Empowering Automatic Data-Center Management with Machine Learning

Josep Ll. Berral
Universitat Politècnica de Catalunya
Barcelona, Spain
berral@ac.upc.edu

Ricard Gavaldà
Universitat Politècnica de Catalunya
Barcelona, Spain
gavalda@lsi.upc.edu

Jordi Torres
Univ. Politècnica de Catalunya
Barcelona Supercomp. Center
Barcelona, Spain
torres@ac.upc.edu

ABSTRACT

The Cloud as computing paradigm has become nowadays crucial for most Internet business models. Managing and optimizing its performance on a moment-by-moment basis is not easy given as the amount and diversity of elements involved (hardware, applications, workloads, customer needs...). Here we show how a combination of scheduling algorithms and data mining techniques helps improving the performance and profitability of a data-center running virtualized web-services. We model the data-center's main resources (CPU, memory, IO), quality of service (viewed as response time), and workloads (incoming streams of requests) from past executions. We show how these models to help scheduling algorithms make better decisions about job and resource allocation, aiming for a balance between throughput, quality of service, and power consumption.

Categories and Subject Descriptors

C.0 [**Computer Systems Organization**]: *Modeling of computer architecture*; I.2.6 [**Artificial Intelligence**]: Learning—*Induction*; K.6.2 [**Management of Computing and Information Systems**]: Installation Management—*Performance and usage measurement, Pricing and resource allocation*

Keywords

Cloud Computing, Machine Learning, Modeling, Web-Services

1. INTRODUCTION

Cloud Computing has become a crucial model for the externalization of information and IT resources for people and organizations thanks to the "everything as a service" (platform, infrastructure, and services) capabilities. We distinguish three main actors: the cloud service provider (owner of IT resources), the cloud customer (who wants to run services on the cloud), and the final client (who uses the services). The goal of the provider is to provide customers enough resources to fulfill their services Quality of Service (QoS), reducing the amount of used resources to save power.

In order to match services and resources, managers may use low-level measurements (resource, power, and operating system monitors) and high-level data (user behavior and service performance). The scenario is modeled as a set of data-center resources and a set of web services, enclosed in virtual machines (VM), each resource with a maximum quota of usage and energy requirements, and each service with resource requirements (load per time unit), performance requirements, and an execution reward.

As many of the parameters involved in this optimization problem are unknown *a priori* and vary over time, explicit modeling is very difficult. We use data mining and machine learning methods, a more viable option, to create models from past experience for each element in the system (an application type, a workload, a physical machine (PM), a high-level service requirement). Here we present a methodology for using machine learning techniques (ML) to model the main resources of a web-service based data-center from low-level information, and learn high-level information predictors to drive decision-making algorithms for virtualized service schedulers, without much expert knowledge or real-time supervision. [1]

2. MANAGING DATA-CENTERS

In commercial data-centers the customers can run their services without knowing details of the infrastructure, paying the provider on a usage-basis to ensure a Service Level Agreement (SLA) detailing the QoS among others. The provider enables a VM for the customer to deploy his web-services, and adjusts the VM granted resources. Customers base their business on the clients using the service, so a given QoS for each service must be satisfied (e.g. response time RT). The provider goal is to use as minimal resources for the VMs but granting the VMs will have enough to satisfy the QoS agreed in the SLA. Figure 1 shows the business infrastructure.

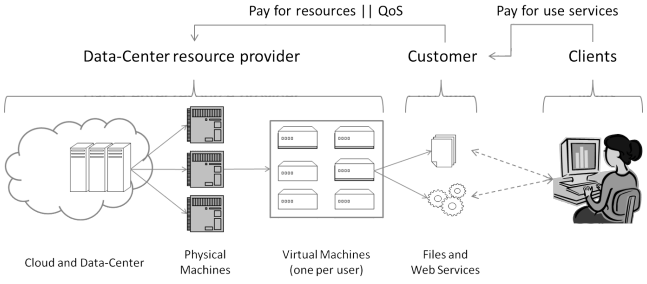

Figure 1: Data-center business infrastructure

[1] An extended version of this work is available on http://www.lsi.upc.edu/dept/techreps/llistat_detallat.php?id=1131

SAC'13 March 18-22, 2013, Coimbra, Portugal.

