Poster: Using Fuzzy PI Controller to Provide QoS on Web Servers

Ka Ho Chan, Xiaowen Chu
Department of Computer Science
Hong Kong Baptist University, Kowloon, Hong Kong
{chxw, khchan}@comp.hkbu.edu.hk

With the drastically increasing usage of Web services, it becomes more and more challenging to design Web servers. It is usually not economically feasible to design Web servers for estimated peak load, because the limited Web server resources can never catch up with the increasing user populations. However, it is possible to provide a better service to premium users, even when the Web server is overloaded. Therefore, providing QoS on Web servers becomes a trend in Web server design. Control theory is a powerful tool in designing QoS schemes by adaptively allocating resources to different classes of requests based on sampled feedback.

Proportional delay differentiation is a popular model for differentiating different classes of services. In [1], a classical PI controller is proposed to guarantee the delay ratios between different classes. The Web server is approximately modeled by a second order system. The system parameters are then determined by system identification technique. However, this classical PI controller cannot get satisfactory results on some of the performance metrics, such as settling time and oscillation.

Fuzzy control theory has also been used in the literature [2]. It shows some advantages over classical PI controller. But the main drawback of fuzzy controller is the large amount of parameters to be tuned. It is especially difficult to make initial approximate adjustment, as suggested in [3]. Furthermore, parameters tuning relies on the quality of the expert knowledge.

In this paper, we propose a new control mechanism called fuzzy PI controller to provide proportional delay guarantee. It combines the advantages of fuzzy controller and classical PI controller. In presence of bursty traffic, it can shorten the settling time and decrease oscillation as compared with the pure PI controller. On the other hand, as compared with fuzzy controller, fuzzy PI controller has the nice property that it is very easy to set the controller parameters. Furthermore, it has no risk to yield a nonzero steady state error.

Our study is based on the Apache Web server, which is currently the most popular one [4]. Apache Web server maintains a pool of workers processes[1], each of which serves a single HTTP connection. The total number of workers is limited by the parameter MaxClients. When an HTTP request arrives, it enters the TCP Accept Queue and waits for a free worker. For HTTP/1.0, a worker process only sends a single object to the client, and then becomes free. But since HTTP/1.1, a worker process usually handles the transmission of lots of objects during a single TCP session. Hence a worker process will be hold by a single client for a long time. During heavy traffic, all worker processes may be busy and new requests have to wait until a worker process becomes free. Let the connection delay denote the time interval between the arrival of a connection request and the time that the connection is accepted. Since we are only interested in the connection delay, we simply use the term “delay” in the remaining part of the paper. The focus of this paper is not to shorten the delay, but to guarantee a shorter delay for clients of premium class. This is done by adaptively adjusting the number of processes allocated to different classes of clients.

For simplicity, we only consider two classes of service: the premium class (class 0) and the basic class (class 1). We want to guarantee that the delay ratio between basic class and premium class is maintained at a predefined constant level. This constant value is called “set point”. However, if the traffic load changes in a sudden, it is not an easy task to adjust the number of processes accordingly to maintain the delay ratio at the set point. One difficulty is that, normally we cannot preempt existing HTTP sessions to free a busy worker process.

The delay ratio is controlled by adjusting the number of workers allocated to the two classes adaptively based on the sampled feedback information. The general system architecture is shown in Figure 1. The connection scheduler classifies incoming requests into different classes. The scheduler maintains a FIFO connection queue and a process counter. The process counter denotes the number of processes allocated to that class. E.g., p_0 is the number of processes allocated to premium class, and p_1 is the number of processes allocated to basic class. A new request is allocated to a process only if the current number of consumed processes for that request’s class is less that that class’s process counter. The delay ratio monitor carries out the measurements of the proportional delay ratio, denoted by C_1 / C_0, where C_1 and C_0 denotes the measured connection delay for class 1 and class 0 respectively.

The detailed structure of the fuzzy PI controller is shown in Figure 2. Its main task is to calculate suitable values of p_0

[1] In Apache 2.0 (or higher), a worker can also be configured as a thread.

and p_1 for the next control loop, based on the measured history record of delay ratios. DR is the delay ratio error, i.e., the difference between the measured delay ratio and the desired delay ratio (i.e., the set point). The two parameters, K_i and K_p, can be obtained by system identification technique and root locus method. Due to limited space, we omit the detailed formulae and fuzzy sets used to calculate the values of p_0 and p_1. For details, please refer to [5].

