogeneous. Although the notion of a cloud existed in one form or another for some time now (its roots can be traced back to the mainframe era [1]), however, recent advances in virtualization technologies in particular have made it much more compelling compared to the time when it was first introduced. The adoption and deployment of clouds has many attractive benefits, such as scalability and reliability; however, clouds in essence aim to deliver more economical solutions to both parties (consumers and providers). By economical we mean that consumers only need to pay for what resources they need while providers can capitalize poorly utilized resources. From a provider's perspective, the maximization of the profit is a high priority. In this regard, the minimization of energy consumption plays a crucial role. Moreover, energy consumption can be much reduced by increasing resource utilization. Energy usage in large-scale computer systems like clouds also yields many other serious issues including carbon emissions and system reliability.

II MODELS

In this section, we describe the cloud, application and energy models, and define the task consolidation problem targeted in this work. The details of the model presented in this section focus on resource management characteristics and issues from a cloud provider's perspective.

2.1 Cloud model

The target system used in this work consists of a set R of r resources/processors that are fully interconnected in the sense that a route exists between any two individual resources (Fig. 1). We assume that resources are homogeneous in terms of their computing capability and capacity; this can be justified by using virtualization technologies. Nowadays, as many-core processors and virtualization tools (e.g., Linux KVM, VMware Workstation & VMware Fusion, Xen, Parallels Desktop for Mac, VirtualBox) .

2.2 Application model

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)
Vol 3, Issue 10, October 2016

Services offered by cloud providers can be classified into software as a service (SaaS), platform as a service (PaaS) and infrastructure as a service (IaaS). Note that, when instances of these services are running, they can be regarded as computational tasks or simply tasks. While IaaS requests are typically tied with predetermined time frames (e.g., pay-per-hour), requests of SaaS and PaaS are often not strongly tied with a fixed amount of time (e.g., pay-per-use). However, it can be possible to have estimates for service requests for SaaS and PaaS based on historical data and/or consumer supplied service information. Service requests in our study arrive in a Poisson process and the requested processing time follows exponential distribution. We assume that the processor/CPU usage (utilization) of each service request can be identifiable. It is also assumed that disk and memory use correlates with processor utilization [6]. Hereafter, application, task and service are used interchangeably.

2.3 Energy model

Our energy model is devised on the basis that processor utilization has a linear relationship with energy consumption. In other words, for a particular task, the information on its processing time and processor utilization is sufficient to measure the Energy efficient utilization of resources in cloud computing systems energy consumption for that task. For a resource ri at any given time, the utilization Ui is defined as Ui =n j=1 ui,j (1) where n is the number of tasks running at that time and ui,j is the resource usage of a task tj . The energy consumption Ei of a resource ri at any given time is defined as Ei = (pmax − pmin) × Ui + pmin (2) where pmax is the power consumption at the peak load (or 100% utilization) and pmin is the minimum power consumption in the active mode (or as low as 1% utilization). In this study, we assume that resources in the target system are incorporated with an effective power-saving mechanism (e.g., [8]) for idle time slots; this results in the difference in energy consumption of resources between active and idle states being significant. Specifically, the energy consumption of an idle resource at any given time is set to 10% of pmin. Since the overhead to turn off and back on takes a nonnegligible amount of time, this option for idle resources is not considered in our

2.4 Task consolidation problem

The task consolidation (also known as server/workload consolidation) problem in this study is the process of assigning a set N of n tasks (service requests or simply services) to a set R of r cloud resources—without violating time constraints—aiming to maximize resource utilization, ultimately to minimize energy consumption. Here, time constraints directly relate to resource usage associated with tasks; that is, the resource allocated to a particular task must sufficiently provide the resource usage of that task. For example, a task with its resource usage requirement of 60% cannot be assigned to a resource for which the resource utilization at the time of that task's arrival is 50%.

III RELATED WORK

As cloud and green computing paradigms are closely related and they are gaining their momentum, the energy efficiency of clouds has become one of most crucial research issues. Advancements in hardware technologies [10], such as low–power CPUs, solid state drives, and energy–efficient computer monitors have helped relieve this energy issue to a certain degree. In the meantime, there also have been a considerable amount of research conducted using software approaches, such as scheduling and resource allocation [11–17] and task consolidation [18–21]. The scheduling and resource allocation approach is primarily enabled using slack reclamation with the support of dynamic voltage/frequency scaling (DVFS; more specifically 'processor undervolting') [22] incorporated into many recent commodity Y.C. Lee, A.Y. Zomaya processors. This technique temporarily decreases voltage supply level at the expense of lowering processing speed. Slack reclamation is made possible primarily by recent DVFS-enabled processors and the parallel nature of the deployed tasks. For example, when the execution of a task is dependent on two predecessor tasks and these two tasks have different completion times, the predecessor task with an earlier completion time can afford additional run–time (slack); this slack can then be exploited using undervolting for energy saving. Since most DVFS-based energy-aware scheduling and resource allocation techniques are static (offline) algorithms with an assumption of tight coupling between tasks and resources (i.e., local tasks and dedicated resources),

IV TASK CONSOLIDATION ALGORITHM

Consolidation is an effective means to manage resources particularly in clouds both in the short and

