TRANSFORMATION OF A TOKEN BUCKET TO A WORK CONSERVING LINK

Jakub DAUBNER, Katarína BACHRATÁ
Department of InfoComm Networks, University of Žilina, Žilina, Slovak Repubic,
e-mail: jdaubner@gmail.com, katarina.bachrata@fri.uniza.sk

ABSTRACT

Quality of Service (QoS) in IP multimedia converged networks which provides different serving levels of different traffic flows is a crucial issue. The token bucket mechanism is one of core QoS concept whose well understanding is required and improve modelling and implementation techniques required in such multimedia networks. This paper describes and defines two basic network elements - a work conserving link and a token bucket. We define two different types of a token bucket and we show that both of them can be transformed into some work conserving link. Transformation means that the work conserving link allows input unit to pass if and only if the token bucket would allow input unit to pass.

Keywords: *work conserving link, token bucket, quality of service, IP multimedia, VoIP*

1. INTRODUCTION

Technologies for transporting of a voice over IP networks has become one of the most important technologies leading the evolution of network convergence process [4]. To provide guarantees required for best services performance certain mechanisms needs to be implemented into network. The token bucket and (work conserving) link are one of fundamental QoS concepts which are used in such multimedia networks.

In this paper, we show that how are related QoS (*quality of service*) parameters of token bucket to QoS parameters of work conserving link.

Work conserving link (Fig.1) is a network element, which has a capacity, which represents how many input units can be processed by one time unit. Capacity can vary in time. If in one time unit arrives more input units than the capacity is, then the rest of the input units is stored in a input buffer, where they have to wait until capacity allow them to pass through the link. In real networks, buffer has finite size, however in theoretical models we can calculate also with infinite buffer size. If the buffer size is finite and there are more inputs then buffer size plus capacity, then the rest of inputs are thrown away, or we called them *lost inputs*, because they will never pass through the link.

Network elements like work conserving link occur in many real networks and usually we need to know how many input units are lost and how long do the input units have to wait in input buffer.

Another network element, a token bucket (Fig. 2), occur in many real networks, too, and usually we need to know the same parameters as within a work conserving link. Input units can pass through the token bucket if there are free tokens for them. Tokens represents permissions for input units. Tokens are generated in every time unit, and the number of generated tokens is called a capacity of a token bucket. Capacity can be a constant or it can vary in time. Generated tokens are stored in a token buffer, which has finite size. If there is no room in the token buffer, then the rest of generated tokens is thrown away (they are lost). One input unit always take one token when passing through a token bucket. If there is no free token for a input, then the input is stored in the input buffer, which is similar to a input buffer of a work conserving link.

In this paper we will show, that from the view of input units which are passing through the given network element, token bucket is the same network element like work conserving link. Formally, for a given token bucket we can construct a work conserving link, which allows input unit to pass if and only if the token bucket would allow input unit to pass.

We consider a discrete-time system with time indexed by $t = 0,1,2,\ldots$ We describe a discrete-time process of any measurable units by a sequence of variables $A \equiv \{A(t)|t = 0,1,2,\ldots\}$ or $A \equiv \{a(t)|t = 0,1,2,\ldots\}$, where $A(t)$ is cumulative number of units by time t and $a(t)$ is number of units at time t. Usually, we assume that there are no units at time 0, i.e. $A(0) = 0$ and $a(0) = 0$. We can compute $a(t)$ from $A(t)$ or vice versa, i.e. $A(t) = \sum_{i=0}^{t} a(t)$ and for $t > 0$ we have $a(t) = A(t) - A(t-1)$. Following the terminology in communication networks, the unit can be an *arrival*,*packet* or a *byte* or any other measurable unit.

An ideal *link* with capacity c (Fig.1) is a network element for which the number of departures from the link is bounded above by c packets per unit of time (per time slot). The buffer at the link is assumed to be infinite. An ideal link with capacity c is *work conserving* if the number of departures from the link is c packets pre unit of time when there are backlogged packets in buffer. We denote buffer length (*queue*) process by Q and departure process by B. To be precise, let define *work conserving link* as follows.

