Profit-Aware Cloud Resource Provisioner for Ecommerce

Nicolas Poggi, David Carrera, and Eduard Ayguadé, and Jordi Torres
Technical University of Catalonia (UPC-BarcelonaTech)
Barcelona Supercomputing Center (BSC)
Barcelona, Spain

***Abstract*—In recent years, the Cloud Computing paradigm has proven effective in scaling dynamically the number of servers according to simple performance metrics and the incoming workload. However while some applications are able to scale-out, as current scaling metrics do not relate system performance to sales, hosting costs and profits are not optimized completely. The following article proposes a novel technique for dynamic resource provisioning based on revenue and cost metrics, to optimize profits for online retailers in the Cloud. The proposal relies on user behavior models that relate Quality-of-Service (QoS) to service capacity, and to the intention of users to buy a product on an Ecommerce site. We show how such metrics can enable *profit-aware* resource management by setting an optimal number of servers at each time of the day. Experiments are performed on custom, real-life datasets from an Ecommerce retailer contain over two years of access, performance, and sales data from popular travel Web applications.**

I. Introduction and Background

In the Cloud, how many Web servers to keep at any given time it is currently an open question, as Web applications can be served at different response times depending on the number of concurrent users by server. However, this decision has an impact on user satisfaction and is directly linked with sales [1]. A problem that arises with dynamic server provisioning, is to decide when to scale in number of servers and to what number. This is especially a problem as the only usable metric by default in AWS is CPU utilization [1]. For Web applications, CPU utilization is not a good indicator of QoS or server load, as the CPU is rarely a bottleneck. To exemplify this situation, for our production datasets, when the servers are overloaded, less than 45% of the CPU is being utilized.

The following article proposes a new technique for dynamic resource provisioning based on a *profit-aware* policy. The policy leverages revenue and cost metrics, produced by Machine Learning techniques, with the intentions to optimize profits for Ecommerce retailers in the Cloud. We base our proposal on user behavior models that relates QoS (response time), to the intention of users to buying a product on an Ecommerce site. In addition to the QoS-to-sales model, we use as inputs the current Cloud costs and a server capacity model to determine the maximum concurrent sessions per server to offer a specific response time. Experiments are performed on custom, real-life datasets from a popular Ecommerce retailer, Atrapalo.com, representative of the current online travel scenario. The datasets contain over two years of access, performance, and sales data from popular travel Web applications.

A. Conversion Rates as a function of Response Time

In Internet marketing, the conversion rate (CR) can be generally defined as the ratio between the number of *'business goal' achievements* and the number of *visits* to a site. In the scope of this study, a *goal* is achieved when a customer purchases during a browsing session. Furthermore, CRs are not static; CRs vary according to time of the day, application, day of the week, or season. In addition, CRs are particularly sensitive to fluctuations in service response times. From the available sales and performance datasets, we have measured for an average day that the CR at 0.4 seconds —the minimum average response time recorded by the application— is at 3%. So at the best Response Time (RT), only 3% of visitors end-up purchasing a product. As RT increases, at 5 seconds, more than 25% of sales would be lost, while at 10 seconds, 80% of sales would be lost, following a non-linear trend [1]. In a previous work we have shown that CRs are usually higher when there are more users in the system, thus, when the incidence of high response times is higher. This situation makes particularly important to give good QoS to incoming sessions at specific times of the day to improve sales.

B. Server Capacity

As Web applications can be served at different QoS (response times), depending mainly on the number of users that consume resources concurrently per server, to be able to set response times to a desired level, first the servers need to be modeled. From the supplied datasets we have found that the average response time to serve Web requests goes from 0.4 —the lowest the application can serve— to 15 seconds, as the number of requests per minute grow from 500 to 3000. The capacity of the servers with this dataset used as input in the prototype implementation follows the logarithmic regression: $y = 664.96ln(x) + 1147.7$

II. Prototype and Experimentation

In [1] we have presented the algorithm for AUGURES, our prototype implementation for profit-aware resource management. AUGURES runs iteratively at fixed time intervals to predict the most optimal server configuration according to the time of the day and the predicted workload. The first step in AUGURES is predicting the expected number of requests and user sessions for the next time interval e.g., 20 minutes. Second, a profit matrix with all the possible server configuration options by *RT* is calculated and returned. The profit matrix contains: the number of servers (cost), conversion rate (sales), and profit for each possible *RT* for the time interval. After obtaining the different profits for each response time combination, the profit matrix is used find the maximum profit according to system and high-level policies; discarding non-matching combinations. Policies can include constraints such as minimum and maximum number of servers (cost), maximum and minimum response times (QoS), target CRs, date and times, and a combination of these features. After constraints are checked, the RT policy yielding the maximum potential profit is selected to scale out or down the current number of servers.