Our decision maker relies on a middleware such as OpenNebula [6], for monitoring (collecting high- and low-level data) and for acting (managing tasks, workloads, and VM and PM resources). Monitors get the load and resources from PMs and VMs, obtaining the following set of attributes per time unit: timestamps; number of requests; average response times; average requested bytes; and resource usage and bandwidth. From this information we can make decisions and do the following actions: migrate VMs among PMs and adjust VM granted resources.

When making decisions in this context, often the required information 1) is not available, 2) is available but highly uncertain, or 3) cannot be read because of privacy issues. In order to solve these lacks of information and uncertainty, we employ here Machine Learning (ML) methods, setting base for our work in a ML hypothesis: *For each situation, there may be a model obtained by careful expert modeling and tuning better than any ML-learned model. But, for each situation, ML can obtain semi-automatically a model which is as good as or better than a generic model built without intensive expert knowledge or intensive tuning work.*
The advantage of ML over explicit expert modeling is when systems are complex enough that no human expert can explore all relevant possibilities, when no experts exist, or when system changes over time so models must be constantly rebuilt.

3. METHODOLOGY AND LEARNING

First of all we model the VM and PM behaviors (CPU, Memory and IO) from the amount of load received, to be predicted on-line, complementing the decision making algorithm (here the PM×VM scheduler) with extra information. The input data is the load information (e.g. the estimated requests per time unit, the average computing time per request, and the average number of bytes exchanged per request). For the expected CPU and IO usage we selected the M5P algorithm [4], a decision tree holding linear regressions on its leaves, as CPU and IO usage may be in significantly different load regimes, but reasonably linear in each. While for Memory, as being the web-services memory greedy (constantly get memory, then flush memory occasionally), a Linear regression is enough, using the load information and the memory usage from $t-1$.

Secondly we predict the QoS variables (the RT in this case of study). Giving each VM always the maximum resources would not consolidate resources as much as could be, and giving each VM less than the minimum required given the load would degrade the RT. A common "Response Time to QoS function" in SLAs is to set a threshold α and a desired response time RT_0, and set SLA fulfillment level to degrade linearly from 1 to 0 in between RT_0 and $\alpha \cdot RT_0$. Our decision making method (allocator) predicts the degree SLA fulfillment of a VM from its load parameters and its context, i.e. the features of the PM where it is currently or tentatively placed, the load parameters of the VM in the same PM, and the amount of physical resources currently allocated and demanded by each VM. Here we use again the M5P method, since simple linear regressions were incapable of representing the relations between resources and RT.

By learning the function $f(load) \rightarrow \mathrm{E}[CPU, MEM, IO]$, lectures from inside the VM can be replaced, and predict the estimated effective resources required by a VM depending only on its received load without interferences of stress on the VM or occupation on the PM or network. And by learning a function expecting the RT from placing a VM in a PM with a given occupation $f(status, resources) \rightarrow \mathrm{E}[RT]$, scheduler can consolidate VMs without risking the RT in excess, and grant resources playing safe. Figure 2 shows our decision making schema.

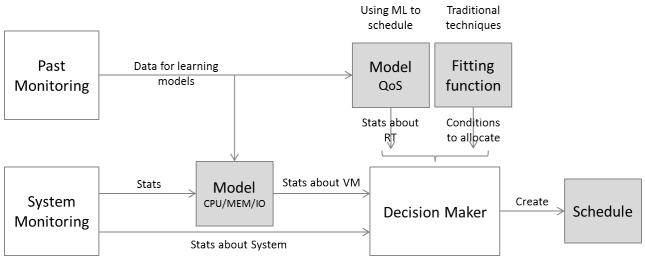

Figure 2: Information flow schema using models

Finally, following the schema from MUSE [2], the data-center benefit optimization problem can be formulated as a Mixed Integer Program (MIP), maximizing the sum of the income from customers per executed VM, minus penalties for SLA degradation, minus power costs by the used (turned on) machines. Due to MIP exponential cost in the number of variables and constraints, solving it becomes unfeasible for realistic settings. Here we use the generic for bin packing problems, Ordered First-Fit and Best-Fit algorithms, also the BackFilling and λ-Round Robin algorithms [3], specialized for load-balancing via consolidation.

