Investigation of User Arrival and Departure in a Second-Order Proportionally-fair Rate Allocation Algorithm

P. Goudarzi[†+], H. Saidi[+], F. Sheikholeslam[+], A. Askarian[†+*] and M. K. Moussavi[*]
Iran Telecom Research Center[†], Isfahan Univ. of Tech.[+], University of Tehran[*]
pgoudarzi@itrc.ac.ir , hsaidi@cc.iut.ac.ir , a.askarian@ece.ut.ac.ir *and* mmousavi@ut.ac.ir

Abstract

Proportional fairness criterion which has been proposed the first time by F.P.Kelly and his colleagues has a number of properties in allocating users' rates. For example, it resembles the AIMD in the TCP-Reno [1] in rate allocation to users and there exists a well-established stability analysis in Kelly's work relating to stability of rate allocation algorithm. Another outstanding feature is that Kelly et al. try to solve the optimization problem of maximizing the aggregate utility of users in a distributed manner by decomposing the overall system problem into two sub problems that can be solved by network and individual users by introducing a pricing scheme [2]. In the current paper, a new high-speed second-order rate allocation algorithm has been proposed which is based on the Jacobi method. The performance of the algorithm, under users' arrival and departure and background variable bit-rate traffic is evaluated in comparison with the conventional Kelly's algorithm. Simulation results show that proposed method outperforms that of Kelly in convergence rate and is particularly suitable for short-lived users.

1. Introduction

Rate allocation is becoming more demanding as the network capacity grows. Using conventional TCP rate allocation scheme greatly wastes network resources in high bandwidth-delay product environments. There are proposals like FAST TCP [3] and XCP [4] for these environments. XCP achieves its goal by decoupling congestion controller from fairness controller. It assigns AIMD, which is proven to achieve desired fairness properties, to its fairness controller, and MIMD for congestion controller, which is shown to utilize resources more efficiently. FAST TCP is a generalization to TCP Vegas [5] and uses estimated round trip time (RTT) as indication of congestion instead of loss probability.

In this paper, starting from the notion of *proportional fairness* proposed by F. P. Kelly [6], [7], [8], we utilize Jacobi method to solve the optimization problem. As opposed to Kelly's algorithm that uses gradient descent method, we show that we obtain higher rate of convergence which is suitable for high-speed networks and works especially well for short lived flows.

Here we assume our network traffic can adapt itself to network conditions. In another word, we use the term '*elastic*' for our traffic which was introduced by S. Shenker in [9]. As well-known examples of such traffic type we can mention TCP traffic in the current Internet and ABR traffic in the ATM networks.

The paper is organized as follows. In §2 we review some related works and specially the work of F.P. Kelly. In §3 we introduce more closely the high-speed method. §4 is devoted to simulation results and finally we conclude in §5 our paper with conclusion.

2. Background

Consider a network with a set J of resources or links and a set R of users and let C_j denotes the finite capacity of link $j \in J$. Each user r has a fixed route R_r, which is a nonempty subset of J. Also, define a zero-one matrix A, where $A_{rj} = 1$ if link j is in user r's route R_r and $A_{rj} = 0$ otherwise. When the allocated rate to the user r is x_r, user r receives utility $U_r(x_r)$. The utility $U_r(x_r)$ is an increasing, strictly concave and continuously differentiable function of x_r over the range $x_r \geq 0$. Furthermore, assume that the utilities are additive so that the aggregate utility of rate allocation $\chi = (x_r , r \in R)$ is: $\Sigma_{r \in R} U_r(x_r)$. This is a reasonable assumption since these utilities are those of independent network users. Assume that user utilities are logarithmic, then Kelly's formulation of the proportionally-fair rate allocation would be:

$$x_r[n+1] = \left\{ x_r[n] + k_r \cdot \left(\omega_r - x_r[n] \cdot \sum_{j \in R_r} \mu_j[n] \right) \right\}^+ \quad (1)$$

Where:

$$\mu_j[n] = p_j\left(\sum_{s:j \in R_s} x_s[n] \right) \ , \ \{x\}^+ \triangleq \max(0,x) \quad (2)$$

Parameter k_r controls the speed of convergence in equation (1). $p_j(y)$ is the amount that link 'j' penalizes its aggregate traffic 'y' and is a non-negative, continuous increasing function and tends to infinity as aggregate rate 'y' tends to link capacity C_j. Let λ_r be the aggregate charge per unit flow for user r i.e., $\lambda_r = \sum_{j \in R_r} \mu_j[n]$. Given λ_r, user r selects an amount that is willing to pay per unit time, ω_r, and receives a rate $x_r = \omega_r / \lambda_r$.

