PROCESS CONTROL SOLUTIONS FOR CONGESTION CONTROL IN COMPUTER NETWORKS

Teresa ALVAREZ*, and Jorge SANDOVAL

Department of Automatic Control, University of Valladolid, 47002 Valladolid, Spain
Corresponding Author's E-mail: tere@autom.uva.es

ABSTRACT: The use of Process Control ideas for the solution of congestion control problems is discussed here. Congestion appears in modern data networks when routers receive more data than they can process. If this congestion is not properly treated congestive collapse occurs, with the router providing very low throughput, as packages have to be resent multiple times. This problem is not dissimilar to some fluid control problems in Process Control, where flows have to be controlled to avoid overflows. Thus, it can be modelled using fluid models, so it is discussed in this paper how standard process control ideas can be adapted to deal with these congestion control problems. A case study involving a set of AQM-based routers under TCP protocol is carried out, using a non-interacting PID controller previously proposed for Process Control. Equivalent fluid models are provided and its control carried out. The results show how process control ideas make possible to avoid congestion when the controller is adequately tuned, despite inherent variations.

Keywords: Fluid Models, Process Control, PIDs, Congestion Control.

1 Introduction

Routers are main components of modern data networks: they are responsible of moving data between different networks, being responsible of ensuring that data gets where it needs to go. As they connect different networks, they are frequently affected by congestion, when they receive more data than their available capacity, so some data are discarded (see, for example, Azuma et al., 2006). This is a parallel problem to overflows in Process Control, aggravated by the fact that data not received at the receptor is resent by the emitter: If this congestion is not properly treated so called congestive collapse would appear (Jacobson, 1988), which is a parallel problem to Wind-up in Process Control.

It is discussed in this paper how standard process control ideas (such as those condensed in Aström and Hägglund, 2006; and O'Dwyer, 2009) can be adapted to correct this problem by carefully regulating the data that is discarded. This parallelism is based on the fact that data networks can be approximated by dynamic fluid models. As explained in Do Young (2005) fluid modelling captures the average rate of how each flow evolves and it is useful to know convergence conditions, nevertheless as networks have an inherent randomness sometimes a stochastic model can be usuful. But as starting point fluid modelling is a well-established and robust approach for understanding networks and tuning controllers.

The possibility of using algorithms derived from process control systems has already being suggested (see, for example, Hollot et al., 2002; Bolajraf et al., 2010; Alvarez and Martínez, 2013 and references therein). This will be illustrated for a specific case study involving a set of AQM-based routers (presented in Figure **2**). More precisely, under a standard TCP protocol it is shown that a PID-like controller (the so-called non-interacting controller type 5 proposed for process control problems by Hansen (1995), and further studied by O'Dwyer, 2009, pp. 370), is an adequate solution for this problem; a controller is then tuned and tested based on this structure. To develop the controller, first equivalent fluid models are discussed (following Bolajraf et al., 2010) and its control carried out using the selected PID-like controller.

In summary, this shows how process control ideas make possible to avoid congestion when the controller is adequately tuned, despite inherent variations in network parameters (Figure **3**).

2 Fluid Models in Congestion Control

Many different models have been proposed in the literature for congestion control problems (see, for example, Veres and Boda, 2000, the book by Srikant, 2012 and references therin). This paper solely concentrates on fluid models, that have the advantage of providing a set of nonlinear differential equations that represent faithfully the main dynamics of these process. From these nonlinear differential equations it is then possible to carry out analysis of the problem (see, for example, the stability analysis by Mazenc and Niculescu, 2003), and to obtain approximate linear models in transfer function form, that can be directly use to design and tune controllers using Process Control ideas (see, for example, Hollot et al., 2002 or Bolajraf et al., 2010).

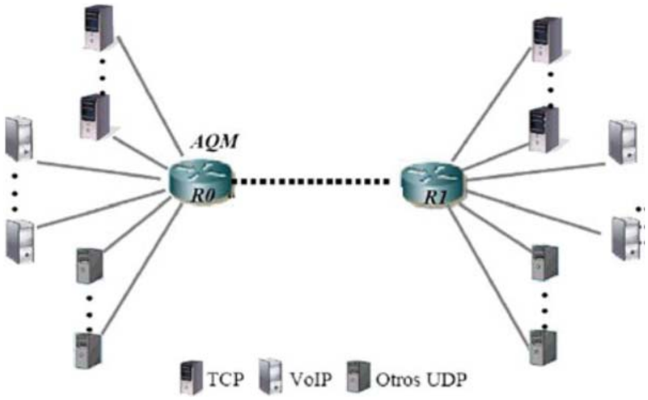

Figure 1: Dumbbell topology

2.1 *TCP/IP NewReno dynamic model*

Now, the TCP/IP NewReno network dynamics is presented. The dynamics of an AQM router are complex due to the number of variables that come into play: packet sources, protocols, etc. Nevertheless, it is possible to obtain a nonlinear model that represents the dynamics of the system (See Hollot et al., 2002) considering that the protocol used is TCP. The model relates to the average value of the network variables and is described by the following coupled, nonlinear differential equations:

