Neural Network Based Model Reference Controller for Active Queue Management of TCP Flows

Kourosh Rahnami, Payman Arabshahi, Andrew Gray
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, CA 91109
{rahnamai,payman,gray}@jpl.nasa.gov

Abstract—We discuss here, implementation of a Neural Network (NN) based Model Referenced Control (MRC) algorithm to improve transient and steady state behavior of Transmission Control Protocol (TCP) flows and Active Queue Management (AQM) routers in a network setting. Based on a fluid theoretical model of a network, two neural networks are trained to control the traffic flow of a bottleneck router. Results show dramatic improvement of the transient and the steady state behavior of the queuing window length. The results are compared to the traditional RED algorithm and the P and PI controllers of classical control theory.

TABLE OF CONTENTS

1. INTRODUCTION

Active queuing management is an important method to inform traffic sources of possible network problems at early stages of congestion. Using this feedback methodology, sources with heavy demand would reduce their flow rate to avoid network congestion and erratic network behavior. This concept has been the subject of many research papers [1-10]. However, recently much work has been done in modeling a network of Active Queue Management (AQM) routers using fluid dynamics theory [1-3]. Hollot and Misra [3-4] have shown that based on the linear model of a typical network, classical control system theories and design methodologies can be used to improve network performance and avoid network congestion.

2. FLUID-BASED TCP FLOW MODEL

In this section, we show in detail the process of developing the differential equation governing the TCP traffic flow. Fluid models have been used successfully to obtain nonlinear and linear models to describe dynamics of TCP and AQM [2-4]. The following coupled nonlinear differential equations describe the behavior of a typical TCP.

$$\frac{d\dot{W}(t)}{dt} = \frac{1}{R(t)} - \frac{W(t)}{2}\frac{W(t-R(t))}{R(t-R(t))}p(t-R(t)) \quad (1)$$

$$\frac{d\dot{q}}{dt} = \frac{N}{R(t)}W(t) - C \quad (2)$$

$$R(t) = \frac{q(t)}{C} + T_p \quad (3)$$

where W is the expected TCP window length (packets), q is the queue length (packets), R is the round trip time (seconds), C is the link capacity (packets/s), T_p is the propagation delay (seconds), N is the number of TCP sessions, and P is the probability of packet marked/drop. Obviously W and q cannot take negative values and are bounded respectively by $W_{\max}$ and $q_{\max}$, i.e., $W \in [0, W_{\max}]$, and $q \in [0, q_{\max}]$.

Eq. (1) describes the behavior of the window-length. At each time iteration, window length is increased by one packet unless a packet loss occurs, then window length is reduced to half of its current value. Eq. (2) captures the dynamic behavior of the expected queue length and Eq. (3) is a static equation describing the round-trip delay as the sum of the propagation and queuing delays.

3. LINEAR MODEL DERIVATION

In general, for a nonlinear function $x(t)$

$$\dot{x}(t) = f(x(t), u(t), t) \quad (4)$$

Assuming that $f(\cdot)$ is smooth and has continuous derivative around the operating point, and using a Taylor series expansion, we can find the linear model of this nonlinear equation as

[1] 0-7803-8870-4/05/$20.00© 2005 IEEE
[2] IEEEAC paper #1408, Version 3, Updated November 22, 2004

$$f[x(t),u(t),t] = f[x_0(t),x_0(t),t] + f_x(t)\delta x(t) + f_u(t)\delta u(t) + o(\delta x,\delta u) \quad (5)$$

where

$$f_x(t) = \left.\frac{\partial f}{\partial x}\right|_{x_0,u_0}, \; f_u(t) = \left.\frac{\partial f}{\partial u}\right|_{x_0,u_0} \quad (6)$$

Then

$$\dot{\delta}x = f_x\delta x + f_u\delta u \quad (7)$$

Now a linear model [2-3] for Eqs. (1-3) can be derived around a nominal solution $Q_0 = (W_0, R_0, q_0, p_0)$. Let N and R be considered as constants; then Eqs. (1) and (2) can be rewritten as:

$$\frac{d\dot{W}(t)}{dt} = \frac{1}{R} - \frac{W(t)}{2}\frac{W_R}{R}p(t-R) \quad (8)$$

$$\frac{d\dot{q}}{dt} = \frac{N}{R}W(t) - C \quad (9)$$

$$x(W, W_R, q, p) = \frac{1}{R} - \frac{W(t)}{2}\frac{W_R}{R}p(t-R) \quad (10)$$

$$u(W,q) = \frac{N}{R}W(t) - C \quad (11)$$

where $W_R(t) = W(t-R)$.

