IJCCCE Vol.14, No.3, 2014

Development of Model Predictive Controller for Congestion Control Problem †

Dr. Amjad J. Humaid [1], Dr. Hamid M. Hasan [2], and Dr. Firas A. Raheem[3]

[1,2,3] Control and Systems Eng. Dept., University of Technology

e-mail: Amjad.j.h@uotechnology.edu.iq, hamid2012net@gmail.com, dr.firas7010@yahoo.com

Received: 23/06/2014

Accepted:11/09/2014

Abstract— Nowadays congestion in computer networks is pointed out as an important and a challenging problem. TCP (Transmission Control Protocol) has the mechanism to avoid congestion in computer networks. TCP detects congestion by checking acknowledgements or time-out processing and adjusts TCP window sizes of senders. However, this control method shows low efficiency in communications because it is based on a mechanism that avoid congestion after congestion once appears in computer networks. TCP random early detection RED is another popular congestion control scheme. The fundamental idea behind this control algorithm randomly drops the incoming packets proportional to the average queuing length and to keep the queuing length to a minimum. To achieve high efficiency and high reliability of communications in computer networks, many control strategies based on advanced control theories have been introduced to tackle the congestion problem. Model Predictive Control (MPC) is the only practical control method that takes account of system constraints explicitly, and the only 'advanced control' method to have been adopted widely in industry. MPC is a model-based method which uses online optimization in real time to determine control signals. The solution to optimization problem is usually formulated with the help of a process model and measurements. At each control interval, an optimization algorithm attempts to determine the plant dynamics by computing a sequence of control input values satisfying the control specifications. In this work, a planning strategy based on MPC will be developed for congestion control problem. A "preset controllers" approach will be introduced for such application. The effectiveness of considered controller will assessed in terms of how well it could show good tracking performance, maximizing the utilization of the available bandwidth and to what extent it could cope with system uncertainties.

Keywords: AQM/TCP congestion, PID controller, Model Predictive Control

† *This paper has been presented in ECCCM-2 Conference* and accredited for publication according to IJCCCE rules.

1. Introduction

Recently computer networks as represented by Internet are becoming common widely and amount of data communicated in computer networks is increasing rapidly. The rapid increment of amount of data causes difficulty of smooth data transfers in computer networks and this difficulty is called as congestion. Now this congestion in computer networks is pointed out as important problems to be solved [1].

Transmission Control Protocol (TCP), which is a usual protocol to communicate each computer in Internet, has the mechanism to avoid congestion in computer networks. Most currently in-use networks are based on window flow control for the TCP. The size of the window length is used as a feedback mechanism to control the network's congestion problem. The packet losses are used as feedback signal for reducing the queuing window size in TCP. This congestion control protocol increases the window length gradually until a packet loss occurs at the gateway, indicating that the transmission bandwidth limits have been reached. At this point, if the transmission source keeps increasing its window length more packet losses will occur. Therefore, as soon as a packet loss is detected, the source host will reduce its window size to avoid network congestion. TCP detects congestion by checking acknowledgements or time-out processing and adjusts TCP window sizes of senders accordingly [1,2].

Figure (1) shows a simple sender–receiver connection passing through a bottleneck router. Under TCP, a sender probes the network's available bandwidth by linearly increasing its rate until data packets are lost. Upon packet loss, the receiver signals the sender to reduce its rate. Some drawbacks in this packet-dropping scheme include low inefficiency, flow synchronization and performance degradation due to excessive time-outs and restarts [2].

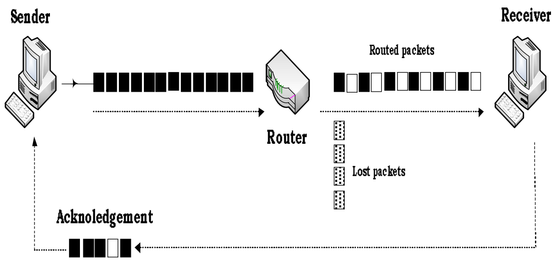

Fig.1. A schematic of a sender–receiver connection [2].