All such algorithms use as an oracle used to evaluate how well a VM "will fit" into a PM which has already been assigned some VMs. We substitute the conventional fitting functions (i.e. $cpupm_h + cpuvm_{vm} \leq MaxCPU_h$) by the learned functions mapping tasks descriptions and assigned functions to response times (i.e. is $E[RT_{vm}] \geq \alpha \cdot RT_{0,vm}$, or find best profit according to $E[RT]$).

4. EXPERIMENTS

The details for the learning process are shown in Table 1. An important detail after the learning process is that each model showed the relevance of each attribute over each resource, so operators and architects can also learn from their system (e.g. CPU depends basically on the amount of requests, IO on the average bytes per requests, and Memory depends on its previous state).

We have performed different test to demonstrate how ML can match or improve approximate and ad-hoc algorithms using explicit knowledge, and to validate the models on real machines. The experiments have been performed using real workloads [1] and environments (Intel Xeon 4core + Oracle VirtualBox + XAMPP software) for the model learning process, an analytic simulator (R version for EEFSIM [5]) to compare the different ML-augmented algorithms, and real hosting machines for the model validation. Also for pricing we fixed costs to 0.17 euro/hour (current EC2 pricing in Europe) and power cost to 0.09 euro/KWh (representative of prices with most cloud-providing companies). The services on workload have as RT_0 the values $\in [0.4, 0.12]$s, as experiments on our data-center showed that it is a reasonable response value obtained by the web service without stress or interferences. The initial α parameter is set to 2. Figure 2 show the results for the different algorithms running 20 VMs

	ML Method	Training	Validation	MRE	MAE	StDev	Data range
Predict CPU	M5P ($M = 50$)	3968 inst	7528 inst	0.164	2.530%	4.511	$[2.37, 100.0]$% CPU
Predict MEM	Linear Reg.	107 inst	243 inst	0.0127	4.396 MB	8.340	$[124.2, 488.4]$ MB
Predict IN	M5P ($M = 30$)	1623 inst	2423 inst	0.193	926 Pkts	1726	$[56, 31190]$ #Pkts
Predict OUT	M5P ($M = 30$)	1623 inst	2423 inst	0.184	893 Pkts	1807	$[25, 41410]$ #Pkts
Predict RT	M5P ($M = 4$)	38040 inst	15216 inst	0.00878	9.9 ms	0.0354	$[0, 2.78]$s, $\widehat{RT}$ 17ms

Table 1: Learning details per element. All training processes are done using random split of instances (66/34)

within 20 PMs for a 24 hours workload.

	Euro	Watth/h	Avg.QoS	Migrs	Avg.PMs/h
λRoundRobin	33.94	2114	0.6671	33	9.416
BackFilling	31.32	1032	0.6631	369	6.541
FirstFit	28.77	1874	0.5966	139	6.542
FirstFit+ML	29.99	1414	0.6032	153	5.000
BestFit	29.85	778	0.5695	119	2.625
BestFit+ML	31.771	1442	0.6510	218	4.625

Table 2: Algorithms vs relevant business values

From the results we observe that the versions using the learned model perform similar or better than the versions including expert knowledge, and they approach relatively well to the ad-hoc expert algorithms, backfilling and λ-RR, using the optimal configurations for this kind of data-center calculated in [3]. While ML version of the approximated algorithms are better than their expert-knowledge versions, the Best Fit + ML approach is close to the ad-hoc expert algorithms in QoS and benefit.

After the initial experiments on the simulator, we moved to validate and test the method in a real environment. The set-up consists in a small workbench composed by 5 Intel Xeon machines, 3 as data-center nodes, 1 as gateway and 1 attacking machine reproducing client requests scaled by 100-300 times to produce heavy load, in a different data-center than the previous training.