At operating point Q_0, the steady state conditions for Eqs. (8) and (9) are given by

$$\dot{W} = 0, \quad \Rightarrow 0 = \frac{1}{R_0} - \frac{W_0}{2}\frac{W_0}{R_0}p_0$$

$$W_0^2 p_0 = 2 \quad \Rightarrow \quad W_0^2 = \frac{2}{p_0}, \; p_0 = \frac{2}{W_0^2} \quad (12)$$

$$\dot{q} = 0 \quad \Rightarrow \quad \frac{N}{R_0}W_0 - C$$

$$W_0 = \frac{R_0}{N}C \quad (13)$$

Calculating partial derivatives around the operating point will result in

$$\left.\frac{\partial x}{\partial W}\right|_{(W_0,R_0,p_0,q_0)} = \left.\frac{\partial x}{\partial W_R}\right|_{(W_0,R_0,p_0,q_0)} = 0 - \frac{W_0}{2R_0}p_0$$

$$\left.\frac{\partial x}{\partial W}\right|_{(W_0,R_0,p_0,q_0)} = -\frac{W_0}{2R_0}\frac{2}{W_0^2} = -\frac{1}{R_0W_0}$$

Using Eq. (13)

$$\left.\frac{\partial x}{\partial W}\right|_{(W_0,R_0,p_0,q_0)} = -\frac{1}{R_0W_0} = -\frac{N}{R_0^2C} \quad (14)$$

$$\left.\frac{\partial x}{\partial q}\right|_{(W_0,R_0,p_0,q_0)} = \left.\frac{\partial}{\partial q}\left[\frac{1}{R} - \frac{W(t)}{2}\frac{W_R}{R}p(t-R)\right]\right|_{(W_0,R_0,p_0,q_0)}$$

$$= \left.\frac{\partial}{\partial q}\left[\frac{1}{R} - \frac{W^2}{2R}p\right]\right|_{(W_0,R_0,p_0,q_0)}$$

Using Eq. (3)

$$\left.\frac{\partial x}{\partial q}\right|_{(W_0,R_0,p_0,q_0)} = \left.\frac{\partial}{\partial q}\left[\frac{1}{\frac{q}{C}+T_p} - \frac{W^2}{2\left(\frac{q}{C}+T_p\right)}p\right]\right|_{(W_0,R_0,p_0,q_0)}$$

$$\left.\frac{\partial x}{\partial q}\right|_{(W_0,R_0,p_0,q_0)} = \left.\frac{-\frac{1}{C}}{\left(\frac{q}{C}+T_p\right)^2} + \frac{W^2p\frac{1}{C}}{2\left(\frac{q}{C}+T_p\right)^2}\right|_{(W_0,R_0,p_0,q_0)}$$

$$\left.\frac{\partial x}{\partial q}\right|_{(W_0,R_0,p_0,q_0)} = \frac{-\frac{1}{C}}{R_0^2} + \frac{W_0^2p_0\frac{1}{C}}{2R_0^2}$$

$$= \frac{-\frac{1}{C}}{R_0^2} + \frac{W_0^2\frac{2}{W_0^2}\frac{1}{C}}{2R_0^2} \quad (15)$$

$$= \frac{-\frac{1}{C}}{R_0^2} + \frac{\frac{1}{C}}{R_0^2} = 0$$

$$\left.\frac{\partial x}{\partial p}\right|_{(W_0,R_0,p_0,q_0)} = \left.\frac{\partial}{\partial p}\left[\frac{1}{R} - \frac{W(t)}{2}\frac{W_R}{R}p\right]\right|_{(W_0,R_0,p_0,q_0)}$$

$$= \left[-\frac{W_0}{2}\frac{W_0}{R_0}\right] = -\frac{W_0^2}{2R_0} = -\frac{\left(\frac{R_0}{N}C\right)^2}{2R_0} \quad (16)$$

$$= -\frac{R_0C^2}{2N^2}$$

Partial derivatives of Eq. (9) are

$$\left.\frac{\partial u}{\partial W}\right|_{(W_0,R_0,p_0,q_0)} = \left.\frac{\partial}{\partial W}\left[\frac{N}{R}W - C\right]\right|_{(W_0,R_0,p_0,q_0)} = \frac{N}{R_0} \quad (17)$$

$$\left.\frac{\partial u}{\partial q}\right|_{(W_0,R_0,p_0,q_0)} = \left.\frac{\partial}{\partial q}\left[\frac{N}{\left(\frac{q}{C}+T_p\right)}W - C\right]\right|_{(W_0,R_0,p_0,q_0)}$$

$$= \left.\left[\frac{-NW\frac{1}{C}}{\left(\frac{q}{C}+T_p\right)^2}\right]\right|_{(W_0,R_0,p_0,q_0)}$$

$$\left.\frac{\partial u}{\partial q}\right|_{(W_0,R_0,p_0,q_0)} = \frac{-NW_0\frac{1}{C}}{R_0^2}$$