Motivated by these network inefficiencies, the Random Early Detection (RED) scheme was introduced in to allow the router to assist TCPs management of network performance. Rather than waiting for packet loss to occur, RED acts preemptively by measuring the router's queue length and throttling the sender's rate accordingly. Since TCP is an end-to-end protocol, RED achieves this feedback indirectly by randomly dropping/marking packets and routing them to the receiver. The receiver, in turn, completes the feedback by acknowledging the receipt of marked packets to the sender. Upon receipt of such acknowledgments, the sender increases or decreases its rate according to the TCP algorithm. The randomness in REDs packet-marking scheme is meant to eliminate flow-synchronization while queue-averaging was introduced to attenuate the effects of burst traffic on the feedback signal [2,3].

Since TCP RED is a feedback control problem, one can apply classical as well as modern control theories to adjust the size of the queuing window. The principal performance objectives for any modern controllers are [4]:

- To obtain a good tracking performance in terms of fast transient response with minimum peak overshoot and steady state error.
- Robustness, to maintain closed-loop performance in spite of plant uncertainties, N, R_o and C.
- Efficient queue utilization, to avoid overflow or emptiness of the queue buffer.

In the present work, the Model Predictive Controller is suggested and developed to achieve the above tasks.

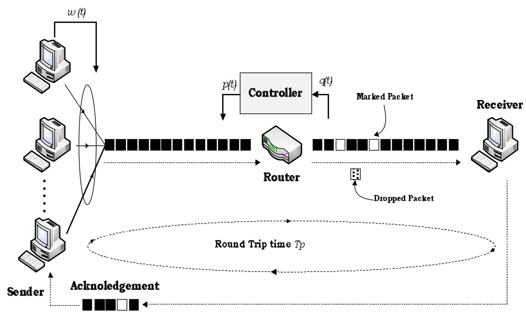

Fig.2. Structure of TCP network systems.

2. Model Predictive Control (MPC)

Model predictive control has had a major impact on industrial practice with thousands of applications worldwide. It has been reported that there were over 4,500 applications worldwide by the end of 1999, primarily in oil refineries and petrochemical plants. In these industries, MPC has become the method of choice for difficult multivariable control problems that include inequality constraints [5].

Model predictive control offers several important advantages:

- The process model captures the dynamic and static interactions between input, output, and disturbance variables,
- Constraints on inputs and outputs are considered in a systematic manner,
- The control calculations can be coordinated with the calculation of optimum set points, and
- Accurate model predictions can provide early warnings of potential problems. Clearly, the success of MPC (or any other model-based approach) depends on the accuracy of the process model. Inaccurate predictions can make matters worse, instead of better.

A block diagram of a model predictive control system is shown in Figure (3). A process model is used to predict the current values of the output variables. The differences between the actual and predicted outputs serve as the feedback signal to a *Prediction* block. The predictions are used in two types of MPC calculations that are performed at each sampling instant: set-point calculations and control calculations [5, 6].

Inequality constraints on the input and output variables, such as upper and lower limits, can be included in either type of calculation. One can easily note that the model acts in parallel with the process and the residual serves as a feedback signal. However, the coordination of the control and set-point calculations is a unique feature of MPC.

The set points for the control calculations, also called *targets*, are calculated from an economic optimization based on a steady-state model of the process, traditionally, a linear model. Typical optimization objectives include maximizing a profit function, minimizing a cost function, or maximizing a production rate. The optimum values of set points change frequently due to varying process conditions, especially changes in the inequality constraints. The constraint changes are due to variations in process conditions, equipment, and instrumentation, as well as economic data such as prices and costs. In MPC the set points are typically calculated each time the control calculations are performed.