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

$$\dot{q}(t) = \begin{cases} -C + \dfrac{N_{TCP}(t)}{R(t)}W(t), & q > 0 \\ \max\left\{0,\ \ -C + \dfrac{N_{TCP}(t)}{R(t)}W(t)\right\}, & q = 0 \end{cases}, \tag{1}$$

where

W: average TCP window size (packets),

q: average queue length (packets),

R: round-trip time = $q/C+T_p$ (secs),

C: link capacity (packets/sec),

T_p: propagation delay (secs),

N_{TCP}: load factor (number of TCP sessions) and

p: probability of packet mark.

As explained by Hollot et al. (2002), the first differential equation in (1) describes the TCP window control dynamic, and the second equation models the bottleneck queue length, as an accumulated difference between the packet arrival rate and the link capacity. The queue length and window size are positive, bounded quantities, i.e., $q \in [0,\bar{q}]$ and $W \in [0,\overline{W}]$, where $\bar{q}$ and $\overline{W}$ denote buffer capacity and maximum window size, respectively. In this formulation, the congestion window size $W(t)$ is increased by one every round-trip time if no congestion is detected, and is halved when congestion is detected.

2.2 *Linearized model*

Although an AQM router is a non-linear system, in order to analyse certain types of properties and design controllers, a linearized model is used. To linearize (1), it is assumed that the number of active TCP sessions and the link capacity are constant, i.e., $N_{TCP}(t)= N$ and $C(t)=C$.

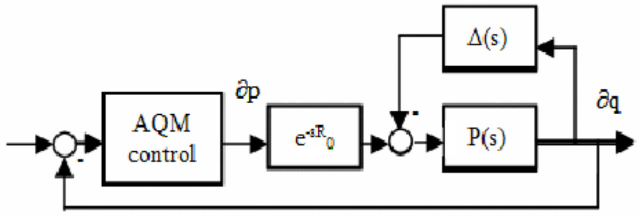

Figure 2: Block diagram of AQM as a feedback control system

To simplify the development of a model adequate for controller design, the dependence of the time delay argument $t-R$ on queue length q is ignored, so it is assumed to be $t-R_0$. This approximation is acceptable when the round-trip time is dominated by the propagation delay (see Chen et al., 2006), which occurs when the capacity C is large. R_0 should be chosen such that the above hypothesis can be ensured, so a value that works in the worst case scenario should be advisable. When the model is linearized, the same supposition is made. It cannot be denied that calculations are significantly simplified, but further research in the matter would be advisable. Local linearization of (1) around the operating point results in the following differential equations:

$$\left.\begin{aligned}\partial \dot{W}(t) &= -\frac{N}{R_0^2 C}\left(\partial W(t) - \partial W(t-R_0)\right)\\ &\quad -\frac{1}{R_0^2 C}\left(q(t) - \partial q(t-R_0)\right) - \frac{R_0^2 C}{2N^2}\partial p(t-R_0)\\ \partial \dot{q}(t) &= \frac{N}{R_0}\partial W(t) - \frac{1}{R_0}\partial q(t)\end{aligned}\right\}, \tag{2}$$

where $\partial \dot{W}(t) = W - W_0$, $\partial q = q - q_0$ and $\partial p = p - p_0$, represent the perturbed variables. The operating point for a desired equilibrium queue length q_0 is given by:

$$R_0 = \frac{q_0}{C} + T_p \,,\; W_0 = \frac{R_0 C}{N_{TCP}} \text{ and } p_0 = \frac{2}{W_0^2}\,. \tag{3}$$

The linearized model (2) can be rewritten by separating the low frequency ('nominal') behaviour *P(s)* of the window dynamic from the high frequency behaviour Δ(s) which is considered as parasitic.

$$\begin{aligned}P(s) &= \frac{C^2/(2N_{TCP})}{\left(s + (2N_{TCP})/(R_0^2 C)\right)\left(s + 1/R_0\right)},\\ \Delta(s) &= \frac{2{N_{TCP}}^2}{R_0 C^3}\left(1 - e^{-R_0 s}\right)\end{aligned} \tag{4}$$

Taking (4) as a starting point, Hollot et al. (2002) give a feedback control description of AQM (Fig. 1). The action implemented by an AQM control law is to mark packets with a discard probability *p(t)*, as a function of the measured queue length *q(t)*. The larger the queue, the greater the discard probability becomes.