Using Eq. (13)

$$\left.\frac{\partial u}{\partial q}\right|_{(W_0,R_0,p_0,q_0)} = \frac{-N\frac{R_0}{N}C\frac{1}{C}}{R_0^2} = \frac{-1}{R_0} \qquad (18)$$

Now expressing Eqs. (12) through (18) in the form of Eq. Eq. (5) we obtain

$$\dot{W}(t) = W_0 - \frac{N}{R_0^2 C}\left[\delta W(t) + \delta W(t-R_0)\right] - \frac{R_0 C^2}{2N^2}\delta p(t-R_0)$$

$$\dot{q}(t) = q_0 + \frac{N}{R_0}\delta W(t) - \frac{1}{R_0}\delta q(t)$$

Defining

$$\delta W(t) \equiv W(t) - W_0 \Rightarrow \delta\dot{W}(t) \equiv \dot{W}(t) - W_0$$

$$\delta q(t) \equiv q(t) - q_0 \Rightarrow \delta\dot{q}(t) \equiv \dot{q}(t) - q_0$$

$$\delta p(t) \equiv p(t) - p_0$$

The two linear differential equations describing TCP flow can now be written as

$$\delta\dot{W}(t) = -\frac{N}{R_0^2 C}\left[\delta W(t) + \delta W(t-R_0)\right] - \frac{R_0 C^2}{2N^2}\delta p(t-R_0) \qquad (19)$$

$$\delta\dot{q}(t) = +\frac{N}{R_0}\delta W(t) - \frac{1}{R_0}\delta q(t) \qquad (20)$$

Taking the Laplace transform of these equations

$$s\delta W(s) = -\frac{N}{R_0^2 C}\left[\delta W(s) + e^{-sR_0}\delta W(s)\right] - \frac{R_0 C^2}{2N^2}e^{-sR_0}\delta p(s)$$

$$s\delta q(s) = +\frac{N}{R_0}\delta W(s) - \frac{1}{R_0}\delta q(s)$$

$$\left[s + \frac{N}{R_0^2 C}\left(1 + e^{-sR_0}\right)\right]\delta W(s) = -\frac{R_0 C^2}{2N^2}e^{-sR_0}\delta p(s)$$

$$\left[s + \frac{1}{R_0}\right]\delta q(s) = +\frac{N}{R_0}\delta W(s)$$

The transfer functions $H_{TCP}(s)$ and $H_{queue}(s)$ can be defined as

$$H_{TCP}(s) = \frac{\delta W(s)}{\delta p(s)} = \frac{-\frac{R_0 C^2}{2N^2}e^{-sR_0}}{\left[s + \frac{N}{R_0^2 C}\left(1 + e^{-sR_0}\right)\right]} \qquad (21)$$

$$H_{queue}(s) = \frac{\delta q(s)}{\delta W(s)} = \frac{\frac{N}{R_0}}{\left[s + \frac{1}{R_0}\right]} \qquad (22)$$

If $W_0 >> 1$ [3], which is a reasonable assumption, then the delay term in the denominator of Eq. (21) will not be significant and Eq. (21) can be simplified to

$$H_{TCP}(s) = \frac{\delta W(s)}{\delta p(s)} = \frac{-\frac{R_0 C^2}{2N^2}e^{-sR_0}}{\left[s + \frac{2N}{R_0^2 C}\right]} \qquad (23)$$

Figure 1 shows the block diagram of the TCP model. Transcendental transfer functions such as the delay in the TCP model are much more difficult to handle than rational transfer functions. Most control theory design methodologies are based on linear models of systems. There are many different methods for approximating a time delay inherent in a system by a rational function. One such method is to approximate the exponential function by a Maclaurin series

$$e^{-R_0 s} \cong 1 - R_0 s + \frac{R_0^2 s^2}{2} \qquad (24)$$

$$e^{-R_0 s} \cong \frac{1}{1 + R_0 s + \frac{R_0^2 s^2}{2}} \qquad (25)$$

Another method would be the Pade approximation. Keeping only the first two terms will result in

$$e^{-R_0 s} \cong \frac{1 - \frac{R_0}{2}s}{1 + \frac{R_0}{2}s} \qquad (26)$$

For the remainder of this paper, we will use Eq. (26) as an approximation of the TCP round-trip delay model.

$$H_{TCP}(s) = \frac{\delta W(s)}{\delta p(s)} = \frac{1 - \frac{R_0}{2}s}{1 + \frac{R_0}{2}s}\frac{-\frac{R_0 C^2}{2N^2}}{\left[s + \frac{2N}{R_0^2 C}\right]} \qquad (27)$$

Therefore, the total linear model for the Active Queuing Management (AQM) is given by

$$H_T(s) = H_{TCP}(s)H_{queue}(s) = \frac{1-\frac{R_0}{2}s}{1+\frac{R_0}{2}s}\frac{-\frac{R_0C^2}{2N^2}}{\left[s+\frac{2N}{R_0^2C}\right]}\frac{\frac{N}{R_0}}{\left[s+\frac{1}{R_0}\right]} \quad (28)$$