Definition 1. *Work conserving link is set of four processes A,C,Q,B. We say that $a(t)$ is the number of arrivals at time t, $c(t)$ is the capacity at time t, $q(t)$ is the queue length at time t and $b(t)$ is the number of departures at time t. Queue process must satisfy Lindley equation for $t \geq 0$*

$$q(t+1) = (q(t) + a(t+1) - c(t+1))^{+}, \tag{1}$$

where $x^{+} = \max\{x,0\}$. Departure process B must satisfy

$$B(t) = q(0) + A(t) - q(t). \tag{2}$$

Work conserving link is one of the most basic network elements. Therefore, there are known many theoretical and

practical results for this element, i.e. theory of effective bandwidth. In this paper we will show that another network element (*token bucket*) can be modelled by work conserving link.

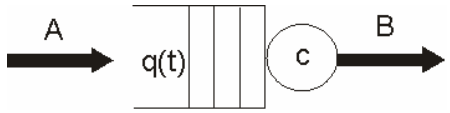

Fig. 1 Work conserving link

2. VARIOUS TYPES OF TOKEN BUCKET

As shown in Fig. 2, a token bucket consists of two buffers: a packet buffer and a token buffer. The size of the packet buffer is assumed to be infinite and the size of token bucket is h. A packet that arrives at a token bucket and finds a token in the token buffer will take the token and leave the token bucket immediately. Otherwise, it will be stored in the packet buffer until another token is generated. Tokens are generated at rate c, i.e. c tokens are generated per unit of time. A token will be added to the token buffer if the token buffer is not full.

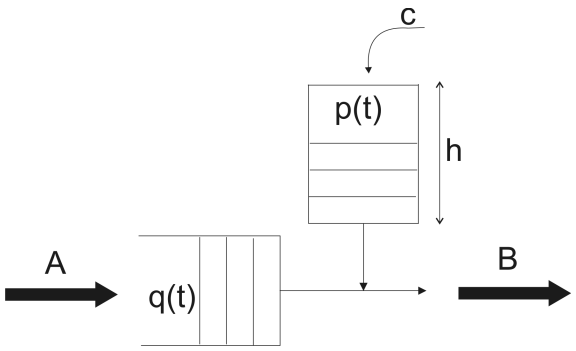

Fig. 2 Token bucket

Let consider that in one time slot is generated 1 token and 1 packet arrives. If the token is generated before the packet arrives, then it has to stored in token buffer and after the packet arrives they both will leave the token bucket. But, if the token buffer is full, then the token cannot be stored and it is thrown away. However, if the token is generated after packet arrives, then they leave token bucket immediately and the token does not have to be stored in the token buffer.

Therefore, we define two basic types of a token bucket. First one is more typical and it assumes that all packets arrive before all tokens in one time slot. Second one assumes that all tokens arrive before all packets in one time slot.

Definition 2 (Token bucket)**.** *Token bucket is set of five processes A,C,Q,P,B with constant h. We say that $a(t)$ is the number of arrivals at time t, $c(t)$ is the number of generated tokens at time t, $q(t)$ is the arrivals queue length at time t, $p(t)$ is the token queue length at time t, $b(t)$ is the number of departures at time t and h is token buffer size. Processes must satisfy following equations for all $t \geq 0$*

$$q(t+1) = (q(t)+a(t+1)-p(t)-c(t+1))^+ \quad (3)$$

$$p(t+1) = \min\{(p(t)+c(t+1)-q(t)-a(t+1))^+, h\} \quad (4)$$

$$B(t) = q(0)+A(t)-q(t). \quad (5)$$

Definition 3 (Token bucket*)**.** *Token bucket* is set of five processes A^*,C^*,Q^*,P^*,B^* with constant h^*. We say that $a^*(t)$ is the number of arrivals at time t, $c^*(t)$ is the number of generated tokens at time t, $q^*(t)$ is the arrivals queue length at time t, $p^*(t)$ is the token queue length at time t, $b^*(t)$ is the number of departures at time t and h^* is the token buffer size. Processes must satisfy following equations for all $t \geq 0$*