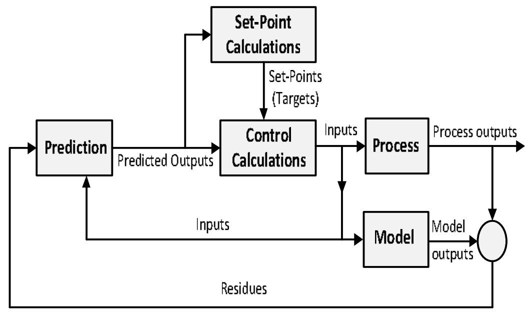

Fig. 3. Block diagram for model predictive control [5].

Applications typically involve two types of calculations [5,6]:

1. A steady-state optimization to determine the optimum set points for the control calculations, and

2. Control calculations to determine the MV changes that will drive the process to the set points.

The success of model-based control strategies such as MPC depends strongly on the availability of a reasonably accurate process model. Consequently, model development is the most critical step in applying MPC.

The MPC calculations are based on current measurements and predictions of the future values of the outputs. The objective of the MPC control calculations is to determine a sequence of *control moves* (that is, manipulated input changes) so that the predicted response moves to the set point in an optimal manner. The actual output y, predicted output and manipulated input $u$for SISO control are shown in Fig. (3). At the current sampling instant, denoted by k, the MPC strategy calculates a set of M values of the input $u(k+i-1)\ i=1,2,\ldots.,M.$ The set consists of the current input $u(k)$and $M-1$ future inputs. The input is held constant after the M control moves. The inputs are calculated so that a set of Ppredicted outputs $\hat{y}(k+i)$, $i=1,2,\ldots,P)$ reaches the set point in an optimal manner. The control calculations are based on optimizing an objective function. The number of predictions$P$is referred to as the *prediction horizon* while the number of control moves M is called the *control horizon* [5-7].

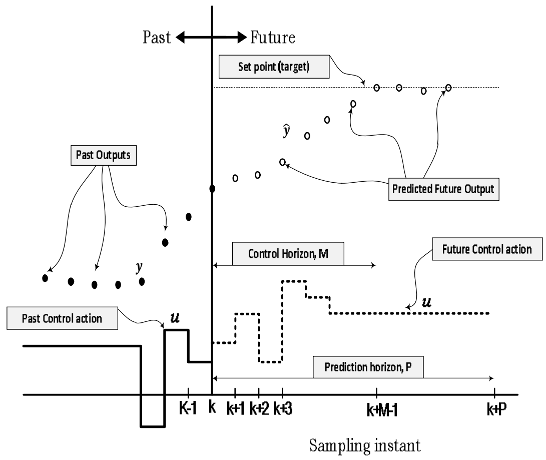

Fig.4. Basic concept for model predictive control.

3. System Dynamics

The mathematical model of the system is a prerequisite for most control system designs. The TCP network system is characterized by nonlinear, time-varying delayed model.

A dynamic model of TCP behavior was developed using fluid-flow and stochastic differential equation analysis. The model relates the average value of key network variables and is described by the following coupled, nonlinear differential equations as follows [1-4,8]:

$$\dot{w}(t)=\frac{1}{R(t)}-\frac{w(t)-w\big(t-R(t)\big)}{2\,R\big(t-R(t)\big)}\big(p(t)-R(t)\big) \qquad (1)$$

$$\dot{q}(t)=\begin{cases}\dfrac{N(t)}{R(t)}w(t)-C(t) & q(t)>0\\ 0 & else\end{cases} \qquad (2)$$

where $w(t)$ denotes the expected TCP sending window size (packets), $C(t)$ is the actual capacity, $p(t)$ is probability packet marking/drop, $q(t)$ denotes the expected queue length (packets) and $N(t)$ is number of TCP sessions. The round-trip time is given by

$$R(t) = \frac{q(t)}{c(t)} + T_p \qquad (3)$$

where is T_p propagation delay (secs). The queue length and window size satisfy $q(t) \in [0, q_{max}]$, and $w(t) \in [0, w_{max}]$ where q_{max} and w_{max} denote buffer capacity and maximum window size, respectively. The dropping probability $p(t)$satisfies $p(t) \in [0,1]$.