Now the linear model of TCP window-control, shown in Fig. 2, can be used to design a variety of controllers. Numerous papers have been published addressing this. In this study we will present the results of applying a model referenced neural network controller and compare the performance of this control structure with that of a standard random early detection (RED) and RED with PI. The RED algorithm for controlling AQM applies proportional control with saturation limits as shown if Fig. 3[3]. The RED nonlinear feedback function (Fig. 4) is given by

$$p(q) = \begin{cases} 0 & 0 \le q \le q_{\min} \\ \dfrac{q-q_{\min}}{q_{\max}-q_{\min}}p_{\max} & q_{\min} \le q \le q_{\max} \\ 1 & q_{\max} \le q \end{cases} \quad (29)$$

Figures (3) and (4) and equation (29) clearly describe the details of the RED algorithm. For values of q greater than $q_{\min}$ and less than $q_{\max}$, RED algorithm behaves similar to a proportional feedback. For values outside this range, p will take on the limiting values of zero or one.

4. Model Reference Neural Network

Model Reference control consists of two neural networks as shown if Figure 5. One neural network is used to model the system and one neural network is used to control the system.

Using the plant measurement values, the model network is trained offline. This block estimates the plant behavior, and the output of this block is used to calculate the modeling error. Then the controller is trained such that system response follows a reference model. Controller parameters are updated based on the error signal computed from the system output and the neural network model of the plant.

Each network has two layers with delayed inputs and outputs. Number of delayed inputs can be selected based on complexity of the plant. For all the results presented here two or three delayed inputs and outputs where selected. Normally the number of delayed units in the network is selected proportional to the order of the system. The higher the system order and plant complexity the higher should be the number of delayed inputs and outputs of the neural network to allow the network to properly capture the dynamics of the system.

5. Results

The traditional RED algorithm is shown in Fig. 3. Much research has been performed [5] in implementing traditional control algorithms on this system with documented success in increasing system performance and stability. Fig. 6 shows the block diagram of a PID controller implementation. It has been shown [4] that P and PI controllers improve the overall performance of the system.

MRC RED and PI RED algorithm as shown in Figs. 5 and 6 are implemented using MATLAB/Simulink at the link of a network for active queue management. The considered network topology consists of a single bottleneck link, with channel capacity of C = 2000 pks/s and 40 ms round trip propagation delay.

The network traffic consists of 50 identical long-lived TCP sessions with RED parameters set to, $q_{\min}$ = 200 packets, $q_{\max}$ = 800 packets, and $p_{\max}$ = 0.5.

Fig. 7 shows the simulation result when 10 TCP sessions drop at 5 seconds into the simulation. As can be seen the neural network transient response is much better than the PI controller, which exhibits oscillatory behavior. The difference between the steady state values of the two controllers is due to the difference between the open-loop gains of the two designs. In this study we will not concentrate on matching the steady state values, rather on exploring the transient response and stability behavior exhibited by the two controllers.

As can be seen from Fig. 8, as the number of drop out flows doubles the transient response of the PI controller becomes much worst than the previous case while remaining stable. The NN-MRC controller performs very nicely with no overshoot and a fast rise time of less than one second.

Fig. 9 shows the instability of the PI controller when 30 TCP flows drop out at 5 seconds into the simulation. NN MRC performs with excellent transient response and adapts to dynamic change of the system. For the next set of simulation runs the reference queue length window is increased such that the drop out probability, Eq. (29), is in saturation range and equal to one.

Figs. 10 and 11 show unstable behavior of the PI controller when operated in saturation range of the RED algorithm, and inability of this controller to perform under severe session drop-out. NN-MRC performs very well and exhibits stable behavior with no overshoot and fast rise time of less than 2 s.

Next we retrained the networks several times and the following results show that the steady state behavior of NN-MRC can be improved with proper network training. The network traffic again consists of 50 identical long-lived TCP sessions with RED parameters set to, $q_{\min}$ = 200 packets, $q_{\max}$ = 800 packets, and $p_{\max}$ = 0.5. q_{ref} is set to 560 packets at 5 seconds into the simulation with 10 load factors (TCP sessions) drop out.

[3] Notice the positive feedback sign in Fig. 3. This is due to an inherent 180 phase shift built into the linear model. Because of the negative sign in $H_{TCP}(s)$ in Eq. (23), input and output of this system are out of phase by 180 degrees, and a positive feedback is required.

Comparing Figs. 7 and 11, we can see that retraining the networks did improve the steady response of the NN-MRC matching that of PI controller. After the session drop outs the PI controller again exhibits signs of instability.

