Pricing and Autonomic Control of Web Servers with Time-Varying Request Patterns

Yiyu Chen Amitayu Das Natarajan Gautam Qian Wang Anand Sivasubramaniam
The Pennsylvania State University, University Park, PA 16802.
e-mail: {*adas, anand*}*@cse.psu.edu*, {*yzc107, ngautam, quw6*}*@psu.edu*

1. Introduction

The motivation for this paper stems from two important emerging trends: (i) the growth of commercial services on the Internet, and (ii) the need for deploying systems that can automatically tune themselves. The need for automatic tuning for price (charged by the service provider) becomes particularly acute not only because of high costs of involving humans in system operation but also because such tuning should be nimble enough to adapt to workload changes. This paper introduces a dynamic pricing structure which is fair to both clients and servers. It also examines techniques for dynamically setting the price. During widely time-varying loads [1], if the server charges a fixed price regardless of load, it can result in either being unfair to the clients or a loss in revenue earned. Again, fixing the price to a low value can decrease the revenue that a server could potentially make.

Our first contribution in this paper is a combination of pricing and admission control strategy for such commercial services that take the above-mentioned factors into consideration. In the first step, the customer/client is offered a price package. The second step enforces a level of admission control imposed by the server. It can so happen that a client despite the high price, chooses to get service at a time of high load. The server can use the second step as a way of rejecting this client if it feels that the revenue it would make from this client can be offset by the lower revenue due to degradation in QoS for others. The second contribution in this paper is in developing and evaluating three heuristics for dynamically modulating the above pricing/admission-control strategy to maximize revenue. Some of the factors that need to be taken into consideration when designing techniques for modulating the parameters are revenue enhancement (for the server), implementability and computational complexity. We specifically focus on a web service on the internet that needs to cater to the http file requests of a large number of clients on a continuous basis.

2. System Model and Assumption

A simple web service scenario with a single web server is considered in this paper. The 2-level architecture is illustrated in Figure 1. The server dynamically (or statically) determines the price to charge each request. The clients will be informed the current price when they arrive. The clients can choose to accept the price and enter the system or choose to leave the system if the price is too high for them. After the clients enter the system, the server can impose a second-level admission control and reject requests to improve system performance and service profit. For example, this second-level admission control can be based on the state of the system such as the queue length (the request will be rejected if the queue length exceeds a threshold value) or based on certain profit constraints (e.g. the request will be rejected if profit becomes negative).

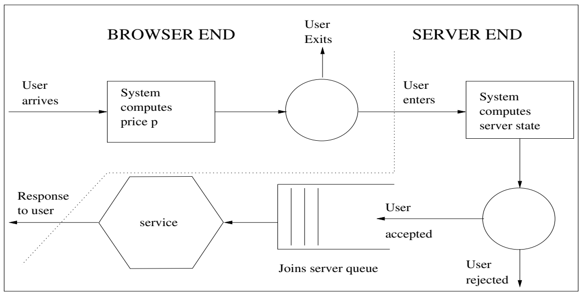

Figure 1. System Architecture for 2-level Control

In this paper, we consider a representative demand curve characterizing the relation of price (p) versus the probability (q) that an arriving customer would accept the price and enter the system. In particular we consider a decreasing, differentiable and invertible function $f(\cdot)$ such that

$$q = f(p). \tag{1}$$

Our control design and optimization schemes do not depend on the choice of $f(\cdot)$.We chose $f(p) = 1 - p^2/G^2$ (G is a constant). Let $\bar{\lambda}$ be the customer arrival rate, then the expected number of customers that accept the price in a unit of time is $\lambda = q\bar{\lambda} = f(p)$. The effective arrival rate that actually enters the system is λ. The web server serves the requests in an FCFS manner. For each request, the server (i) charges a fixed price p as aforementioned, (ii) charges a rate a per unit time of service that the user requests (i.e. the charge of a request is proportional to the byte size of the requested file), and (iii) pays a penalty with a rate c per unit of down experienced by an arrival into the system. Let S be the service time requirement of a request and W be the response time experienced by the user. Then, the expected revenue gained by server from a single request is $p + aE[S] - cE[\frac{W}{S}]$. Assuming that the service time requirement is independent and identically distributed (iid), the expected revenue per unit time (R) for the server is:

$$R = \lambda\Big(p + aE[S] - cE\Big[\frac{W}{S}\Big]\Big). \tag{2}$$

3. Admission Control Strategies

The main criterion for deciding pricing and control strategy is revenue obtained by the web service provider. Customer QoS (in terms of slowdown) is taken into account by charging a penalty. By autonomically setting price as well as admitting arriving requests, the revenue gained by the web server is maximized. We assume that workload information in terms of arrival rates are not available. We use models based on time-series analysis to predict the mean arrival rates based on recent history and seasonality effects. For our analysis we assume that service times are independent and are sampled from an appropriate distribution (such as Pareto) whose parameters are obtained from the estimates of mean and variance of service times. For our calculations, we use real traces which include size of the document requested. We develop three heuristics and compare them with a base line "do nothing" strategy. The next four subsections describe the four strategies.