3 Controller Design for the Case Study

It can be seen from (4) that the linearized model corresponds to a system with a delay and two real poles:

$$G_m(s) = \frac{K_m e^{-s\tau_m}}{(1 + sT_{m1})(1 + sT_{m2})} \tag{5}$$

where the parameters are given by:

$\tau_m = R_0$

$T_{m1} = \frac{R_0^2 C}{2N_{TCP}}$

$T_{m2} = \frac{1}{R_0}$

It has been proposed by O'Dwyer (2009) that for the class of systems described by (5) the most adequate PID-like controller is the one presented in Figure 3, which corresponds to the following structure:

$$U(s) = K_c\left(b + \frac{1}{T_i s}\right)E(s) - (c + T_d s)Y(s) \tag{6}$$

Some tuning rules are suggested in O'Dwyer (2009) for this controller based only on the value of the delay and time constant.

As the models in (1) and (4) correspond to approximations of the dynamics of the system, a full test was carried out using the simulation tool ns2 (see Issariyakul, and Hossain 2011 for a general description of this tool). ns2 provides discrete-event data network simulators, which are extensively used in data network research. Next section describes in detail these tests.

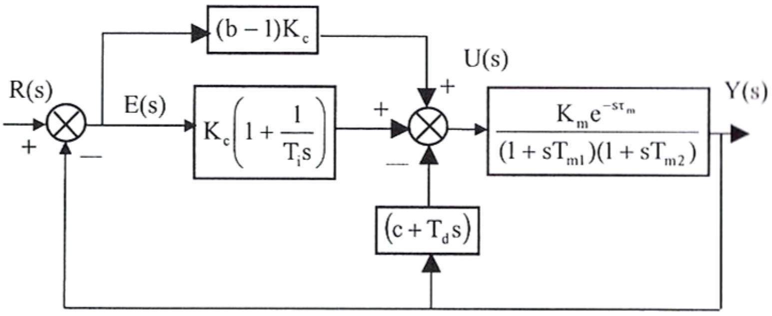

Figure 3: Non-interacting PID-like controller 5 [3]

4 Simulation results

This section describes a set of experiments that have been carried out to show the proposed approach.
First, a non-interacting PID-like controller that follows the structure presented in Figure **5** was tuned using the linear model in (4). The PID controller was tuned for the nominal case that corresponds to N_{TCP}=325 links, delay R_0=100ms; the link between the sources and Router1 has a capacity of C=20 Mb/s. The controller tuned for the nominal case had the following nominal parameters (in adequate units): b=0.143, c=-0.00078, K_c=-0.0015, T_i=0.9889, T_d=-0.00018. Once tuned, it was implemented using Simulink following the diagram block shown in Figure **3**. Then the corresponding discrete controller was obtained and compared to the continuous PID. These simulations gave some insight in the system performance and confirmed the adequacy of the controllers (results are not presented here due to lack of space).

Then, the non-linear simulation tool ns2 was introduced to simulate more realistic scenarios, based on the topology presented in Figure 4. For this detailed simulations standard blocks from ns2 libraries were used, that are known to faithfully reproduce real data networks. The PID proposed in Figure 3 was not available in those libraries, so it was coded. This makes possible to test the designed controller in several scenarios with different traffic parameters, changing number of sources and different delays, using exponential distributions for the generation of data by the sources. Some results are presented in Figures 5-8 and are now discussed in detail.