Figs. 13, 14, and 15 show how stable the NN-MRC behaves as compared to unstable behavior of the PI controller. Figure 16 shows the structure of the Simulink program that was used to perform the above simulation runs. A special S-function had to be created to allow dynamic change of simulation parameters and to capture the network dynamic changes.

Next we modified the structure of the controller by removing the RED algorithm block from the NN-MRC feedback loop. In this scenario, the neural network has to capture the nonlinear behavior of the RED algorithm as well. This structure is shown in Fig. 17.

With the network traffic of 50 identical long-lived TCP sessions and RED parameters set to, $q_{\min}$ = 200 packets, $q_{\max}$ = 800 packets, and $p_{\max}$ = 0.5. q_{ref} we ran two simulation runs. In these runs we had 2 drop out points, one at 5 and one at 10 seconds into the simulation run. At each stage 10 TCP sessions dropped out.

We can see from Figs. 18 and 19 that with the RED saturating block removed, the response of the NN-MRC has some overshoot but performs gracefully under severe modeling changes as opposed to the unstable behavior of the PI controller.

6. CONCLUSION

In this paper, we have shown that a neural network based model referenced control algorithm improves transient and steady state behavior of AQM controller over classical RED, P, and PI. While P and PI controllers perform very well at the design points, they exhibit instability when system model is changed drastically. On the other hand NN-MRC controllers performed very well even under sever modeling errors or system dynamic changes.

REFERENCES

[1] D. Lin and R. Morris, "Dynamics of random early detection," *Proc. ACM/SIGCOMM*, 1997.

[2] V. Misra, W.-B. Gong, and D. Towsley, "Fluid-based analysis of a network of AQM routers supporting TCP flows with an application to RED," *Proc. ACM/SIGCOMM*, 2000.

[3] C. Hollot, V. Misra, D. Towsley, and W.-B. Gong, "A control theoretic analysis of RED," *Proc. IEEE Infocom*, 2001.

[4] C. Hollot, V. Misra, D. Towsley, and W.-B. Gong, "On designing improved controllers for AQM routers supporting TCP flows," *Proc. IEEE Infocom*, 2001.

[5] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Trans. Networking*, vol. 1, pp. 397- 413, August 1993.

[6] V. Jacobson, "Congestion avoidance and control," *Proc. SIGCOMM '88*, pp. 314–329, August 1988.

[7] M. May, T. Bonald, and J.-C. Bolot, "Analytic evaluation of RED performance," *Proc. IEEE Infocom*, 2000.

[8] M. Christiansen, K. Jeffay, D. Ott, and F.D. Smith, "Tuning RED for web traffic," *Proc. ACM SIGCOMM*, 2000.

[9] F. Paganini, J.C. Doyle, and S.H. Low, "Scalable laws for stable network congestion control," *Proc. Conference on Decision and Control*, 2001.

[10] M. Kisimoto, H. Ohsaki, and M. Murata, "On transient behavior analysis of random early detection gateway using a control theoretic approach," *Proc. Intl. Conf. Control Apps.*, Sept.18-20, 2002 Glasgow, Scotland.

BIOGRAPHY

Kourosh Rahnamai *has more than seventeen years of academic and industrial experience. His research interests are in the areas of Communications, controls, fuzzy, neural-based systems, navigation, optical communications, spacecraft control, systems Engineering, Estimation theory, Linear and Nonlinear Kalman Filters. He has developed Laboratories for low-cost rapid prototyping and hardware in the loop test and validation of complex algorithms and has developed and analyzed mathematical models for complex multi-disciplinary systems.*

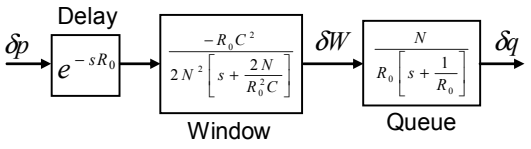

Figure 1: TCP model.

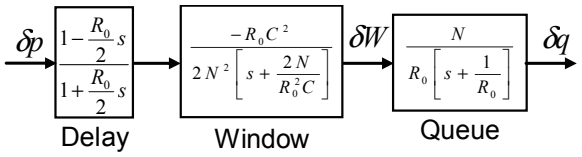

Figure 2: AQM with rational delay model.

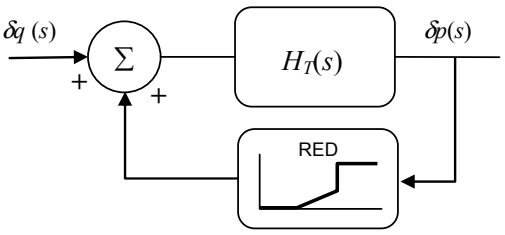

Figure 3: RED feedback block.