Many modern control strategies have been suggested for solving the TCP/AQM problem adopts linearized model of Eq.(1)-Eq.(3). Such linearization may give incorrect nominal operating points and AQM system variables should not be perturbed to a large extent; otherwise the system model may deviate from the actual model characteristics. To avoid this problem and to account for Model Predictive Controller, which requires a near-actual system model to give an efficient performance, it is preferable to express the system into a nonlinear discrete model [8,9]

$$w(k+1) = w(k) + \frac{T_s}{R(k)} - T_s \frac{w(k)\, w\big(k - R(k)\big)}{2\, R(k - R(k)} \big(p(k) - R(k)\big) \qquad (4)$$

$$q = \begin{cases} q(k) + T_s \dfrac{N(k)}{R(k)} w(k) - T_s c(k) & q(k) > 0 \\ q(k) & else \end{cases} \qquad (5)$$

Where T_s denotes the sampling period. In the considered AQM model, the router needs to sample the queue length in its packet buffer with the sampling period T_s. Equation (4) reflects variation of TSP window sizes and Eq. (5) reflects variation of queue lengths of the router.

Compared with the linearized AQM systems, the nonlinear model have no parameters involving equilibrium points, permits the variables to perturb larger from the operating points without loss of system characteristics. This would lead will lead to a more efficient control for the AQM network

Based on the dropping probability, the TCP window size, the round-trip time and the number of TCP sessions, one can calculate the throughput of the router as follows [9]:

$$q_{out} = \frac{N(k)}{R(k)} w(k)[1 - p(k)] \qquad (7)$$

Based on the above equation, one can establish the condition avoiding the congestion,

$$\frac{N(k)}{R(k)} w(k)[1 - p(k)] < C \qquad (8)$$

assuming that all TCP senders have the same average dropping probability, the average TCP window size and the average round-trip time.

The condition given by Eq.(8) tells that the dropping probability plays an important role in the avoidance of network congestion. The AQM program needs to adjust the $p(k)$ according to the arriving packets rate $N(k)\, w(k)/R(k)$. If we design a good control algorithm of dropping probability, then the network can be free from congestion.

4. Planning Configuration of MPC [9]

The types of plants we consider are those with

$$y(k+1) = f\big(x\big(x(k), u(k), d(k)\big) \qquad (9)$$

where $y(k)$ is the measured output and f is a generally unknown smooth function of the control input $u(k)$, measurable state $x(k)$ and $d(k)$ disturbance. Let the tracking error is given by:

$$e(k+1) = r(k+1) - y(k+1) \qquad (10)$$

Generally, the primary objective will be to make the tracking error as small as possible for all time, and in particular, we would like it to asymptotically approach zero so that the output follows the reference input.

It is, therefore, necessary to formulate a criterion to decide which plan is the best. In the present work, the cost function $J\left(u^i[k,N]\right)$ will be used to quantify the quality of each candidate plan $u^i[k,N]$ using f_m model. If the reference input $r(k)$ is assumed either known for all time or at least that at time k it is known up till time $k+N$. Generally, we want the cost function to qualify over the next N steps how well the objective is met. The cost function used in the present work would be of the form

$$J\left(u^i[k,N]\right) = w_1 \sum_{j=1}^{N} \left(r(k+j) - y_m^i(k,j)\right)^2$$

$$+w_2 \sum_{j=1}^{N} (q_{out} - C)^2 \; q_{out} < C \qquad (11)$$

where $w_1 > 0$ and $w_2 > 0$ are scaling factors that are used to weight the importance of achieving the tracking error closely (first term) or maximizing the queue utilization (second term). The condition imposed in Eq,(11) is used to avoid overflow of the queue buffer. This condition can be implemented by rejecting all trajectories or solutions violating the condition of non-congestion.