3.1. Strategy 1: Do nothing

This strategy is called "do nothing" because we do nothing to control requests. All requests are assigned price $p = 0$ in every interval and admitted into the server. Thus, this strategy serves as a benchmark as it most closely represent the current web server implementation.

3.2. Strategy 2: Queue length threshold and static price

We charge all users a static price p, and we reject customers whenever the number in the system exceeds L. We use a trial-and-error technique to obtain the p and L values.

3.3. Strategy 3: Optimal price under no drop

Here, we solve the optimal price p to charge in each interval (in our case of 1 hour length) to maximize R in Equation (2), however no requests are rejected at the admission control stage. Notice that this scheme requires lesser information than Strategy 2 as it does not need to maintain any real-time system state information. In a given interval let $\bar{\lambda}$ be the estimated potential request arrival rate. We calculate λ, the actual request arrival rate such that $q = \lambda/\bar{\lambda}$ is the fraction of traffic that accepts price p and enters the system. Based on the user-defined function $f(p)$ in Equation (1) between price p and acceptance probability q, we know that given a p, $\lambda = f(p)\bar{\lambda}$. We assume that in a period, the request arrivals are according to a Poisson process with mean rate λ requests per unit time. For an arbitrary arrival into the system, let W_q be the waiting time experienced in the queue (not including service) for this request. Although the response time W and the service time S are correlated, since $W_q = W - S$ is independent of S, the expected slowdown can be written as

$$E\left[\frac{W}{S}\right] = E\left[1 + \frac{W_q}{S}\right] = 1 + E[W_q]E\left[\frac{1}{S}\right].$$

Inserting $p = f^{-1}(\lambda/\bar{\lambda})$ in Equation (2) and assuming a Pareto distributed service time, using the Pollaczek-Khintchine formula (assuming steady state is reached in the 1 hour interval) for W_q, we get the revenue per unit time as

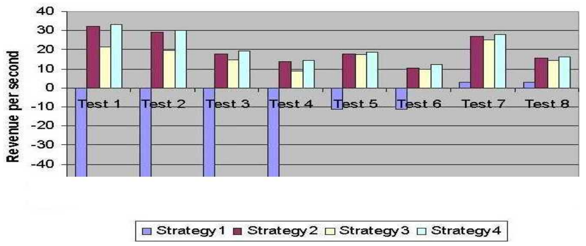

Figure 2. Revenue per second using the 4 strategies

$$R = \lambda\Big(f^{-1}\big(\tfrac{\lambda}{\bar{\lambda}}\big) + a\tfrac{\beta K}{\beta-1} - c - \tfrac{\lambda c K \beta^2(\beta-1)}{2(\beta-2)(\beta+1)(\beta-1-K\beta\lambda)}\Big).$$

Taking the derivative of R w.r.t. λ in above equation and setting it to zero, we obtain the optimal λ^* in the interval $[0, min(\bar{\lambda}, 1/E[S])]$. Thereby the optimal price p^* can be calculated in terms of λ^* by $p^* = f^{-1}(\lambda^*/\bar{\lambda})$.

3.4. Strategy 4: Non-negative profit and static price

This strategy drops requests whenever the revenue from that customer is negative. In achieving this, the server needs to maintain a lot of information such as total service time requirements of all requests ahead and including the new arrival. In addition, we charge all users a fixed and static price p.

4. Evaluation Results

To compare the four strategies, we perform experiments and evaluate the revenue obtained using four strategies. Upon running multiple replications of simulations on the real traces [3], we obtained average revenue per second for four strategies. The results are illustrated in Figure 2. From that, it can be concluded that by rejecting some customers, it is possible to make the system extremely profitable. In addition, it must be noted that in all cases strategy 3 performs reasonably well using very minimal information. Obtaining p for strategies 2 and 4 is very time-consuming through trial and error. For more details, refer to [2].

Acknowledgemenmts

This research is partially supported by NSF grant ACI-0325056 "Data-driven Autonomic Performance Modulation for Servers".

References

[1] C. Lu et. al. Modeling and Performance control of internet servers. In *39th IEEE Conference on Decision and Control*, Sydney, Australia 2000.

[2] Y. Chen et. al. Pricing and Autonomic Control of Web Servers with Time-Varying Request Patterns. Technical Report CSE-04-007, Dept. of CSE, The Penn State University, Feb 2004.

[3] S. Balbach, August 24 - September 03 1995. http://ita.ee.lbl.gov/html/contrib/ClarkNet-HTTP.html.