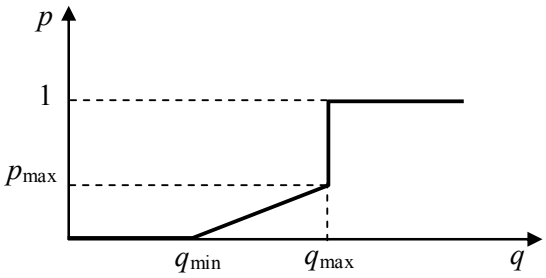

Figure 4: RED Nonlinear feedback function.

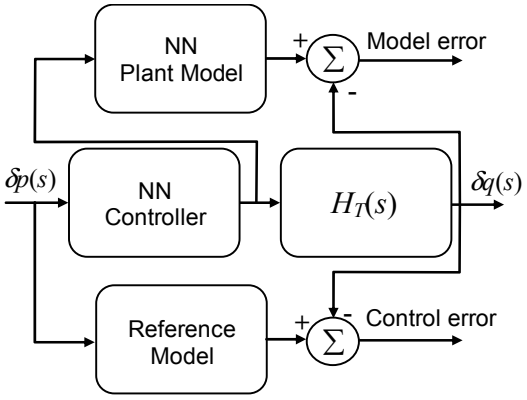

Figure 5: Neural Network MRC.

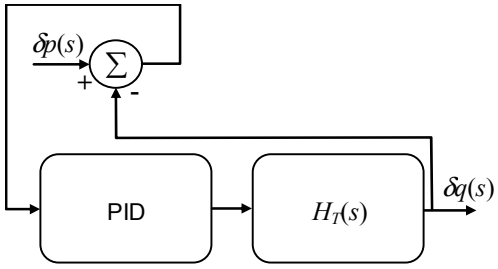

Figure 6: PID controller.

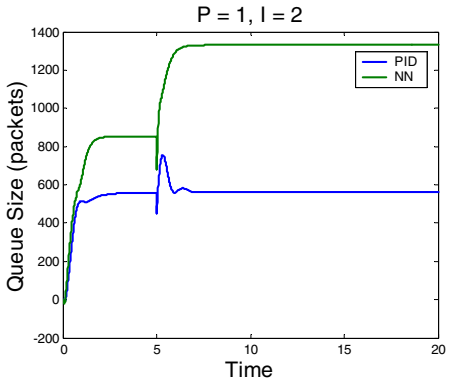

Figure 7: Simulation 1, 10 TCP flow drop out.

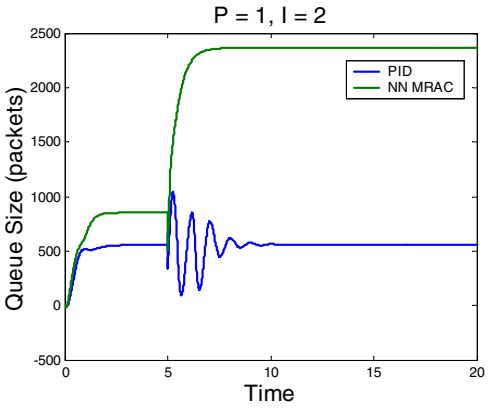

Figure 8: Simulation 1, 20 TCP flow drop

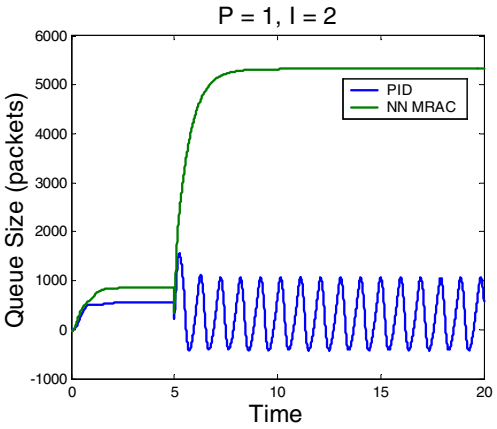

Figure 9: Simulation 1, 30 TCP flow drop out.

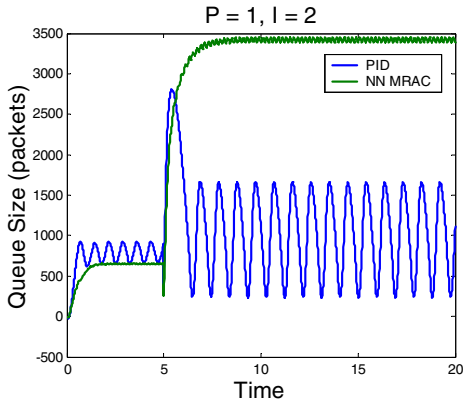

Figure 10: Simulation 1, q_{ref} = 1040 packets, 30 TCP drop out.