$$q^*(t+1) = (\max\{q^*(t)-p^*(t)-c^*(t+1),-h^*\} + a^*(t+1))^+ \quad (6)$$

$$p^*(t+1) = (\min\{-q^*(t)+p^*(t)+c^*(t+1),h^*\} - a^*(t+1))^+ \quad (7)$$

$$B^*(t) = q^*(0)+A^*(t)-q^*(t). \quad (8)$$

3. TRANSFORMATIONS

Next, we will show how can we model both types of token bucket by work conserving link. First transformation (token bucket) is well-known, see for example [1], [6, 8]. Second transformation is analogical to first one, however, it is a new result.

3.1. Token bucket

Note that in token bucket cannot be packets in packet buffer and tokens in token buffer at the same time, because packets should already leave token bucket if there are tokens. It means that for all $t \geq 0$ one of the values $q(t)$ or $p(t)$ must be zero. So, let look at the new process/variable

$$z(t) = q(t)-p(t). \quad (9)$$

In this case we look at tokens as if they were *negative* packets. The important fact is that we can calculate $q(t)$ and $p(t)$ from $z(t)$

- $z(t) > 0$
  If the difference of $q(t)$ and $p(t)$ is positive, then $p(t)$ must be zero and so we have $q(t) = z(t)$ and $p(t) = 0$.
- $z(t) = 0$
  If the difference of $q(t)$ and $p(t)$ is zero, then both values must be same, thus, both must be zeros, so we have $q(t) = 0$ and $p(t) = 0$.
- $z(t) < 0$
  If the difference of $q(t)$ and $p(t)$ is negative, then $q(t)$ must be zero, so we have $q(t) = 0$ and $p(t) = -z(t)$.

To abbreviate the notation, we can write

$$q(t) = (z(t))^+ \quad (10)$$

$$p(t) = (-z(t))^+ \quad (11)$$

We see in Fig. 3 what happens, if we concatenate packet buffer and token buffer in one buffer $z(t)$. From definition of $z(t)$ we have also following recurrent equation for all $t \geq 0$

$$z(t+1) = \max\{z(t) + a(t+1) - c(t+1), -h\}. \quad (12)$$

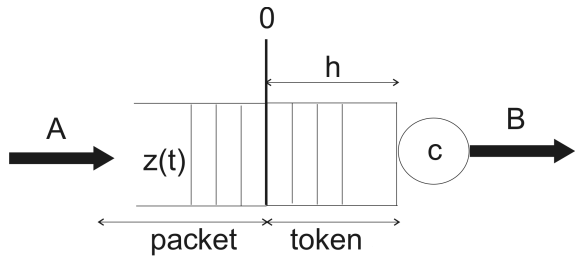

Fig. 3 Token bucket's modification for $z(t)$

We see that we can obtain Lindley equation (1) from (12) by adding h to right side. So we let $y(t) = z(t) + h$ and then we obtain

$$\begin{aligned} y(t+1) &= z(t+1) + h \\ y(t+1) &= \max\{z(t) + a(t+1) - c(t+1), -h\} + h \\ y(t+1) &= \max\{y(t) - h + a(t+1) - c(t+1), -h\} + h \\ y(t+1) &= \max\{y(t) + a(t) - c(t+1), 0\} \\ y(t+1) &= (y(t) + a(t+1) - c(t+1))^+. \end{aligned}$$