Let us consider the plan to be a sequence of possible control inputs, where the i^{th} plan of length N at time k is denoted by

$$u^i[k,N] = u^i(k,0), u^i(k,1), \ldots, u^i(k,N-1) \qquad (12)$$

Our objective is to develop a controller that is based on a planning strategy. To do this, a model and an optimization method are used to evaluate the equality of each plan. This will provide a ranking of the quality of the plans. After that we will choose the plan that is best (call it plan i^*), and let the control input at each time instant k be

$$u(k) = u^{i^*}(k,0) \qquad (13)$$

That is, at each time k we choose the best plan $u^{i^*}(k,N)$, then use the first input from the control sequence as the input to the plant. The process is repeated at each time instant.

The model used for planning could be continuous or discrete, and it could be linear and nonlinear. It may be deterministic, or it may contain an explicit representation of the uncertainty in the problem domain so that plans can be chosen taking the uncertainty into account.

In industrial practice and in "model-predictive control" method, linear models are often used for the plant. However, in the present work, linearized model has been avoided and the nonlinear model is utilized for prediction with the exceptions that the dependence of the time delay argument t−R on queue length q, is ignored and it is assumed to be fixed to $\mathrm{t - R_o}$.

The general nonlinear discrete time model is given by:-

$$y_m(j+1) = f_m\left(x_m(j), u(j)\right) \qquad (14)$$

with output $y_m(j)$, state $x_m(j)$, and input $u(j)$ for $j = 0, 1, \ldots, N-1$.

Let $y_m^i(k,j)$ denotes the j^{th} value generated at time k using the i^{th} plan

$u^i[k,N]$; similarly for $x_m(k,j)$. In order to predict the effects of plan i (project into future) at time k one should compute for $j = 0,1,2,\dots,N-1$,

$$y_m^i(k,j+1) = f_m\left(x_m(k,j), u^i(k,j)\right) \quad (5)$$

At time k to simulate ahead in time, for $j = 0$ one has to initialize with $x_m(k,0) = x(k)$. Then, generate $y_m(k,j+1)$, $j = 0,1,2,\dots,N-1$, using the model and generate values of $u^i(k,j)$, $j = 0,1,2,\dots,N-1$ for each i.

Suppose that there are $\mathcal{S}$ plan templates, which have the form of functions F_u^i

$$u^i(k,j) = F_u^i[x(k,j), r(k+1)]i = 1,\dots,\mathcal{S} \quad (16)$$

where $r(k+1)$ is assumed to be measured. Hence at each step we take these $\mathcal{S}$ plans and project them into the future. The best plan is selected to form the best control input u^{i^*}, where i^* is the best plan as measured by the cost function. For some practical applications the value of $\mathcal{S}$ need be bounded; otherwise instability problems may arise.

In the present, the plan templates are the terms of Proportional Derivative Integral (PID) controller. In fact, one can simply create a grid of ranges for the different gains of PID controller by considering all possible combinations of

$$\begin{aligned} K_p &= K_{p\_min}, \dots, K_{p\_max} \\ K_d &= K_{d\_min}, \dots, K_{d\_max} \\ K_i &= K_{i\_min}, \dots, K_{i\_max} \end{aligned} \quad (17)$$

If the size of K_p, K_d and K_i are N_p, N_d and N_i, respectively, then there are $N_p \times N_d \times N_i$ different plans (controllers) that are evaluated at each step.

5. Simulated Results

In order to illustrate the effectiveness of the Model Predictive Controller for AQM/TCP congestion problem, the network simulation will be presented taking into account the parameters listed in Table (1).