One of the interpretations is that using (1), the system tries to equalize ω_r with $x_r[n]. \sum_{j \in R_r} \mu_j[n]$ by adjusting $x_r[n]$ value. System (1)-(2) show that the unique equilibrium x^*_r is the solution of the following equation:

$$\omega_r = x^*_r \cdot \sum_{j \in R_r} p_j\left(\sum_{s:j \in R_s} x^*_s \right) \qquad , r \in R \quad (3)$$

3. High-Speed Algorithm

The high-speed algorithm is composed of a two-level hierarchical structure. The inspiration is from the differentiated service notion [10], that looks at the aggregate traffic instead of individual flows. First look at an example. Consider the Fig.1. Let's assume that the network consists of 11 elastic sources that are included in four virtual source-destination users. Dotted lines show the boundaries of the virtual users and thick lines show the aggregate flow of each virtual user that is traversing through the links that belong to backbone (these links are denoted by letters L6, L7 and L8). Sources (destinations) are denoted by 's_i' ('d_i') and as mentioned before, the rate associated with each source-destination pair is denoted by 'x'. Links are unidirectional and in Fig.1, links 6, 7 and 8 constitute the backbone.

As Kelly has shown in [5], stabilized rates of users are:

$$x^*_r = \omega_r / \lambda^*_r \ , r \in \Re$$

Where:

$$\lambda^*_r = \sum_{j \in R_r} p_j\left(\sum_{u:j \in R_u} x^*_u \right)$$

Since it is assumed that the congestion may only occur in the links which belong to backbone, we may consider that λ^*_r is only affected by backbone links and is approximated by:

$$\lambda^*_r \cong \sum_{\substack{j \in R_r \\ \& j \in \text{Backbone}}} p_j\left(\sum_{u:j \in R_u} x^*_u \right) \quad (4)$$

For example, for users 's_1' and 's_2' in Fig.1, we would have:

$$x^*_1 = \frac{\omega_1}{\lambda^*_1} , \ x^*_2 = \frac{\omega_2}{\lambda^*_2} \quad (5)$$

Define:

$$\Lambda^*_1 \overset{\Delta}{=} p_6\left(\sum_{u:L_6 \in R_u} x^*_u \right)$$

Where Λ^*_1 is the aggregate penalty of users 's_1' and 's_2' (λ^*_1 and λ^*_2) in backbone of the network (link '6' in this case).

Then, at the equilibrium point, the aggregate rate of virtual user 1 is:

$$x^*_1 + x^*_2 = \frac{\omega_1}{\lambda^*_1} + \frac{\omega_2}{\lambda^*_2} \cong \frac{\omega_1 + \omega_2}{\Lambda^*_1} \quad (6)$$

In another word, the virtual user 1 might be regarded as a user with logarithmic utility function ($\Omega_1 \log(\chi_1)$) in which $\Omega_1 = \omega_1 + \omega_2$.

If we denote the aggregate rate of virtual user 1 with χ_1, at the equilibrium point we have:

$$\chi^*_1 = \frac{\omega_1 + \omega_2}{\Lambda^*_1} \quad (7)$$

By considering equations (5) and (7) and the assumption that $\lambda^*_1 \cong \Lambda^*_1$, then:

$$x_1^* \cong \frac{\omega_1}{\Omega_1} \cdot \chi_1^* \qquad (8)$$

Let $\mathcal{S} \triangleq \{\mathcal{S}_i \mid i=1,2,\ldots,Q\}$ and $\mathcal{D} \triangleq \{\mathcal{D}_i \mid i=1,2,\ldots,Q\}$ be the sets that represent the virtual sources and virtual destinations . Where, Q represents the number of virtual sources (destinations). For example, in Fig.1 we have Q=4 and $\mathcal{S}_3=\{s_6,s_7\}$, $\mathcal{D}_3=\{d_6,d_7\}$.

If the rate associated with virtual user 'i' at iteration 'n' is denoted by '$\chi_i[n]$', and the rate of end users (as mentioned before) are denoted by small 'x' letter, the algorithm behaves in the following manner:

At the beginning, the algorithm works in the first level of hierarchy and allocates rates to the virtual sources using some high-speed algorithm (such as Jacobi method). Then, each virtual user assigns some proportions of its rate to each end-user within the virtual user. Afterwards, by defining a temporary variable 'w', each user updates its corresponding 'w' parameter and when these new parameters are sent back to the virtual users, the first-level algorithm repeats its computations.

