Tạp chí Tin học và Điều khiển học, T.21, S.3 (2005), 201–207

THE POSSIBILITY OF USING TOKEN BUCKET FOR DROPPING/MARKING PACKETS AT CORE ROUTERS IN IP NETWORKS

LE HUU LAP[1], NGUYEN HONG SON[2]

[1]*PTIT, Hanoi, Vietnam*
[2]*Faculty of Information Technology II, PTIT Ho Chi Minh city branch, Vietnam*

Abstract. RED algorithm is the most popular AQM scheme currently in use in Diffserv Networks. Numerous variants of RED have been proposed. However, the symptoms of RED are quite pathological, e.g. , it is nearly impossible to select such RED parameters that the impact on network performance doesn't get worse. Dropping/marking packets according to probability seem to be a incorrect job. Moreover, RED is not convenient to implement AF classes, especially their subclasses. In this paper, We propose an approach that use token bucket as a mechanism for dropping/marking packets explicitly. Relying on that, the AF classes and their drop precedences are easily implemented. The token bucket undertakes to drop/mark packets and is controlled by a controller so that the queue isn't congested and obtains the highest utility level.

Tóm tắt. Thuật toán RED hiện nay đang là một cơ chế AQM phổ dụng nhất được sử dụng trong các mạng diff.serv. Rất nhiều phiên bản của RED đã được đề xuất, tuy nhiên, RED đã buộc lộ nhiều điểm bất hợp lý, ví dụ rất khó tìm được các tham số của RED sao cho chúng không ảnh hưởng xấu đến phẩm chất của mạng. Việc huỷ hay đánh dấu các gói theo xác suất được thực hiện trong thuật toán RED rường như không được chính xác. Hơn nữa, RED không thuận tiện trong việc tạo ra các AF class, đặc biệt là các subclass của AF. Bài báo này, chúng tôi đề xuất giải pháp dùng token bucket như một cơ cấu huỷ hay đánh dấu các gói một cách tường minh. Nhờ đó, các AF class và các mức huỷ gói của nó được hiện thực một cách dễ dàng. Token bucket chịu trách nhiệm huỷ hay đánh dấu các gói, và hoạt động của nó được điều khiển bởi một bộ điều khiển sao cho hàng đợi không bao giờ bị nghẽn và đạt được hiệu quả sử dụng cao nhất.

1. INTRODUCTION

Nowadays, the most challenging research area in IP network is to adapt to needs about QoS of various applications. The Internet Engineering Task Force (IETF) has proposed new network architectures such as Integrated Services (Intserv) [1], Differentiated Services (Diffserv) [2] in order to provide QoS to these new applications effectively. In these architectures, there is a common mechanism, which includes classifying, metering, conditioning, shaping, dropping, and marking. Developing router mechanisms to protect users from congestion traffic is very important inside of them. Buffer management techniques use a scheme in which the router drops packets probabilistically even when the buffer is not full, by detecting congestion early. Random Early Detection (RED) [3] is a popular buffer management strategy, which is implemented in some networks. RED routers compute the average queue size continuously. When the average queue size exceeds a preset threshold, the router drops or marks each packet with a certain probability that depends on the instantaneous queue size. The advantage of RED is no maintain per-flow information. However, the way of RED to do is not convenient for implementing these above architectures. Moreover, it is not totally effective for fair bandwidth allocation [4]. RED is considered as an active queue management (AQM) scheme and is theorized by [5]. Then, other papers proposed various advanced approaches for applying RED in diffserv networks. In [6], a PI-type AQM was proposed as a congestion controller at core routers. This AQM was shown to be able to maintain buffer level at reference set point in the face of dynamic network conditions. Token buckets were introduced in order to maintain source throughput at a target rate x. However, [7] showed that one cannot guarantee that resulting throughputs are equal to or greater than the token bucket rate. To overcome this inherent limitation, [8] proposed a feedback structure around a token bucket termed ARM. The purpose of ARM is to regulate the token bucket rate A_i such that $x_i \geq \underline{x_i}$ (if the network is sufficiently provisioned). In general, all these papers used a common fluid flow model that mixes the congestion mechanism in the TCP layer and AQM mechanism in IP layer into their analyses. This seem like no obey the layered principle of ISO on the data communication system.