Table 1. System parameters

Symbol	Definition	Value
q_r	The desired queue length	200
T_p	Propagation delay	0.2 sec.
T_s	Sampling instant	0.0063 sec.
N	Number of TCP connections	60
C	Link Capacity	3750(packets/sec)
R_o	Initial round trip time	0.246 sec.
W_o	Initial window size	15 packets
P_o	Initial probability of packet mark/drop	0.08
q_o	Initial queue length	175 packets

Firstly, AQM/TP closed system is implemented with PID controller. The setting of the controller terms have been set based on try and error procedure. In fact, it has been shown that the system stability is much sensitive to parameter changes of PID controller. Moreover, these values are of small values which may reach to the order of 10^{-9} . This, of course, may be attributed to highly complexity of the system model. Out of many responses, the response shown in Figure (5) is selected as the best one. The corresponding PID controller settings are listed below:

$$K_p = 11 \times 10^{-5}$$
$$K_i = 10 \times 10^{-6}$$
$$K_d = 10^{-4}$$

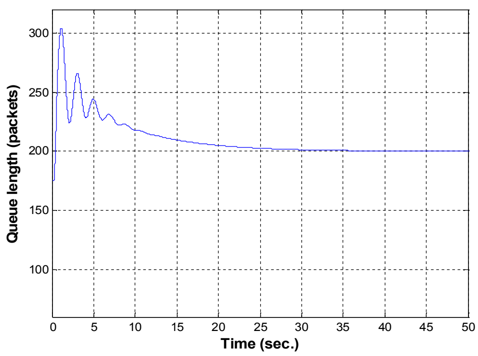

Figure. 5 Queue length response based on PID controller

Figure (6) shows the queue length response using MP controller. Based on the best tuned values of PID controller obtained from the previous scenario, the suggested ranges of K_p, K_i and K_d used in model predictive process are given by:

$$K_p = 11 \times 10^{-5}$$
$$K_i = (1:0.25:20) \times 10^{-7}$$
$$K_d = (1:0.25:20) \times 10^{-5}$$

It has been shown, via simulations, that no improvement of system response if the proportional gain is set a value over or lower than $K_p = 11 \times 10^{-5}$. Therefore, the above setting has been chosen and fixed over next simulations. For the present simulation, the horizon number is selected to be 10^4 iterations; i.e, it is equivalent to $(10^4 \times T_s = 63\, sec.)$.

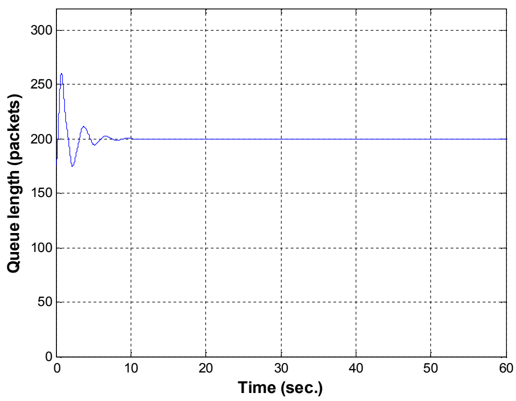

Figure 6. Queue length response based on MPC (Horizon Number=1000 iteration)

Figure (7) shows a cluster or group of suggested solutions and trajectory generated at specified time instant. The number of these trajectories every time instant is determined by the dimension of space spanned by the size of K_p, K_d and K_i. Since the value of K_p is kept constant over different simulations, then, there are $N_d \times N_i$ different trajectories at every time instant.

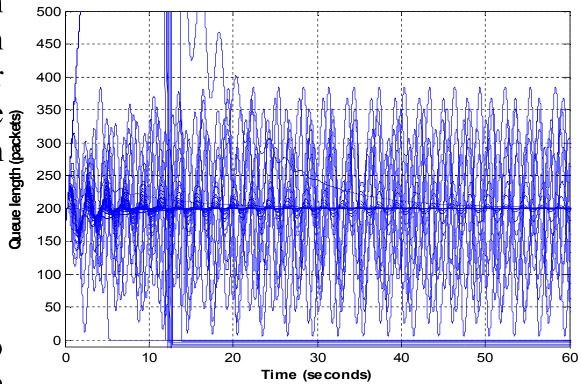

Figure 7. Generated solutions at every time instant

To examine the robustness of suggested controllers against parameter variations, two system parameters have been allowed to be changed; specifically, the propagation time and number of senders. Figures (8) and (9) show the queue dynamic response of both PID and MP controllers respectively, when the number of senders are set to values N =60, 80 and 100. It is evident from the figures that PID controller could not cope with such parameter variations, while the MP controller tries to compensate such variation and keep the response unchanged.