If the assumption in equation (4) is true, when the system is in the vicinity of equilibrium point, users' rates are close to the optimal values. It will be shown that by repeating this procedure, the rates will converge to the optimal rates. We must emphasize here that the 'w' parameters which are updated in the algorithm by end-users have not the interpretation of users' willingness to pay (in contrast with what is discussed in [4] about "ω") and are merely temporary variables. Now, in the mathematical terms [11], the rate assignment by virtual user 'i' to a user 'u' located within virtual user 'i' is:

$$x_u[n+1] = \chi_i[n].\frac{w_u[n]}{W_i[n]} \quad , n = 0,1,2,... \qquad (9)$$
$$i = 1,2,...,Q \quad , u \in i$$

Where notation 'u∈i', means that user 'u' is located within virtual user 'i' and:

$$W_i[n] \stackrel{\Delta}{=} \sum_{u \in i} w_u[n] \qquad (10)$$

Updating $\chi_i[n]$ in the equation (9) is as Jacobi iteration [12] (i=1,2,…,Q):

$$\chi_i[n+1] = \left\{ \chi_i[n] + K_i \cdot \left[W_i[n] - \chi_i[n] \cdot \Lambda_i[n] \right] \Big/ \left[\frac{W_i[n]}{\chi_i[n]} + \chi_i[n] \cdot \frac{\partial}{\partial \chi_i(t)} \Lambda_i(t) \Big|_{t=n} \right] \right\}^+$$

(11)

Where $\chi_i[0] = \varepsilon \cong 0 \quad , \forall i$ and also:

$$\Lambda_i[n] \stackrel{\Delta}{=} \sum_{\substack{j \in R_{S_i} \\ \& j \in \text{Backbone}}} p_j\Big(\sum_{u: j \in R_u} \chi_u[n] \Big)$$

Each 'w' parameter is updated in time scale which is much larger than that of x's using the following relation:

$$w_u[n+1] = \begin{cases} \left\{ w_u[n] + \alpha_u \cdot \left(\dfrac{\dfrac{\omega_u}{\lambda_u[n]} - \dfrac{w_u[n]}{\Lambda_i[n]}}{\dfrac{\omega_u}{\lambda_u[n]}} \right) \right\}^+ & \text{for } n = 0, N, 2N, \ldots \\ w_u[n] & \text{otherwise} \end{cases} \qquad (12)$$

$$i=1,2,\ldots,Q \quad , u \in i$$

Where $w_u[0]= \omega_u$ (the user-logarithmic utility function parameter), $u \in i$, i=1,2,…,Q and N is some large positive integer. α_u is some positive constant ($0 < \alpha_u < \delta_u \quad , \forall i, u \in i$) that controls the convergence speed in equation (12) and $\delta_u > 0$ is an upper bound for α_u.

Equation (11) is in fact a form of the projected Jacobi method, as Bertsekas et al. have defined in [12]. The idea behind equation (12) is that users try to adjust their final rate which are assigned to them by first-level algorithm i.e. ($w_u[n] / \Lambda_i[n]$) to the Kelly's rate i.e. ($\omega_u / \lambda_u[n]$) by changing their 'w' parameters. The stability property of this algorithm is discussed in [13].

4. Simulation Results

Consider the network topology of Fig.2 which is composed of 87 elastic users and 94 links. Gray nodes are the network's backbone boundary. Simulation results are composed of two parts.

Part one:

In part one, we assume that odd-numbered users (for example user 1,3,5,…) arrive in the system with a Poisson distribution and their existence persist with an exponential distribution and assume that even-

numbered users persist all over the simulation time. In this part, we assume that links 11, 15, 17, 47, 48, 49 and 91 are actual bottlenecks and their capacities are listed in table (1). Other link capacities are selected much larger than the bottleneck links such that they can not impose our rate assignment. In Kelly's method, we have selected $k_r = 0.00005, r \in R$ and in the proposed method, we have selected K_i=0.00005, i=1, 2… Q. We have selected Q=22 ، N=1000 and α_i=0.6 for each i in equation (12). As we have mentioned before, the users' utility functions are logarithmic and their ω parameters are given in table (2). Links penalty functions in the Kelly method and proposed method are selected according to relations (13) and (14) respectively with $\varepsilon_1=10^{-2}$ and $\varepsilon_2 = 10^{-8}$ which are selected small enough to approximate an exact penalty function.

$$p_j(y)= (y-c_j+\varepsilon_1)^+/\varepsilon_1^2 \quad , j\in J \qquad (13)$$

$$p_j(y)= \varepsilon_2 . \tan(\pi . y/(2c_j)) \quad , j\in J \qquad (14)$$

In Fig.s 3 to 6 the rate allocated to two temporary user and two permanent users are compared. As it can be verified, the rate allocated to short-lived users in the proposed algorithm is larger than that of Kelly but instead, as simulations show, the rate allocated to permanent users may be less in some times in the proposed method. In the real-time applications that almost some short-time greedy users are present, it can be concluded that the proposed method is more suitable than that of Kelly.