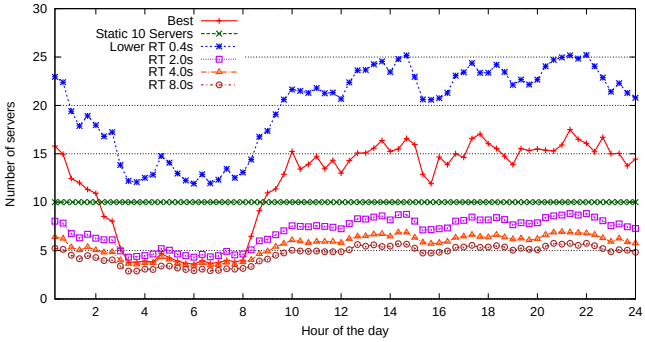

Fig. 1. Number of servers needed by each strategy during a 24hr period

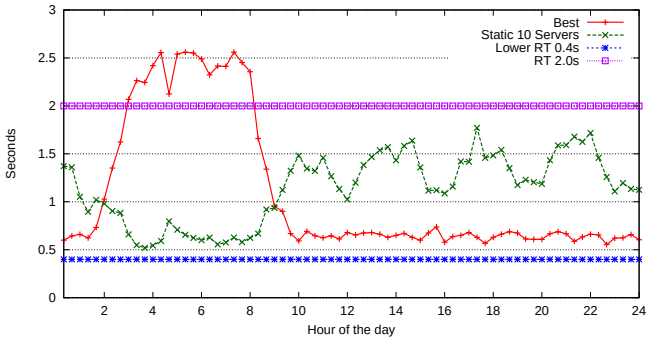

Fig. 2. Response time in seconds for Best, Static 10 servers, Lower, and 2 seconds

To test the applicability of the proposal, we have replayed the supplied datasets into AUGURES to simulate the effect of changing the numbers of servers dynamically during the day and contrast it to different fixed response time and server strategies. The strategies being: *Best*, our proposal of dynamically changing *RTs* during the day; *Lower*, always offering the best response time possible for the application at 0.4 seconds; *Static 10 Servers*, the effect of having a fixed number of Web servers as the company had; the rest represent the strategies of maintaining fixed 2, 4, and 8 seconds following industry standards [1].

Figure 1 shows the number of servers required to achieve each strategy for an averaged 24-hour period. While Figure 2, the corresponding *RT* given by each strategy for the same 24-hour period (skipping 4 and 8 seconds for scale purposes). The *Best* strategy benefits by saving costs in number of servers when the conversion rate (*CR*) is low and it does not compensate to give a better response time (*RT*) to users. It does so by varying the *RT* during the day; the resulting number of servers needed for every 20-minute period as can be seen in Figure 1. During the night and early morning, it offers up to 2.5s in response time as conversion rates are low during these times. This is due to most of the traffic during these hours corresponds to crawlers and other automated bots, and the *CR* is very low [1] at this time. During daytime, the chosen *RT*s are between 0.6 and 0.8 seconds, it does not offer the best *RT* (0.4 s.) as the ratio of user sessions per server and the *CR* improvement is not the most profitable according to the server capacity model. On the contrary it lowers the number of required servers closer to the rest of the fixed *RT* strategies as shown in Figure 1.

Figure 2 also shows the dynamic *RT* under a fixed number configuration of 10 servers, as in a static cluster without dynamic provisioning. Under a static configuration, the RT is dependent on the number of sessions arriving to the system. It can be seen that during night time, the response time is low, while during the day the response time is at 1.5 seconds in average, losing a small percentage of sales. As a consequence of increasing traffic or a traffic spike, a static cluster configuration will degrade response times —losing sales—, but keeping fixed server cost prices. While a dynamic policy would increase hosting prices temporarily —according to maximum defined values— to accommodate load, and keep the desired *RT*.

From Figure 1 it can also be seen that the difference in number of servers needed from the fixed strategies of 2 to 8 seconds is minimal, caused by the logarithmic behavior of the server capacity model. While for retailers such as Atrapalo.com it is profitable to give low *RT*s, the number of servers needed to serve the site at 2 seconds corresponds to 8 servers during daytime. About one third of required servers for the *Lower* strategy, where up to 25 servers are needed. When the number of server is fixed i.e., 10 servers, the opposite effect can be seen as the *RT* will vary according to the number of users. Resulting in the worst response time when there are more users in the system and the *CR*s higher [1].

III. Conclusions

We have presented a proof-of-concept method which leverages economical metrics relating costs and revenues, to answer the questions of when to scale in number of server and to what number. As well as setting the most convenient response time at each time of the day. We do so by relating service response times to user satisfaction that leads to sales. The approach was tested with a prototype implementation, AUGURES, by predicting the expected number of visitors, and the potential profits to be obtained from the workload according to the expected conversion rates for that date and time of the day. In our results a dynamic policy of changing the response time during the day outperforms the baseline policy of maintaining two-second response time as performance target by 52% in profits. Furthermore, if the company had a policy of keeping the lowest possible response time for users, our *Best* strategy would outperform it in profits by by saving 40% in servers, by degrading slightly the response time following a server capacity model. The presented technique can enable *profit-aware* resource management as shown in our experiments; allowing the *self-configuration* of Ecommerce retailers in the Cloud to an optimal number of servers. This way, potentially reducing hosting costs for an incoming workload. While also keeping a high throughput in sales, following high-level policies of business expectations. For further information, experiments, and bibliography, please refer to [1].

Acknowledgements

We thank Atrapalo.com for the datasets, feedback, and domain knowledge for this study. This work is partially supported by the Ministry of Science and Technology of Spain under contracts TIN2012-34557 and 2014SGR1051.

References

[1] N. Poggi, D. Carrera, E. Ayguadé, and J. Torres. Profit oriented fine-grain web server management. *Technical Report: UPC-DAC-RR-2013-60*, November, 2013.