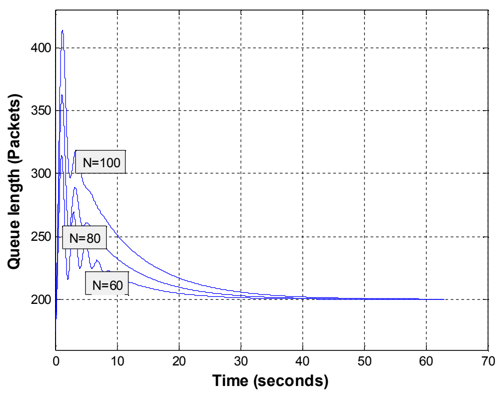

Figure 8.Queue length response based on PID controller with different sender numbers N

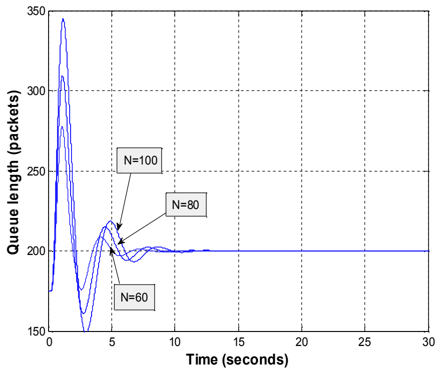

Figure 9. Queue length response based on MPC with different sender numbers N

Figure (10) and (11) show the dynamic response of queue length based on the two suggested controllers as the propagation time has been chanced in steps of T_p =0.1, 0.2 and 0.3 sec. It is clear from the figures that the PID controller shows an oscillatory response when the propagation time is set to value T_p =0.3. A further increase of such parameter could give unstable response without bound. However, PID controller could give satisfactory characteristic as its terms are again retuned. Hence, changing the system parameters, while fixing the PID controller terms, could degrade the system performance. On the other hand, queue length responses under supervision of MP controller are nearly unchanged.

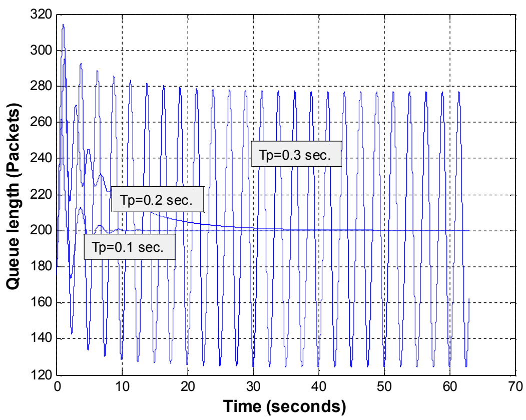

Figure 10. *Queue length response based on PID controller with different values of propagation time* T_p

Next, it is interesting to study the effect of changing the projection (horizon) length on the performance index. It is of importance to find the value of projection length which gives the minimum performance index. However, this value will be deeply recommended if its value were small; since larger values of projection length would take longer time for execution. In the present work, away from the execution time, the value of horizon length which minimizing the performance index, given by Eq.(11), is 90 as shown in Figure (10).