We can see in Fig. 4 what happens, if we add h to $z(t)$. Still, important fact is that we can calculate $q(t)$ and $p(t)$ from $z(t)$

$$q(t) = (y(t) - h)^+ \quad (13)$$

$$p(t) = (h - y(t))^+. \quad (14)$$

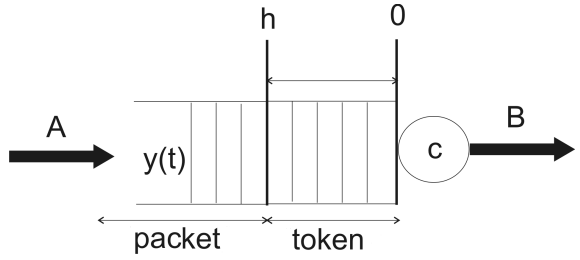

Fig. 4 Token bucket's modification for $y(t)$

Theorem 1. *Let (A, C, Q, P, B, h) be a token bucket and let (A_L, C_L, Q_L, B_L) be a work conserving link. If $A = A_L$, $C = C_L$, $p(0) = h$ and $q(0) = q_L(0) = 0$, then for all $t \geq 0$*

$$\begin{aligned} q(t) &= (q_L(t) - h)^+ \\ p(t) &= (h - q_L(t))^+ \\ B(t) &= B_L(t) + \min\{q_L(t), h\}. \end{aligned}$$

Proof. We have already proved first two equations (see (13) and (14)). So, next we prove only equation for token bucket's output process. We know that

$$\begin{aligned} B_L(t) &= A(t) - q_L(t) + q_L(0) \\ B(t) &= A(t) - q(t) + q(0), \end{aligned}$$

and if we use $q_L(0) = q(0) = 0$ then we get

$$\begin{aligned} B(t) &= B_L(t) - q(t) + q_L(t) \\ B(t) &= B_L(t) - (q_L(t) - h)^+ + q_L(t) \\ B(t) &= B_L(t) - \max\{q_L(t) - h, 0\} + q_L(t) \\ B(t) &= B_L(t) - \max\{-h, -q_L(t)\} \\ B(t) &= B_L(t) + \min\{q_L(t), h\}. \end{aligned}$$

□

For example, if we calculate that $\Pr(q_L(t) < x) \asymp e^{-\theta x}$ (i.e. using theory of effective bandwidth), then we can write also $\Pr(q(t) < x - h) \asymp e^{-\theta x}$.

3.2. Token bucket*

Analogically, we can prove transformation theorem for token bucket*.

Theorem 2. *Let $(A^*, C^*, Q^*, P^*, B^*, h^*)$ be a token bucket* and let (A_L, C_L, Q_L, B_L) be a work conserving link. If $A^* = A_L$, $(\forall t \geq 0) c^*(t+1) = c_L(t)$, $c^*(0) = 0$, $p^*(0) = h^*$ and $q^*(0) = q_L(0) = 0$, then for all $t \geq 0$*

$$\begin{aligned} q^*(t+1) &= (q_L(t) + a(t+1) - h^*)^+ \\ p^*(t+1) &= (h - q_L(t) - a(t+1))^+ \\ B^*(t+1) &= B_L(t+1) - (q_L(t) + a(t+1) - h^*)^+ \\ &\quad + q_L(t+1) \end{aligned}$$

Proof. Let denote $z^*(t) = q^*(t) - p^*(t)$ for all $t \geq 0$. Then we have for all $t \geq 0$

$$\begin{aligned} z^*(t+1) &= q^*(t+1) - p^*(t+1) \\ &= (\max\{q^*(t) - p^*(t) - c^*(t+1), -h^*\} \\ &\quad + a^*(t+1))^+ \\ &\quad - (\min\{-q^*(t) + p^*(t) + c^*(t+1), h^*\} \\ &\quad - a^*(t+1))^+ \\ &= (\max\{z^*(t) - c^*(t+1), -h^*\} + a^*(t+1))^+ \\ &\quad - (\min\{-z^*(t) + c^*(t+1), h^*\} - a^*(t+1))^+ \\ &= \max\{z^*(t) - c^*(t+1), -h^*\} + a^*(t+1) \end{aligned}$$

and we have also $z^*(0) = q^*(0) - p^*(0) = -h$.