Part two:

In part two, we have adopted a similar approach as that of Walrand [7] and Başar[14] for simulating the rates allocated to the users with different propagation delays. We have used the OPNET discrete-event simulator. We have assumed that those users whose numbers are multiple of 5 (such as 5, 10, 15…) act as background variable-rate traffic on other users. The bottleneck links are the same as part one, but their capacity is selected to be 100kBps, other link capacities are selected 100MBps. All links' propagation delays is set to 5 ms. We have assumed that sources have data for sending at all times (greedy sources). All links' buffer sizes are set to 100 packets and so loss is occurred in the network.

We have used *go back* n method for re-sending the packets that are double acknowledged. Links' scheduling discipline is FIFO. As in TCP, Slow-Start method is used for initializing the rate allocation.

Receivers' window sizes are set to unity and sender window size in Kelly and Jacobi method is updated according to relations (15) and (16) respectively.

$$cwnd_r[n+1] = \left\{ cwnd_r[n] + k_r \cdot RTT_r[n] \cdot \left(\omega_r - \frac{cwnd_r[n]}{RTT_r[n]} \cdot d_r[n] \right) \right\}^+ \qquad (15)$$

$$CWND_i[n+1] = \left\{ CWND_i[n] + K_i \cdot RTT_i[n] \cdot \left[W_i[n] - \frac{CWND_i[n+1]}{RTT_i[n]} \cdot d_i[n] \right] \right. \Bigg/ \left. \left[d_i[n] + \frac{CWND_i[n]}{RTT_i[n]} \cdot \left(\frac{d_i[n] - d_i[n-1]}{\frac{CWND_i[n]}{RTT_i[n]} - \frac{CWND_i[n-1]}{RTT_i[n-1]}} \right) \right] \right\}^+ \qquad (16)$$

Where:

$$d_r[n] = RTT_r[n] - \bar{d}_r \qquad (17)$$

$\bar{d}_r$ is the user 'r' propagation delay and its round trip time is RTT_r. We have used $k_r=K_r$=0.0003.

It is important that as congestion occurs only in the bottleneck links located in the backbone, the rate allocation algorithm is only consisted of equations (9) and (11) and equation (12) has no effect on the rate allocation algorithm.

The simulation results for users in Fig. 2 are depicted in Figs. 7 to 10. We have compared in these Figures, the proposed second order method with the Kelly's method and TCP. It can be verified that the proposed method, outperforms that of Kelly in convergence speed.

On the other hand, another outstanding feature of our rate allocation strategy is that the user rates in the proposed method and that of Kelly, have less fluctuations with respect to TCP. Also, the rate allocation is TCP *friendly* because none of the allocated rates in the Jacobi or Kelly's methods are greater than their corresponding TCP rate allocation.

As equations (15) and (16) use only the RTT and propagation delay of the connection, they can be implemented in an End-to-End manner even in the current Internet.

5. Conclusion

In the current paper, we proposed a high speed second order algorithm and compared it with the conventional Kelly's algorithm in the users arrival and departure and background traffic aspect. Simulation results show that the proposed method, allocate more

rates to temporary users and hence is a good candidate in some real-time applications. In the presence of variable bit-rate background traffic, the proposed algorithm, outperforms that of Kelly in convergence speed.

6. References

[1] V. Jacobson, "Congestion Avoidance and control," *ACM SIGCOMM'88*.

[2] R.J. Gibbens and F.P. Kelly. (1998,June) Resource pricing and the evolution of congestion control. [Online].available: http://www.statslab.cam.ac.uk/~frank

[3] C. Jin, D. X. Wei, S. H. Low, **"**FAST TCP: Motivation, Algorithms, Performance,**"** *INFOCOM 2004*

[4] D. Katabi, M. Handley, C. Rohrs, **"** Internet Congestion Control for High Bandwidth-Delay Product Networks" *ACM SIGCOMM 2002*

[5] L.S. Brakmo and L.L. Peterson, "Tcp vegas: End to end congestion avoidance on a global internet," *IEEE J. Select. Areas Commun.*, vol.13, pp. 1465-1480, Oct. 1995.