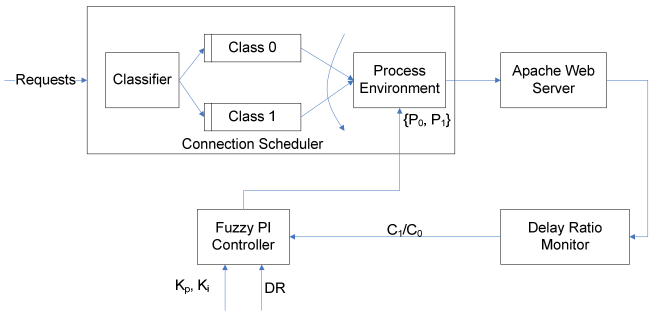

Fig. 1. The System Architecture

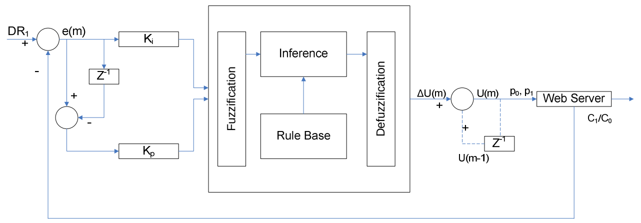

Fig. 2 The structure of the fuzzy PI controller

The overhead of the fuzzy PI controller is very small, because it only contains simple multiplications and additions. Furthermore, the controller only needs to adjust the process assignment once for a sampling period (normally at the order of tens of seconds).

Our experiment platform includes a high-end PC served as Web server and another four PCs served as clients, which are connected by a 1-Gbps Ethernet switch. The client PCs generate Web traffic using the Surge workload generator [6]. We implemented the fuzzy PI controller by modifying the source code of Apache 1.3.9 on Linux platform. The sampling period was set to 30 seconds. At the end of every sampling period, the controller adjusts the number of workers assigned to each class.

At the Web server, the total number of processes was configured to 128. The number of processes allocated to each class was set to 64 at the starting time. The set point is 3. At the clients, in the first 1000 seconds, there are 300 premium class users and 200 basic class users. At 1000th second, the number of users for both classes is increased by 200.

The experimental results are shown in Figure 3. The delay ratios of both controllers deviate from the set point seriously after load changing at 1000th second. After a shorter period of time, the fuzzy PI controller reacted and re-converged to the set point at around 1700th second. However, it takes a longer time for the PI controller to converge to the set point (at around 2200th second). The PI controller also shows a more violent oscillation. A series of experiments with different loading have been done which shown similar results as Figure3.

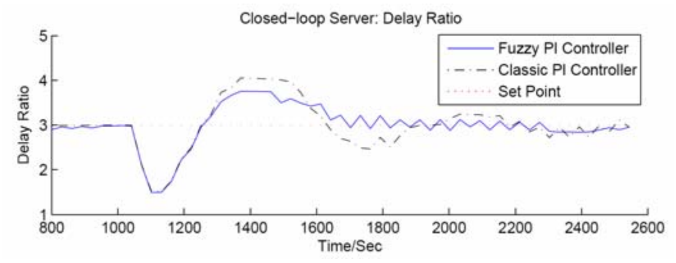

Fig. 3 The delay ratio of the PI and the fuzzy PI controller

Our current work is to design the fuzzy PI controller for a Web server cluster, where a group of back-end servers are connected together to offer high request processing capacity. Load balancing and locality-aware are the two properties of Web server clusters, which make the controller design more challenging than the case of a single node server.

ACKNOWLEDGEMENT

This research was partially supported by Research Grant Council under grant RGC HKBU 210605, and grant FRG/05-06/II34 from Hong Kong Baptist University, Hong Kong.

REFERENCES

[1] C. Lu, Y. Lu, T. F. Abdelzaher, A. Stankovic, and S. H. Son, "Feedback Control Architecture and Design Methodology for Service Delay Guarantees in Web Servers," *IEEE Transactions on Parallel and Distributed Systems*, vol. 17, no. 9, pp. 1014-1027, 2006

[2] Y. Wei, C. Lin, X.-W. Chu, and T. Voigt, "Fuzzy Control for Guaranteeing Absolute Delays in Web Servers," *International Journal of High Performance Computing and Networking, to appear*.

[3] P. Pivonka, "Comparative Analysis of Fuzzy PI/PD/PID Controller Based on Classical PID Controller Approach," In Proc. of *the 2002 IEEE World Congress on Computational Intelligence*, pp. 541-546, 2002.

[4] Apache Software Foundation. http://www.apache.org/.

[5] K.H. Chan and X.-W. Chu, "Design of a Fuzzy PI Controller to Provide Proportional Delay Guarantee for Web Servers," Technical Report, Department of Computer Science, Hong Kong Baptist University, 2006. http://www.comp.hkbu.edu.hk/tech-report/tr06001f.pdf

[6] P. Barford and M. Crovella, "Generating Representative Web Workloads for Network and Server Performance Evaluation," In Proc. of *the ACM SIGMETRICS*, pp. 151-160, November 1998.