Although the token bucket has ever used as a traffic shaper but recently many papers have proposed to use token bucket in different functions such as in [4] proposed a computationally simple mechanism based on token bucket policing to achieve almost equal bandwidth allocation for a set of competing flow. In [9] constructs a new dynamic model for the token bucket algorithm. This model is then augmented by adding a dynamic model for a multiplexor at an access node where the token bucket exercises a policing function. Based on the model they study such issues as QoS, traffic sizing and network dimensioning. Token bucket also acts as an important role in the hop-by-hop congestion control mechanism proposed in [10]. In this paper, we highlight the possibility of using token bucket for dropping or marking packets at core routers so that can replace the dropping/marking packets probabilistically. We use the token bucket as if it is an actually a traffic regulator that provides traffic into the outgoing buffer. To do that, it is necessary to govern the token bucket rate into the bucket according to the instantaneous free space of the buffer. A controller is right in the middle of the outgoing buffer and the token bucket senses the current free space of buffer and regulates the token bucket rate into the bucket, adequately. This help using maximum capacity of buffer without congestion. Another advantage is we can simply implement AF classes and their subclasses by altering the reference size of buffer in each class. In this paper, we have not focused on stability analysis of the system yet. We also don't refer more detail about the design for the controller. We address these issues in a later paper.

The rest of the paper is structured as follows. In section 2, we present a dynamic model for the system that consists of a token bucket connected to a bottleneck queue. Based on control theory we explain the possibility of controlling the token bucket rate so that the buffer runs maximum power without congestion. In section 3, we simply validate the proposed method by a computer simulation and give some guided lines for applying the method. The final section, we present our conclusions and mention certain outstanding issues for future work.

2. DYNAMIC MODEL OF TOKEN BUCKET-BOTTLENECK QUEUE

Obviously, the diffserv belong to the network layer in OSI Reference Model. According to the model, the function of each layer doesn't depend on other layers and can be developed separately. Therefore, we'll consider everything for the application model from IP layer only. We mean that the application model don't involve the congestion control mechanism of TCP or sliding window. In addition, it can ignore delay time because all components are at the core router.

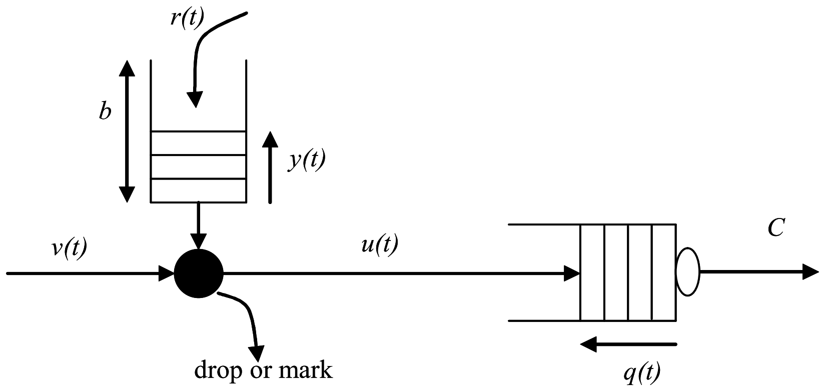

Figure 1. Token bucket _bottleneck queue model at core router

The origin of token bucket is a simple traffic shaping approach that permits burstiness [11]. However, nowadays, token bucket has different functions at different QoS provision. Here, the token bucket holds the role of congestive prevention. The token bucket works based on checking if amount of tokens contained in bucket is greater than or equal the amount of incoming packets, if was the token bucket forwards the packets and in the other case, the packets will be dropped/marked. The token bucket is located in front of output bottleneck queue as shown in figure 1. According to [11] each token bucket have two parameters concerned. The first parameter r is the rate of flow that pours tokens into the bucket. The second parameter b indicates the height of bucket or exactly, this is the maximum amount of tokens can be contained in that bucket. In other applications of token bucket, the parameter r is fixed but it is a varying parameter in my approach, denoted $r(t)$. $r(t)$ is governed by a control mechanism which base on current free space of the buffer. The number of tokens in bucket at the time t is $y(t), 0 \leqslant y(t) \leqslant b$. Packets arrive token bucket at rate of $v(t)$. The rate of packets forwarded from token buket to output buffer is called $u(t)$. Following fluid flow model mathematically represents this:

$$\dot{q}(t) = -1(q(t) > 0)C + u(t)$$

$$u(t) = 1(v(t) > 0).[r(t) + \frac{y(t)}{T}] \tag{1}$$

where, T is the continuous transmitted time. The outgoing link has capacity of C, a constant in diffserv networks.