Using the same machine architecture than the ones for modeling, we could import the learned models for CPU, Memory, and IO. But as network environment was different this time, the RT model had to be learned again. We observed that M5P, in this case, seemed to perform significantly worse than before. We trained a nearest neighbor model, which recovered the previous performance. Let us recall that the contribution we want to emphasize is not the particular models but the methodology: this episode suggests that, methodologically, it is probably a good idea to fix on any particular model kind, and that upon a new environment or system changes, several model kinds should be always tested. Table 3 shows the results comparing Best-Fit versus its ML-augmented version.

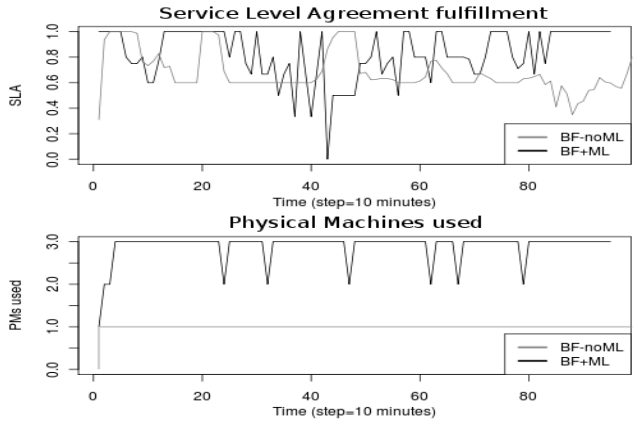

Figure 3: BF-noML vs BF+ML: SLA(RT) and PMs

We can see that best-fit considers that all VMs will fit in CPU and Memory (virtualized and physically) in one machine, which degrades RT. The ML approach, instead, is able to detect from low-level measures situations where RT would not be achieved (because of CPU competition, but also because of memory exhaustion and network/disk competition), hence migrating sufficient VMs to other machines where, for example, network interfaces not so loaded.

5. CONCLUSIONS

In this work we presented a methodology for modeling cloud computing resources of a web-service based data-center using machine learning, obtaining good predictors to empower and drive decision-making algorithms for virtualized job schedulers, without the intervention of much expert knowledge. We observe that the ML-augmented algorithms behave often equal or better than ad-hoc with expert tuning. Response time and quality of service is better maintained on some stress situations when it is possible, by consolidating and de-consolidating by predicting the required computing resources and the resulting RT for a given schedule.

Next steps will focus on scalability and on hierarchically modeling the cloud system as a set of data-centers where services can not only move between machines but among locations around the world. Also we will focus on the network side, including the service time DC-client as another SLA object, bringing the services near their demand.

Acknowledgments

Thanks to *RDlab-LSI* for their support. This work has been supported by the Spanish Ministry of Science under contract TIN2011-27479-C04-03 and under FPI grant BES-2009-011987 (TIN2008-06582-C03-01), by EU PASCAL2 Network of Excellence, and by the Generalitat de Catalunya (2009-SGR-1428).

6. REFERENCES

[1] J. Berral, R. Gavaldà, and J. Torres. Li-BCN Workload 2010, 2011. http://www.lsi.upc.edu/dept/techreps/llistat_detallat.php?id=1099.

[2] J. S. Chase, D. C. Anderson, P. N. Thakar, and A. M. Vahdat. Managing energy and server resources in hosting centers. In *18th ACM SOSP 2001*.

[3] Í. Goiri, F. Julià, R. Nou, J. Berral, J. Guitart, and J. Torres. Energy-aware Scheduling in Virtualized Datacenters. In *12th IEEE Cluster 2010*.

[4] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The WEKA data mining software: an update. *SIGKDD Explor. Newsl.*, 11(1):10–18, 2009.

[5] F. Julià, J. Roldàn, R. Nou, O. Fitó, Vaquè, G. Í., and J. Berral. EEFSim: Energy Efficency Simulator, 2010.

[6] B. Sotomayor, R. S. Montero, I. M. Llorente, and I. Foster. Virtual infrastructure management in private and hybrid clouds. *IEEE Internet Computing*, 13(5):14–22, Sept. 2009.