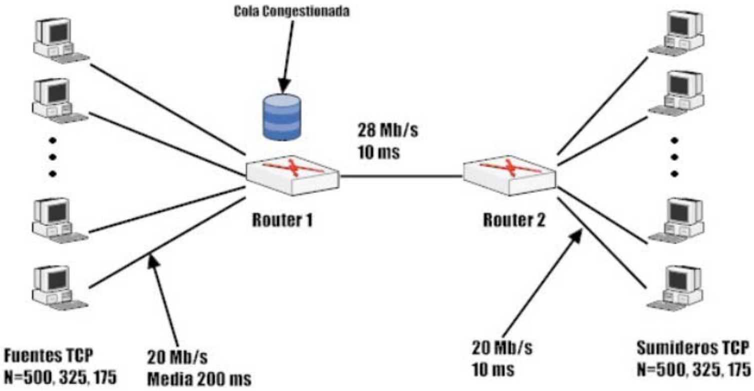

Figure 4: Dumbbell topology and network parameters

Figure 5 and Figure 6 show the queue and probability evolution. Our control variable is the probability of discarding a packet and the controlled variable the queue size in packets. It is shown in the same plot the simulation results for N=175, 325 and 500 sources when the reference is constant and equal to 200 packets. Even when the network conditions are different, the results are adequate with the designed controller.

Table 1 gives a summary of each of the three scenarios in terms of mean, standard deviation and two metrics widely used in communications. The QACD (Quadratic Average of Control Deviation) is a metric to measure how well is the controller working. It gives a measure of how much the real value is deviated from the desired value. The lower the value the better the result.

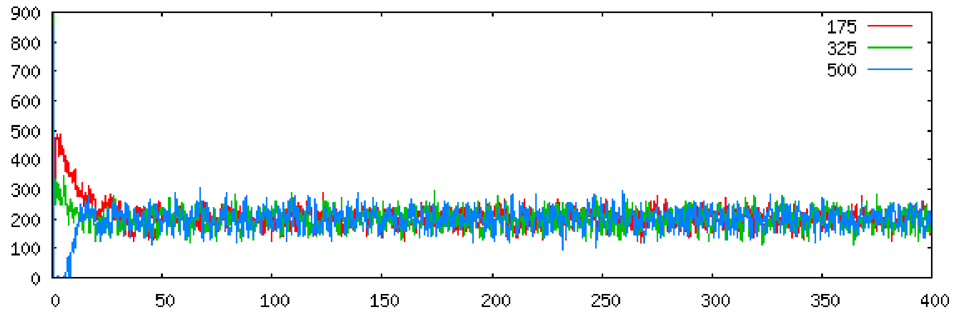

Figure 5: Evolution of the queue length with different number of sources

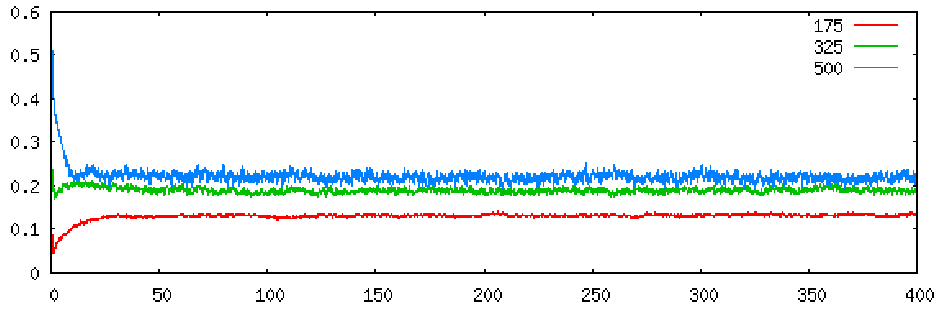

Figure 6: Evolution of the marking probability with different number of sources

Number of Sources *N*	**Queue:** Mean length	**Queue:** Standard deviation	**QACD:** mean	**QACD:** standard deviation
175	201.1	26.1	92.2	82.2
325	199.3	29.8	70.0	57.0
500	199.5	31.3	80.9	67.5

Table 1: Summary of Results for different number of sources with the proposed controller

The controller gives perfect results for the nominal case (*N*=325), and adequate results for the other two scenarios. It should be noted that from the observation of Figure 6 we can conclude that the lower marking probability belongs to the case when there are 175 sources.