From (1), we find that $u(t) \approx 1(v(t) > 0).r(t)$ if T is a considerable time. Dependent on time scale, we always have:

$$u(t)_{\max} = 1(v(t) > 0).[r(t) + y(t)] \tag{2}$$

with T gets the value of unit.

Considering $u(t) = u(t)_{\max}$ as a general case because it is the case filling buffer by the greatest speed. Therefore, the dynamic of the bottleneck queue is given by:

$$\dot{q}(t) = -1(q(t) > 0)C + 1(v(t) > 0).[r(t) + y(t)] \tag{3}$$

Reference to operating model as shown in Figure 1, we find that seem like no any impact on the system when $v(t) = 0$. In Addition, because of trying to use the size of buffer efficiently, We assume the buffer in the state of no empty. From that (3) can be rewritten as follow:

$$\dot{q}(t) = -C + r(t) + y(t) \tag{4}$$

Performing a Laplace transform on the differential equation (4):

$$q(s) = -\frac{C}{s^2} + \frac{r(s)}{s} + \frac{y(s)}{s} \tag{5}$$

The linear dynamics is illustrated in a block diagram form in Figure 2.

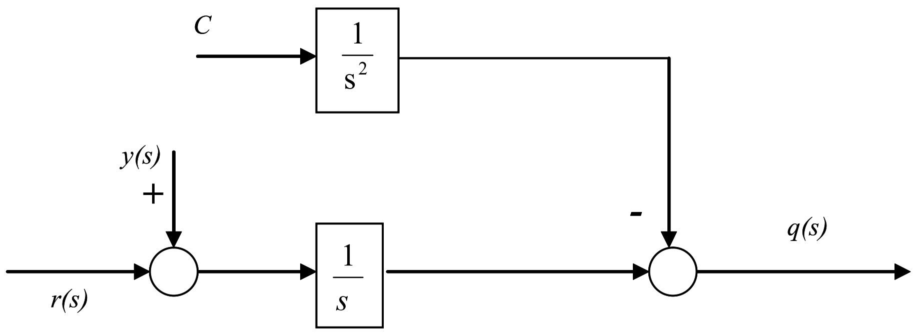

Figure 2. Block diagram of token bucket_ bottleneck queue at core router

The basic effect of the token bucket parameters $r(t)$ and b is that the amount of pakets sent $P(T)$ over any interval of time T obeys the rule:

$$P(T) \leqslant rT + b \tag{6}$$

Hence, $y(t) = b$ corresponds with the case of maximum amount of packets entering the buffer. Thus, we simply consider $y(t) = b$ as the worst case and replace $y(t)$ by b in calculations later.

The rate $r(t)$ must be controlled so that the buffer is never congested and obtain the highest utility level. We use a controller, which controls $r(s)$ according to free space part of the buffer. The dynamics of system is illustrated in Figure 3. This is a feedback control mechanism without delay because of every component at the same place.

Normally, the controller can be a P controller or a PI controller. Since the plant is equivalent to a SISO. The PI controller has a transfer function of the form:

$$G(s) = K_p.\Big(1 + \frac{1}{T_I s}\Big) \tag{7}$$

As is early mentioned, not to have the time delay in this case is a convenient thing for designing the controller. It can be completely designed by the normal way for the system without delay. A PI design involves choosing the value of the gain K_p and integrated constant T_I. There are several methods to determine these parameters such as the first Ziegler-Nichols method, the second Ziegler-Nichols method, the Chien-Hrones-Reswick method, the Kuhn method, etc. In this paper, we don't focusing how to design an optimal controller for the system, instead of that we simply determine these parameters by the second Ziegler-Nichols method, an experimental method, in next computer simulating section.