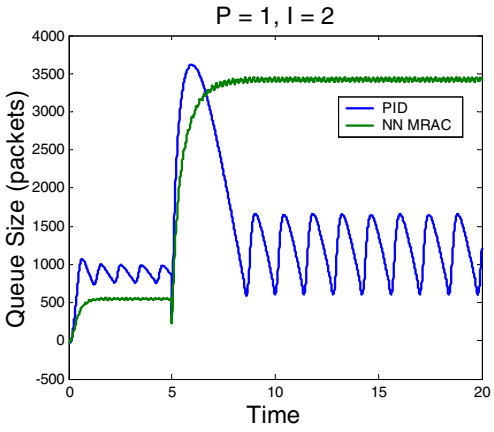

Figure 11: Simulation 1, q_{ref} = 1280 packets, 30 TCP drop out.

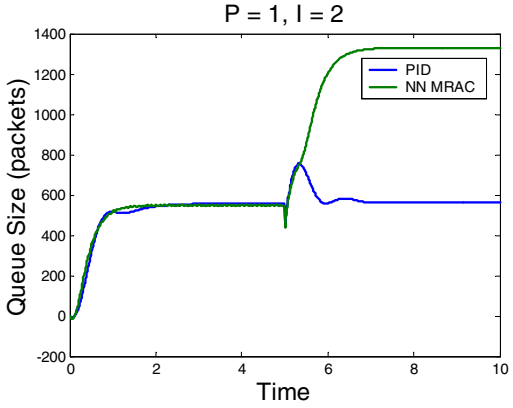

Figure 12: Simulation 2, q_{ref} = 560 packets, 10 TCP session drop out.

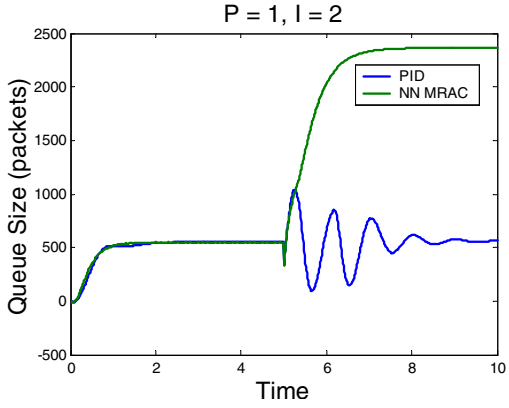

Figure 13: Simulation 2, q_{ref} = 560 packets, 20 TCP session drop out.

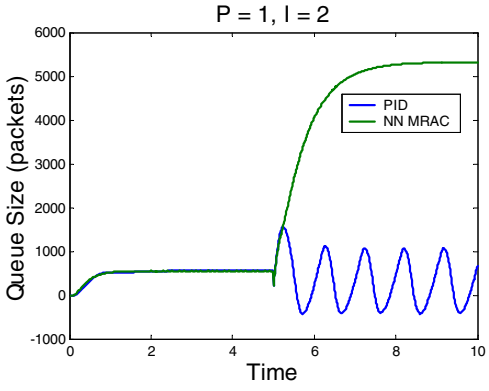

Figure 14: Simulation 2, q_{ref} = 560 packets, 30 TCP session drop out.

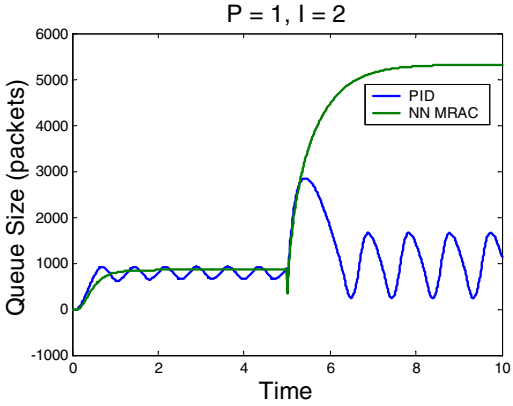

Figure 15: Simulation 2, q_{ref} = 800 packets, 30 TCP session drop out.