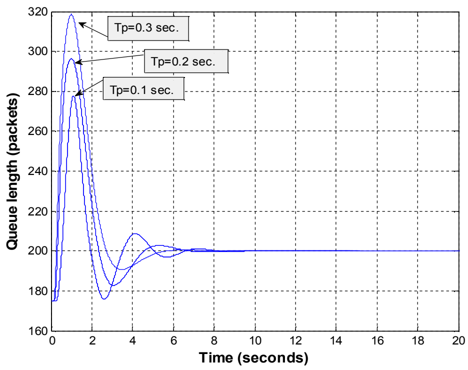

Figure 11. Queue length response based on MP controller with different values of propagation time T_p

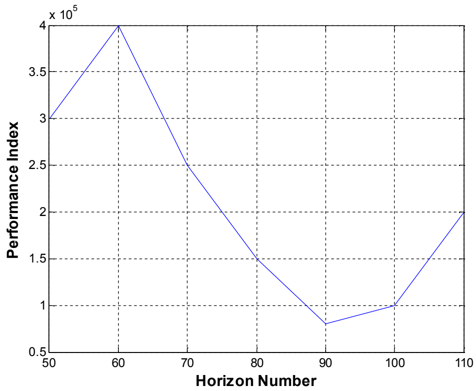

Figure 10. Performance index versus horizon number

6. Conclusions

It is worthy to emphasize that the work presented the dynamic model of AQM as it; i.e., including the nonlinearity and time varying complexities of system and without going to linearization process which may degrade the system characteristics and confine it to work around bounded operating region.

Two controllers have been presented PID and MP controllers. Based on the observations of simulated results, the following points can be highlighted;

- The AQM system based on MP controller shows better transient characteristics than the system with classical PID controller. One can easily see, from Fig.(5) and (6), that the settling time is 30 sec. for the response based on PID controller and it is about 10 sec. for the response based on MP controller.
- The variation in PID controller-based response is larger than that based on MP controller as the number of sender parameter N has been changed equally for both systems (with different controllers). On the other hand, if the same changes in round-trip parameter have been allowed to alter in the system, one may observe that the system with MP controller could keep the response unchanged, while a large variation in the system response based on PID controller has been shown and even at some case $(T_P = 0.3\ sec.)$the response shows instability characteristics. Therefore, one may conclude that the system based on MP controller is more robust than its counterpart.
- It has been shown that the performance index is related to projection (horizon) length in complex manner. However, Fig.(12) shows that the minimum value of performance index occurs at N=90.

References

[1] C. V. Hollot, Vishal Misra, Donald Towsley, and Weibo Gong, "Analysis and Design of Controllers for AQM Routers Supporting TCP Flows," IEEE Transactions on Automatic Control, Vol.47, No.6, June 2002.

[2] Wen-Jer Chang, Po-Hsun Chen, and Cheng-Ting Yang, "Robust Fuzzy Congestion Control of TCP/AQM Router via Perturbed Takagi-Sugeno Fuzzy Models," *International Journal of Fuzzy Systems, Vol. 15, No. 2, June 2013*.

[3] Sana Testouri, KarimSaadaoui, and Mohamed Benrejeb, "Evaluation of a PSO Approach for Optimum Design of First-Order Controllers for TCP/AQM Systems," World Academy of Science, Engineering and Technology 76, 2013.

[4] R. Vilanova, V. M. Alfar, "Robust 2-DoF PID control for Congestion control of TCP/IP Networks," Int. J. of Computers, Communications & Control, Vol. V, No. 5, pp. 968-975, 2010.

[5]Seborg. Et al, "Process Dynamics and Control," Springer-Verlag London, 2011

[6]J. M. Maciejowski, "Predictive Control with constraints" Prentice-Hall, 2000.

[7] Eduardo F. Camacho and Carlos Bordons, "Model Predictive Control," Springer-Verlag London, 1999.

[8] 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,” in *Proc. ACM/SIGCOMM,* 2000.

[9] Y. Xiao and M. –H. Lee, "Nonlinear Control of Active Queue Management for Multiple Bottleneck Network," IEICE TRANS. COMMUN. VOL. No. 1 JANUARY 2006.

[10] Kevin M. Passino, "Biomimicry for Optimization, Control, and Automation," Springer-Verlag, London, 2005.