Let denote $y^*(t) = z^*(t) + h^*$ for all $t \geq 0$. Then we have for all $t \geq 0$

$$\begin{aligned} y^*(t+1) &= \max\{z^*(t) - c^*(t+1), -h^*\} + a^*(t+1) + h^* \\ &= \max\{z^*(t) + h^* - c^*(t+1), 0\} + a^*(t+1) \\ &= (y^*(t) - c^*(t+1))^+ + a^*(t+1) \end{aligned}$$

and we have also $y^*(0) = z^*(0) + h = 0$.

Next, we use mathematical induction to prove that for all $t \geq 0$

$$y^*(t+1) = q_L(t) + a(t+1) \quad (15)$$

- $t = 0$

$$\begin{aligned} y^*(1) &= (y^*(0) - c^*(1))^+ + a^*(1) = 0 + a(1) \\ &= q_L(0) + a(1) \end{aligned}$$

- induction step

$$\begin{aligned} y^*(t+2) &= (y^*(t+1)-c^*(t+2))^+ + a^*(t+2) \\ y^*(t+2) &= (y^*(t+1)-c(t+1))^+ + a(t+2) \\ y^*(t+2) &= (q_L(t)+a(t+1)-c(t+1))^+ \\ &\quad +a(t+1) \\ y^*(t+2) &= q_L(t+1)+a(t+1). \end{aligned}$$

If we use

$$q^*(t+1) = (y^*(t+1)-h)^+ \quad (16)$$
$$p^*(t+1) = (h-y^*(t+1))^+, \quad (17)$$

together with (15) then we get desired results for $q^*(t+1)$ and $p^*(t+1)$.

Finally, we calculate $B^*(t)$. We know that

$$\begin{aligned} B_L(t+1) &= A(t+1)-q_L(t+1)+q_L(0) \\ B^*(t+1) &= A^*(t+1)-q^*(t+1)+q^*(0), \end{aligned}$$

and if we use $q_L(0) = q^*(0) = 0$ and $A^*(t) = A(t)$ then we get

$$\begin{aligned} B^*(t+1) &= B_L(t+1)-q^*(t+1)+q_L(t+1) \\ B^*(t+1) &= B_L(t+1)-(q_L(t)+a(t+1)-h^*)^+ \\ &\quad +q_L(t+1). \end{aligned}$$

If this result for $B^*(t+1)$ is hard to calculate, then we can use following simple inequalities

$$B^*(t+1) \leq B_L(t+1)+h^*. \quad (18)$$
$$B^*(t+1) \geq B_L(t+1). \quad (19)$$

□

Theorem 2 indeed says that token bucket* can be modelled by work conserving link, which has same arrival process and shifted capacity (token rate) process by one time slot.

Moreover, if all processes are random variables and they all converges, i.e. for all k

$$\begin{aligned} \Pr(\lim_{t\to\infty} a^*(t)=k) &= \Pr(\lim_{t\to\infty} a_L(t)=k) \\ &= \Pr(a^*(\infty)=k) \\ \Pr(\lim_{t\to\infty} c^*(t+1)=k) &= \Pr(\lim_{t\to\infty} c_L(t)=k) \\ &= \Pr(c^*(\infty)=k) \\ \Pr(\lim_{t\to\infty} q_L(t)=k) &= \Pr(q_L(\infty)=k), \end{aligned}$$

then as a direct consequence of theorem 2 we have also

$$\begin{aligned} \Pr(\lim_{t\to\infty} q^*(t)=k) &= \Pr((q_L(\infty)+a(\infty)-h^*)^+=k) \\ &= \Pr(q^*(\infty)=k) \end{aligned} \quad (20)$$

and the probability density function of random variable $q^*(\infty)$ can be calculated for all $k>0$ from probability density functions of random variables $q_L(\infty)$ and $a(\infty)$ as follows

$$\Pr(q^*(\infty)=k) = \sum_{i=1}^{k+h^*} \Pr(q_L(\infty)=i)\cdot\Pr(a(\infty)=h^*+k-i). \quad (21)$$