[6] FP Kelly, AK Maulloo and DKH Tan, "Rate control for communication networks: shadow prices, proportional fairness and stability," *J. Oper. Res. Soc.*, vol. 49, no. 3, pp. 237-252, Mar. 1998.

[7] J. Mo and J. Walrand, "Fair End-to-End Window-Based Congestion Control," *IEEE/ACM Transactions on Networking*, vol.8, pp.556-567, no. 5, Oct. 2000.

[8] L. Massoulié and J. Roberts, "Bandwidth sharing : objectives and algorithms," in *Proc. IEEE INFOCOM*, vol.3, 1999, pp. 1395-1403.

[9] S. Shenker, "Fundamental design issues for the future Internet," *IEEE J Selected Areas Commun.*, vol.13, no.7, pp.1176-1188, Sept. 1995.

[10] [RFC 2475] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang and W. Weiss. "An Architecture for Differentiated Services." December 1998.

[11] P.Goudarzi, H.Saidi and F.Sheikholeslam, "Investigation of User Arrival and Departure in a High-Speed Proportionally-Fair Rate Allocation Algorithm", *IST2003*, 16-18 Aug. 2003, Isfahan-IRAN.

[12] D. Bertsekas and J. Tsitsiklis, *Parallel and Distributed Computation.* Englewood Cliffs, NJ: Prentice-Hall, 1989.

[13] P.Goudarzi, H.Saidi and F.Sheikholeslam, "A Fuzzy-Hierarchical Algorithm for Proportionally-Fair Rate Allocation to Elastic Users," IEICE Trans. Commun., Vol. E87-B, No.11, Nov. 2004.

[14] T.Alpcan and T.Başar, **"**Utility-Based Congestion Control Schemes for Internet-Style Networks with Delays**"**, *INFOCOM 2003.*

.

Table 1-Bottleneck link capacities in part one

Capacity	Bottleneck link	Capacity	Bottleneck link
5	11	10	15
7	17	5	47
3	48	8	49
22	91		

Table 2-Users' utility parameters in part one

ω	User	ω	User	ω	User	ω	User
0.03	67	0.04	45	0.04	23	0.05	1
0.025	68	0.07	46	0.07	24	0.05	2
0.025	69	0.03	47	0.025	25	0.03	3
0.03	70	0.025	48	0.03	26	0.03	4
0.05	71	0.025	49	0.02	27	0.04	5
0.05	72	0.03	50	0.05	28	0.07	6
0.03	73	0.05	51	0.03	29	0.03	7
0.03	74	0.05	52	0.03	30	0.025	8
0.04	75	0.03	53	0.04	31	0.025	9
0.07	76	0.03	54	0.07	32	0.03	10
0.03	77	0.04	55	0.025	33	0.02	11
0.025	78	0.07	56	0.03	34	0.05	12
0.025	79	0.03	57	0.02	35	0.03	13
0.03	80	0.025	58	0.05	36	0.03	14
0.05	81	0.025	59	0.03	37	0.04	15
0.05	82	0.03	60	0.03	38	0.07	16
0.03	83	0.05	61	0.07	39	0.025	17
0.03	84	0.05	62	0.023	40	0.03	18
0.04	85	0.03	63	0.05	41	0.02	19
0.07	86	0.03	64	0.05	42	0.05	20
0.03	87	0.04	65	0.03	43	0.03	21
		0.07	66	0.03	44	0.03	22