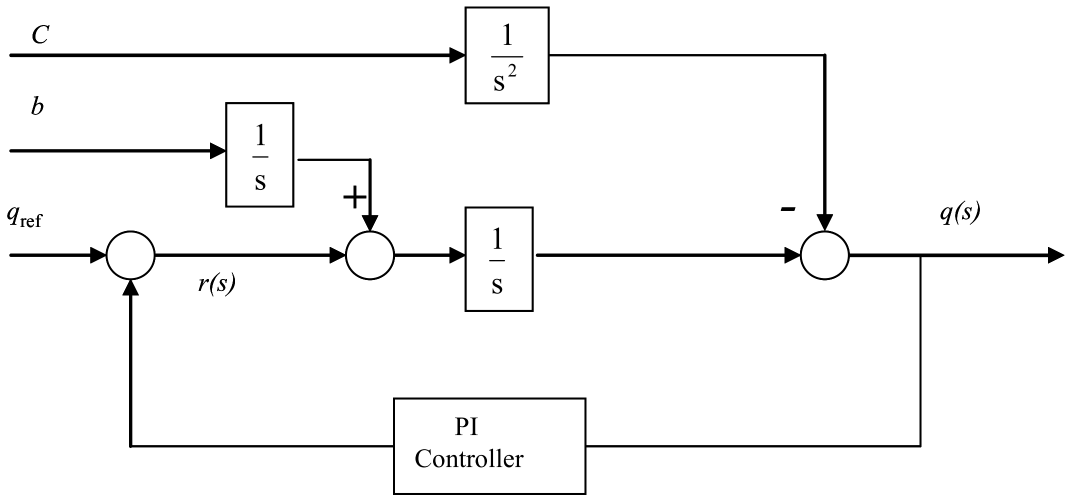

Figure 3. Block diagram of the control mechanism

3. SIMULATION

This section describes the results of simulation on computer. It shows the behavior of the bottleneck queue in a given system. In this simulation, assuming, the buffer has a size of 500 packets; the token bucket can contain up to 50 tokens; the rate of output link $C = 150$ packets per second. To this system, we found the acceptable control parameters: $K_p \approx 7$ and $T_I = 1$.

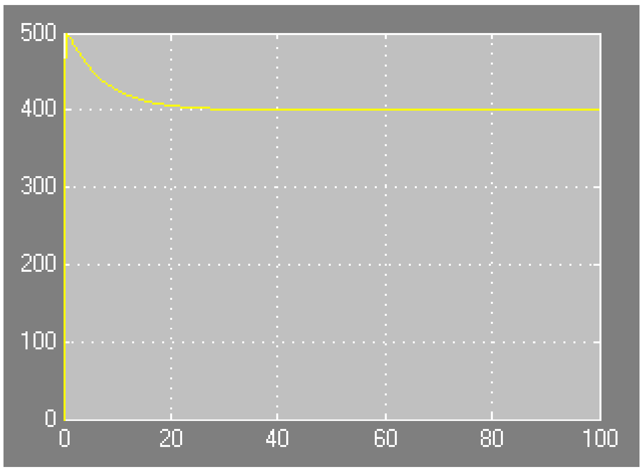

Figure 4. The behavior of $q(t)$ corresponding to $b = 50$

First of all, to observe the queue length $q(t)$ in figure 4, this show that the number of packets in queue starts at the highest level 500, then goes quickly down and stabilizes at 400. Next, we decrease the token bucket size to 20 and obtain the queue behavior as shown in Figure 5. In this figure, $q(t)$ goes quickly down and stabilizes at 360 approximately. This shows that the smaller b is, the smaller the capacity of the buffer is used. Meanwhile, $y(t)$ seems like smaller than b in working period of the mechanism thus we may not reach the highest utility level. However, this problem can be easily addressed by to augment the qref an adequate quantity. Again, q_{ref} is used and in this time, it takes the role of a tuner. The tuner can get the value greater than it owns so that the buffer is used as much as possible. Example, in this case of $b = 20$, can configure $q_{\text{ref}} = 530$ and get the utility level as shown in Figure 6.