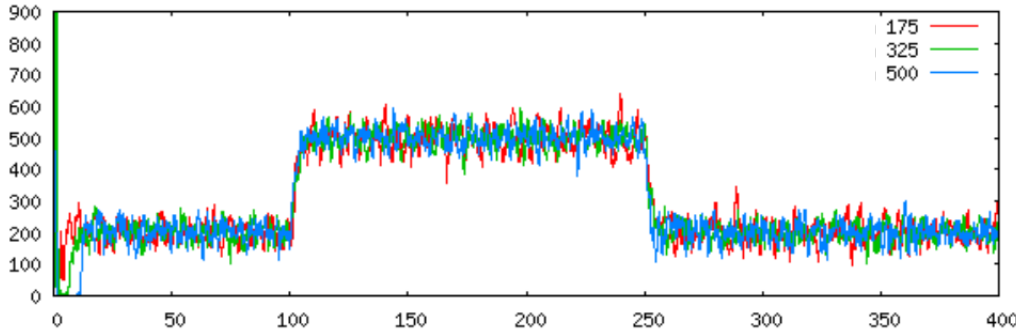

Figure 7: Evolution of the queue length when the reference changes

Another set of experiments consisted in changing the reference at various time points. Some results are presented in Figure 7 and Figure 8: good performance was obtained in the three different situations studied. Figure 7 shows the queue evolution and Figure 8 depicts the probability of marking packets. It can be seen that the controller is well tuned as the reference is correctly tracked.

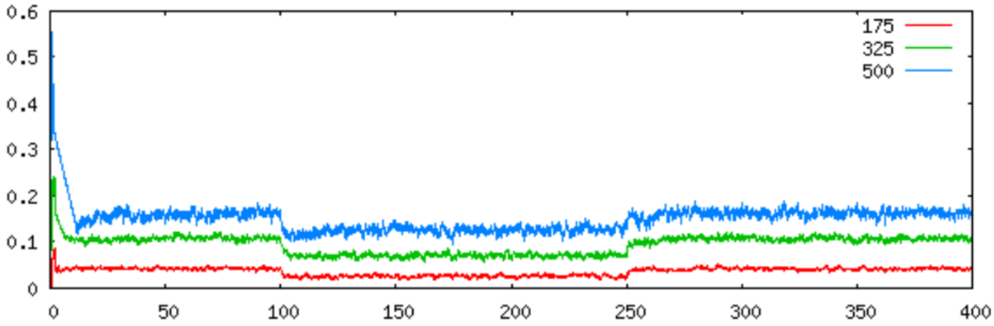

Figure 8: Evolution of the marking probability when the reference changes

5 Conclusions

It has being shown in this paper how process control ideas can be adopted for the congestion control problem in computer networks, by developing PID-like controllers developed from equivalent dynamic fluid models. This has being illustrated for a specific case study, based on the non-interacting controller type 5 used for process control. When this PID-like controller is adequately tuned, it makes possible to robustly avoid congestion, as shown by detailed simulations in expected operating conditions.

Acknowledgements

Funded by MiCInn DPI2014-54530-R and FEDER funds.

References

Aström K.J. and T. Hägglund (2006), Advanced PID control. ISA. NC.

Azuma, T., T. Fujita, T. and Fujita, M. Congestion control for TCP/AQM networks using State Predictive Control, *Electrical Engineering in Japan*, 156, 1491-1496 (2006)

Bolajraf, M., Tadeo, F., Alvarez, T., & Rami, M. A. (2010). State-feedback with memory for controlled positivity with application to congestion control. *IET Control Theory & Applications*, 4(10), 2041-2048.

Chen, J., F. Paganini, M.Y. Sanadidi, R. Wang and M. Gerla (2006). Fluid-flow analysis of TCP Westwood with RED. *Computer Networks*, 50, 132-1326.

Do Young Eun (2005). On the Limitation of Fluid-based Approach for Internet Congestion Control. ICCN, San Diego, USA.

Hansen, Peter D. *Self-tuning controller that extracts process model characteristics*. U.S. Patent No. 5,394,322. 28 Feb. 1995.

Hollot, C.V., V. Misra, D. Towsley, W. Gong (2002). Analysis and Design of Controllers for AQM Routers Supporting TCP flows. *IEEE Transactions on Automatic Control*, 47, 945-959.

Issariyakul, T., & Hossain, E. (2011). *Introduction to network simulator ns2*. Springer Science & Business Media.

Jacobson, V. (1988). *Congestion avoidance and control*. ACM SIGCOMM'88.

O'Dwyer, Aidan. *Handbook of PI and PID controller tuning rules*. London: Imperial College Press, 2009.

Srikant, R. (2012). *The mathematics of Internet congestion control*. Springer Science & Business Media.

Teresa Alvarez, Diego Martínez (2013), Handling the Congestion Control Problem of TCP/AQM Wireless Networks with PID Controllers, *IAENG Transactions on Engineering Technologies*, 365-379.

Veres, A., & Boda, M. (2000). The chaotic nature of TCP congestion control. In Proceedings of the Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2000), 3, 1715-1723).