A Decision Maker for Transport Protocol Configuration*

Jae-Hyun Hwang, Jin-Hee Choi, and Chuck Yoo

Department of Computer Science and Engineering, Korea University
{jhhwang, jhchoi, hxy}@os.korea.ac.kr

Abstract. This paper proposes an approach, called *Protocol Configuration Decision Maker*, for TCP to dynamically adapt to a network environment. The proposed mechanism monitors the network condition with parameters like loss rate. Then it consults a knowledge database to see whether a better performance can be achieved and replaces a relevant TCP module with the one instructed by the database. Through simulation studies, we show that our mechanism helps TCP achieve better throughput than normal TCP Reno, up to 80∼194%.

1 Introduction

Flexibility and extensibility of network protocol software are getting more important as new networking technologies keep coming up. The configurable and extensible network systems have been proposed in TIP, ADAPTIVE, F-CSS projects[1]. Their goals include: 1) make the extension of new service and protocol easy, and 2) configure the protocol stack based on the application requirements. That is, when an application requests some QoS(Quality of Service)-attributes like performance, reliability, timeliness, security and so forth, the system configures the protocol stack that consists of fine-grained configurable modules in order to satisfy such requirements. However, if the system cannot adapt to dynamically changing environment, the performance degradation would come to be inevitable even though the application requirements are taken into account.

The problem is that the design of existing network systems has never considered network environments as protocol configuration criteria, up to our knowledge. To address the configuration criteria for network environments, this paper proposes a mechanism called Protocol Configuration Decision Maker that chooses the adequate functional module based on network environments. The proposed mechanism classifies the scope of configuration into Slow Start, Congestion Avoidance, Error Recovery like TCP and decides the dominant protocol module affecting the improvement of protocol performance. We evaluate our mechanism through simulation studies, and show that our mechanism helps TCP achieve better throughput than normal TCP Reno significantly.

* This work was supported by grant No.R01-2004-000-10588-0 from the Basic Research Program of the Korea Science & Engineering Foundation and a Korea University grant.

V.N. Alexandrov et al. (Eds.): ICCS 2006, Part IV, LNCS 3994, pp. 172–175, 2006.
© Springer-Verlag Berlin Heidelberg 2006

2 Decision Making for Protocol Configuration

The goal of Protocol Configuration Decision Maker is to dynamically adapt TCP to time-varying networks when the information about network environments is inferred. For the purpose of inferring the network conditions, we use four network parameters: round-trip delay, asymmetric ratio[1], loss rate and loss pattern. Because most of the estimation schemes already have been thoroughly evaluated in the literature, we do not compare or argue the accuracy of them but assume that the network parameters can be estimated precisely.

Now, we explain the process of making the protocol knowledge database for each protocol module. First of all, the throughput of all protocols the system has should be estimated for various network environments. This could be measured by simulations or with experiments in real Internet. Then, we represent the estimated throughput to Expected Throughput as follows.

$$Expected\ Throughput_{(SS\_i,CA\_j,ER\_k)}(RD, AR, LR, LP),$$

where SS_i stands for i-th Slow Start instance, CA_j for j-th Congestion Avoidance instance and ER_k for k-th Error Recovery instance, and RD, AR, LR, LP stand for Round-trip Delay, Asymmetric Ratio, Loss Rate, and Loss Pattern respectively.

Second, we calculate the impact degree of each protocol module on the expected throughput using ANOVA[5]. Let us assume that a system has two different versions of each protocol module. Then we can define three variables x_{SS}, x_{CA} and x_{ER} as follows.

$$x_{SS} = \begin{cases} -1 \ for\ SS\_1 \\ 1 \quad for\ SS\_2 \end{cases}, \quad x_{CA} = \begin{cases} -1 \ for\ CA\_1 \\ 1 \quad for\ CA\_2 \end{cases}, \quad x_{ER} = \begin{cases} -1 \ for\ ER\_1 \\ 1 \quad for\ ER\_2 \end{cases}$$

y, expected throughput, is regressed to x_{SS}, x_{CA} and x_{ER}. q_0 is the mean performance of y, and q_{SS} is the effect of x_{SS}.

$$y = q_0 + q_{SS}x_{SS} + q_{CA}x_{CA} + q_{ER}x_{ER}$$

We obtain the following four combinations of equation using the above model.

$$\begin{aligned} thru\_1 &= q_0 - q_{SS} - q_{CA} - q_{ER} \\ thru\_2 &= q_0 + q_{SS} - q_{CA} - q_{ER} \\ thru\_3 &= q_0 - q_{SS} + q_{CA} - q_{ER} \\ thru\_4 &= q_0 - q_{SS} - q_{CA} + q_{ER} \end{aligned}$$

By solving the equations, we can get q_0, q_{SS}, q_{CA} and q_{ER}. Finally, the impact rate of each protocol module can be calculated as follows.