And if $a(t)$ is non-negative random variable, then we have also for all $k \geq 0$

$$\Pr(q^*(\infty)>k) \geq \Pr(q_L(\infty)>k+h^*), \quad (22)$$

what is just simply inequality that can be derived from fact that queue length of token bucket* is always greater or equal to queue length of token bucket (if they have same arrival and token rate processes).

4. CONCLUSION

We have shown the approach that different types of token bucket can be modelled by a basic network element - a work conserving link. This is very useful result, because now it is not necessary to analyse the token bucket or to make any extra research about it. We can just use all known results for work conserving link and apply them to the token bucket.

Therefore, in the future research we will focus on the work conserving link in more complicated conditions. Despite it is a well-known model, there are still many open problems which solutions would be very useful for dimensioning real-world networks.

ACKNOWLEDGEMENT

This work was supported by the Agency of the Slovak Ministry of Education for the Structural Funds of the EU, under project ITMS:26220120007.

REFERENCES

[1] CHANG, C.-S.: Performance Guarantees in Communication Networks. Springer-Verlag London Limited, Great Britain, 2000.

[2] EVANS, J. – FILSFILS, C.: Deploying IP and MPLS QoS for Multiservice Networks: Theory and Practice, Morgan Kaufmann, 2007.

[3] FERGUSON, P. – HUSTON, G.: Quality of Service: Delivering QoS on the Internet and in Corporate Networks, John Wiley & Sons, 1998.

[4] KLIMO, M. – KOVÁČIKOVÁ, T. – SEGEČ, P.: Selected issues of IP telephony, Communications - scientific letters of the University of Žilina, Vol. 6, No. 4, 2004, pp. 63-70.

[5] KOVÁCS, B.: Mathematical Remarks on Token Bucket. Proceedings of the 17th International Conference on Software, Telecommunications and Computer Networks Table of Contents, Hvar, Croatia, 2009.

[6] KÖRNER, U. – KLIMO, M.: Multi Stream Flow Control Over Shared Communication Resources, Techn. Rep. Dep. of Com. Systems, Lund Institute of Technology, Lund Sweden, 1988.

[7] PROCISSI, G. – GARG, A. – GERLA, M. – SANADIDI, M. Y.: Token Bucket Characterization of Long-Range Dependent Traffic, Computer Communications, Vol. 25, No. 11, 2002, pp. 1009-1017(9).

[8] SMIEŠKO, J.: Using Large Deviation Principles for Optimization of Single Server Queue and Definition of Effective Bandwidth, Journal of Information, Control and Management Systems, Vol. 1, No. 2, 2003, ISSN 1336-1716.

[9] TANENBAUM, A. S.: Computer Networks. 3rd edition, Prentice-Hall, 1996.

Received June 24, 2010, accepted November 10, 2010

BIOGRAPHIES

Jakub Daubner was born on 24. 7. 1984. In 2008 he graduated (Mgr) at the Faculty of Mathematics, Physics and Informatics at the Comenius University in Bratislava. Now he is PhD student at the Department of InfoComm Networks at the University of Žilina. His scientific research is focusing on a dimensioning of computer networks to guarantee given QoS parameters.

Katarína Bachratá was born on 9. 7. 1961. In 1984 she graduated at the Faculty of Mathematics and Physics at the Comenius University in Bratislava, she received RNDr degree (mathematical analysis, differential equations, 1985), PHD degree (in transportation and communication technologies, University of Žilina, 2005). Since 2010 she is working as an assistant professor at the Department of InfoComm Networks at the Faculty of Management Science and Informatics at University of Žilina and her research and educational activities have been oriented in the area of applied mathematics, probability and speech processing.