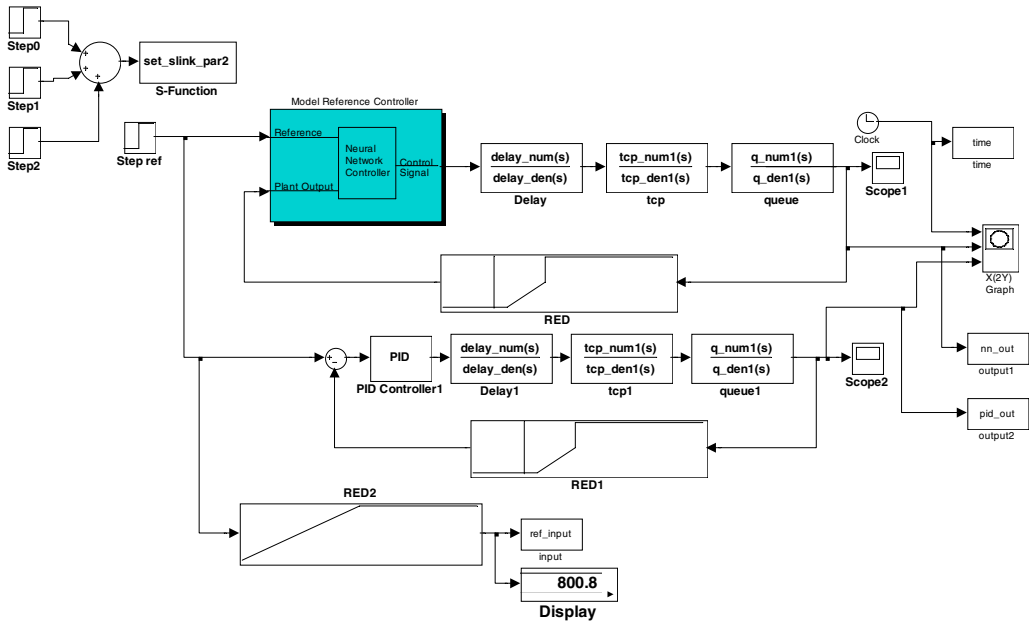

Figure 16: Simulink simulation block.

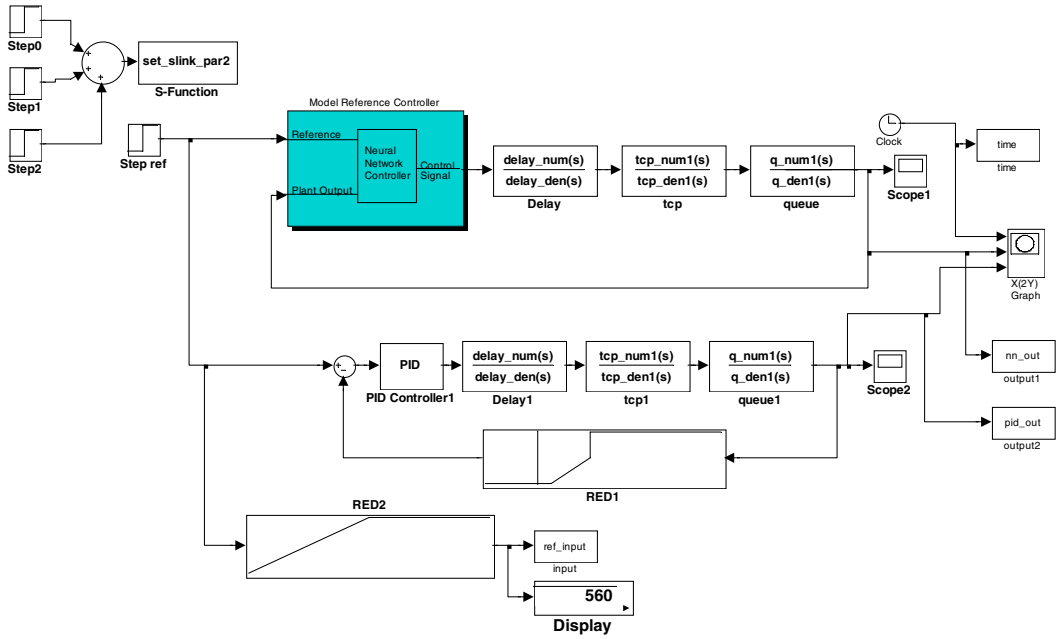

Figure 17: Neural network model reference and PI control structures.

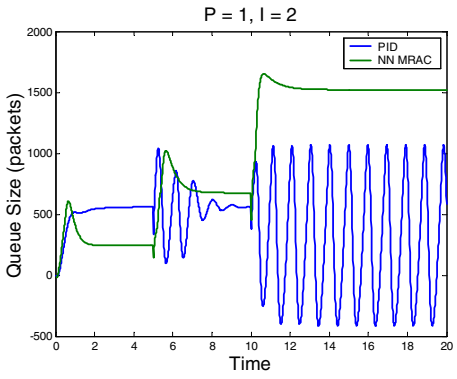

Figure 18: Simulation 3, q_{ref} = 560 packets, 5 session drop at 5 and 10 drop at 10.

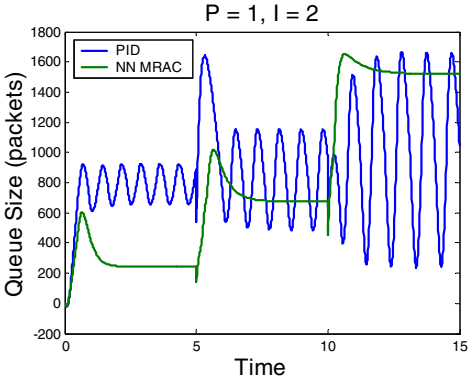

Figure 19: Simulation 3, q_{ref} = 800 packets, 5 session drop at 5 and 10 drop at 10.