[1] The asymmetric ratio is defined as a ratio between forward delay and reverse delay in this paper.

$$x_{SS}'s\ impact = \frac{q_{SS}^2}{q_{SS}^2 + q_{CA}^2 + q_{ER}^2}$$
$$x_{CA}'s\ impact = \frac{q_{CA}^2}{q_{SS}^2 + q_{CA}^2 + q_{ER}^2}$$
$$x_{ER}'s\ impact = \frac{q_{ER}^2}{q_{SS}^2 + q_{CA}^2 + q_{ER}^2}$$

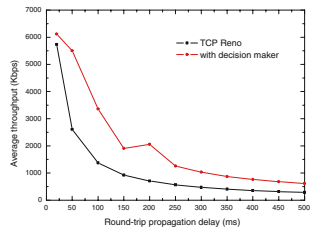
If the system uses [SS_1, CA_1, ER_1] and each protocol module impacts on the throughput improvement by 80%, 15% and 5% respectively, it could get better throughput up to 80% with reconfiguring SS_1 to SS_2.

3 Simulation Result

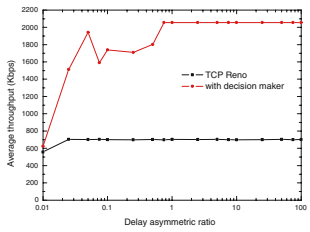
In this section, we validate our decision maker using NS-2 simulator version 2.26[6]. Assuming that the network system supports partial reconfiguration of

Table 1. TCP variants used in simulation

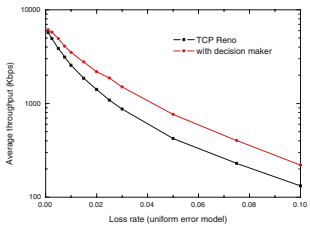
TCP Variant	Slow Start	Congestion	Error Recovery
Reno	Heuristic	AIMD	Duplicate ACK
Westwood[2]	BW Estimate	AIMD	Duplicate ACK
BI[3]	Heuristic	Binary Search	Duplicate ACK
SACK[4]	Heuristic	AIMD	Selective ACK



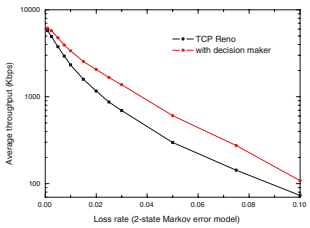
(a) RD: 20∼500(ms)



(b) AR: 0.01∼100



(c) LR: 0.001∼0.1



(d) LP: 2 state Markov

Fig. 1. Average throughput graphs: TCP Reno vs TCP with our decision maker

the transport protocols, we performed the experiment using TCP variants simply to get same results instead of implementing such system. Table 1 shows TCP variants used in our simulation. According to Table 1, we could emulate the partial reconfiguration of protocol modules in the network system that contains two different instances for each protocol module. Each simulation time is 100 seconds and the throughput of each TCP is averaged by results of 100 iterations for the same environment. The default bandwidth, round-propagation delay, asymmetric ratio and loss rate are 10(Mbps), 20(ms), 1.0 and 0.001 respectively.

Through simulation studies, we estimated the average throughput of all TCP variants and calculated the impact rate of each protocol module changing RD, AR, LR and LP. With the impact rate, we could find out the dominant module for throughput improvement. Then, we compared the throughput of 1) the case only using Reno and 2) the case with applied reconfiguration of the protocol modules properly using our decision maker. The average throughput graphs of both cases with various network parameters are shown in Figure 1. As we know from the results, there are always protocol modules that can take the place of Reno's for an improvement of the throughput.

4 Conclusion

In this paper, we proposed Protocol Configuration Decision Maker for TCP to dynamically adapt to time-varying network environments. While the existing flexible and extensible network systems configure the protocol stack based on application requirements, the proposed mechanism is based on network parameters like loss rate. Through simulation studies, we could show that our mechanism helps TCP achieve better throughput than normal TCP Reno, up to 80∼194%.

References

1. Stenfan Böcking, "Object-Oriented Network Protocols," *In Proceedings of IEEE INFOCOM*, 1997.
2. M. Gerla, M. Y. Sanadidi, R. Wang, A. Zanella, C. Casetti, S. Mascolo, "TCP Westwood: Congestion Window Control Using Bandwidth Estimation," *In Proceedings of IEEE GLOBECOM*, Nov. 2001.
3. L. Xu, K. Harfoush and I. Rhee, "Binary increase congestion control for fast long-distance networks," *In Proceedings of IEEE INFOCOM*, 2004.
4. S. Floyd, M. Mahdavi, M. Mathis and J. Widmer, "An Extension to the Selective Acknowledgement(SACK) option for TCP," *IETF*, RFC 2883, 2000.
5. George W. Cobb, "Introduction to Design and Analysis of Experiments," *Springer*, Mar. 1998.
6. ns2 Network Simulator version 2.26, `http://www.isi.edu/nsnam/ns`, 2003.