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)
Vol 3, Issue 10, October 2016

long terms. In the short term case, volume flux on incoming tasks can be "energy-efficiently" dealt with by reducing the number of active resources, and putting redundant resources into a power-saving mode or even turning off some idle resources systematically. In the long term case, cloud infrastructure providers can better model/provision power and resources; this alleviates the burden of excessive operational costs due to over provisioning. The focus in this paper on the short term case, even though task consolidation results our algorithms deliver can be used as an estimator in the long term provisioning case. In this section, we present two energy-conscious task consolidation algorithms, ECTC and MaxUtil. They are in fact described side by side since they share several common features with the main difference being whether energy consumption is taken into account explicitly or implicitly. In other words, MaxUtil makes task consolidation decisions based on resource utilization, which is a key indicator for energy

4.1 Algorithm description

Both ECTC and MaxUtil follow similar steps (Fig. 2) with the main difference being their cost functions. In a nutshell, for a given task, two heuristics check every resource and identify the most energy-efficient resource for that task. The evaluation of the most energy-efficient resource is dependent on the used heuristic, or more specifically the cost function employed by the heuristic. The cost function of ECTC computes the actual energy consumption of the current task subtracting the minimum energy consumption (pmin)—required to run a task—if there are other tasks running in parallel with that task. That is, the energy consumption of the overlapping time period among those tasks and the current task is explicitly taken into account. The cost function tends to discriminate the task being executed alone. The value fi,j of a task tj on a resource ri obtained using the cost function of ECTC is defined as: $f_{i,j} = (p \times u_j + p_{min}) \times \tau 0 - (p \times u_j + p_{min}) \times \tau 1 + p \times u_j \times \tau 2$

4.2 Performance analysis and discussion

As incorporated into our energy model, energy consumption is directly proportional to resource utilization. At a glimpse, for any two task-resource matches, the one with a higher utilization may be selected. However, since the determination of the right match is not entirely dependent on the current task, ECTC makes its decisions based rather on the (sole) energy consumption of that task. In Fig. 3a, task 3 (t3) arrives at time 14 after tasks 0, 1 and 2, and it is assigned onto resource 1 (r1) based on energy consumption (40 with pmax and pmin of 30 and 20, respectively), even though the utilization of resource 0 (r0) is higher (64%, but energy consumption is 80) if t3 is assigned on r0. On the other hand, there are cases in which matches with higher utilization in fact (lead to) consume less energy (Fig. 4b). MaxUtil assigns t3 onto r0 and this leads to a better match for t4 compared with ECTC (Fig. 3b). These contrary situations are often exhibited due to the dynamic nature of clouds—the decision for a given newly arrived task is made based on the current state of task-resource bindings, and thus the decision is only a local optimum. The superiority of performance of our heuristics can be hardly determined since the quality of their task consolidation decisions in any given period of time may differ with characteristics of subsequent tasks after that period.

V CONCLUSION

Task consolidation particularly in clouds has become an important approach to streamline resource usage and in turn improve energy efficiency. Based on the fact that resource utilization directly relates to energy consumption, we have successfully modeled their relationship and developed two energy-conscious task consolidation heuristics. The cost functions incorporated into these heuristics effectively capture energy-saving possibilities and their capability has been verified by our evaluation study. The results in this study should not have only a direct impact on the reduction of electricity bills of cloud infrastructure providers, but also imply possible savings (with better resource provisioning) in other operational costs (e.g., rent for floor space). Of course, the reduction in the carbon footprint of clouds is another important spinoff. Acknowledgements Professor Zomaya's work is supported by an Australian Research Grant DP1097110.

REFERENCES

1. Parkhill D (1966) The challenge of the computer utility. Addison-Wesley Educational, Reading

2. Koomey JG (2007) Estimating total power consumption by servers in the U.S. and the world. Lawrence Berkeley National Laboratory, Stanford University

3. Koch G (2005) Discovering multi-core: Extending the benefits of Moore's law. Technology@Intel Magazine, (http://www.intel.com/technology/magazine/computing/multi-core-0705.pdf)

4. Barroso L, Holzle U (2007) The case for energy-proportional computing. IEEE Comput

5. Bohrer P, Elnozahy E, Keller T, Kistler M, Lefurgy C, Rajamony R (2002) The case for power management in web servers. Power Aware Comput 261–289

6. Fan X, Weber X-D, Barroso LA (2007) Power provisioning for a warehouse-sized computer. In: Proc 34th annual international symposium on computer architecture (ISCA '07), 2007, pp 13–23

7. Lefurgy C, Wang X, Ware M (2007) Server-level power control. In: Proc IEEE international conference on autonomic computing, Jan 2007

8. Meisner D, Gold BT, Wenisch TF (2009) PowerNap: eliminating server idle power. In: Proc 14th international conference on architectural support for programming languages and operating systems (ASPLOS '09), 2009, pp 205–216

9. Microsoft Inc (2009) Explore the features: performance. http://www.microsoft.com/windows/windows-vista/features/performance.aspx

10. Venkatachalam V, Franz M (2005) Power reduction techniques for microprocessor systems. ACM Comput Surv 37(3):195–237

11. Lee YC, Zomaya AY (2009) Minimizing energy consumption for precedence-constrained applications using dynamic voltage scaling. In: Proc the international symposium on cluster computing and the grid (CCGRID '09), 2009, pp 92–99

12. Kim KH, Buyya R, Kim J (2007)