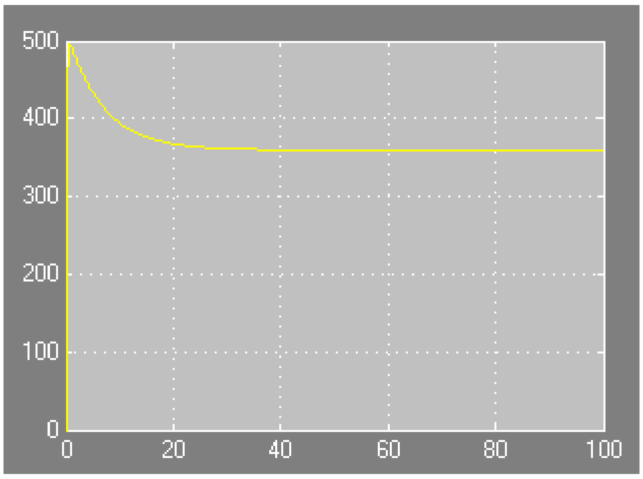

Figure 5. The behavior of $q(t)$ corresponding to $b = 20$

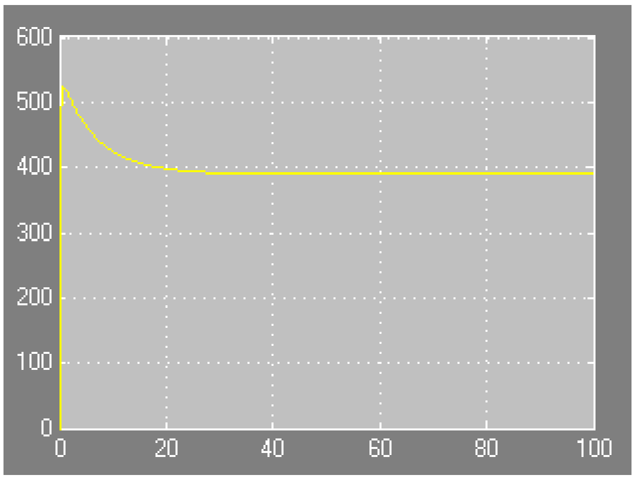

Figure 6. The improvement of using the buffer

4. CONCLUSION

The possibility of using token bucket at core routers in diffserv networks has been presented. This is a new way that makes it easy to implement the AF classes and their drop precedences. By pre-configuring q_{ref}, each active packet flow can be treated appropriately.

The mechanism not only guarantees about congestion control, but also gives a high utility level. The simulation on computer shows that the system is completely stable. The performance of the system depends on some factors such as the type of the controller, the performance of the controller, the sample time, the b parameter of token bucket, etc. Therefore, The performance of the system needs to be studied more details. Beside of dropping/marking function, the token bucket can be also applied to share bandwidth between different aggregates in diffserv networks. These issues will be discussed in next papers.

REFERENCES

[1] Braden, R., Clark, D., and Shenker, S. (1994) Integrated services in the internet architecture: an overwiew, RFC 1633.

[2] Blacke, S., Clark, D., Carlson, M., Davies, E, Wang, Z. and Weiss, W. (1998) An architecture for differentiated service, RFC 2475.

[3] S. Floy and V. Jacobson, Random Early Detection Gateways for Congestion Avoidance, *IEEE/ACM Transactions on Networking* vol. 1 (no. 4) (August, 1993) 397–413.

[4] J. Kidambi, D. Ghosal, B. Mukherjee, Dynamic Token Bucket: A fair bandwidth allocation algorithm for High-Speed Networks, *IEEE/ACM Transactions on Networking* vol. 1 (no. 4) (August, 1999) 24–29.

[5] C.V. Hollot, Vishal Misra, Don Towsley and Wei-Bo Gong, A Control Theoretic Analysis of RED, *Proceedings of IEEE/ INFORCOM,* 2001.

[6] C. V. Hollot, V. Misra, D. Towsley, and W. B. Gong, On designing improved controllers for aqm routers supporting tcp flows, *Procs. INFOCOM*, 2001.

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

[8] Y. Chait, C.V. Hollot, V. Misra, D. Towsley, H. Zhang,and C.S. Lui, Throughput Guarantees for TCP Flows Using Adaptive Two Color Marking and Multi-Level AQM, *Procs. INFOCOM* , pg. 837-844, 2002.

[9] N. U. Ahmed* , QUN. Wang and L. Orozco Barbosa, Systems approach to modeling the token bucket algorithm in computer networks, *Mathematical problems in Engineering* Vol. 8 (3) (2002) 265–279.

[10] G. Bastin, H. Mounier, Vincent Gufens, Hop-by-hop congestion control with token buckets in feedback: compartmetal analysis and experimental validation with UML, submitted to *IEEE Transactions on Automatic Control* (2003).

[11] S. Shenker, J. Wroclawski, "General Characterization Parameters for Integrated Service Network Elements,", RFC 2215, IETF, September 1997.

Received on August 24, 2005