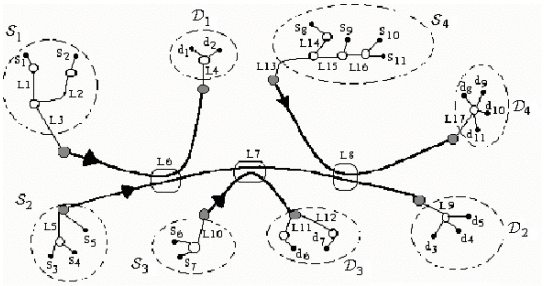

Figure 1. A sample network with two levels of hierarchy

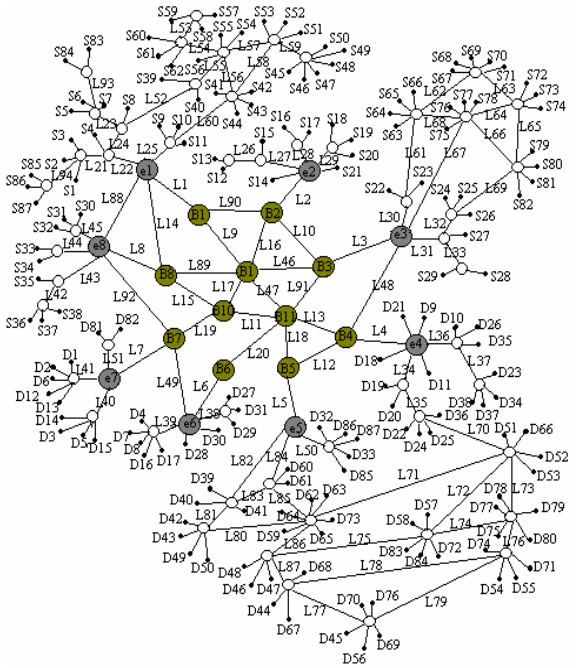

Figure 2. Simulated network topology

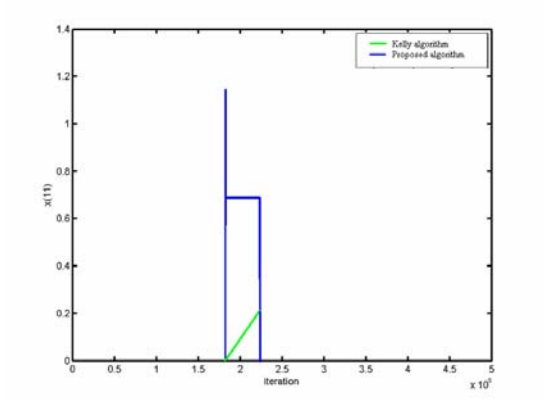

Figure 3. Temporary user 11

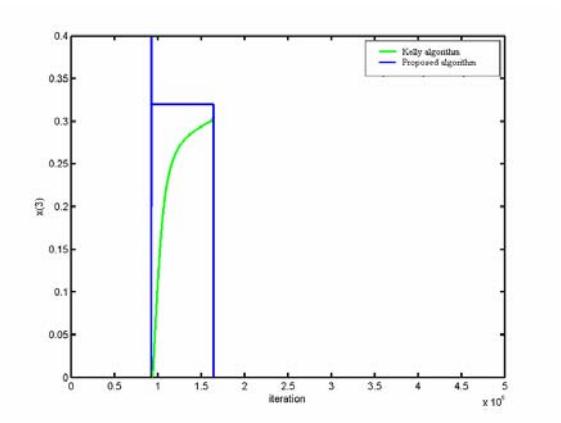

Figure 4. Temporary user 3

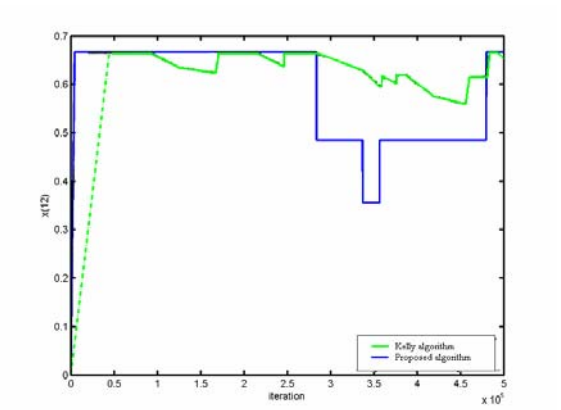

Figure 5. Permanent user 12

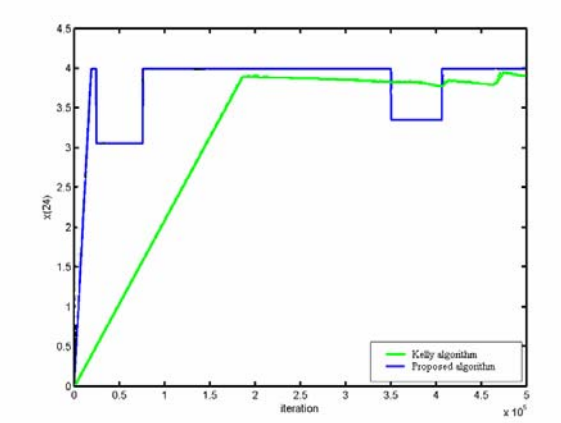

Figure 6. Permanent user 24

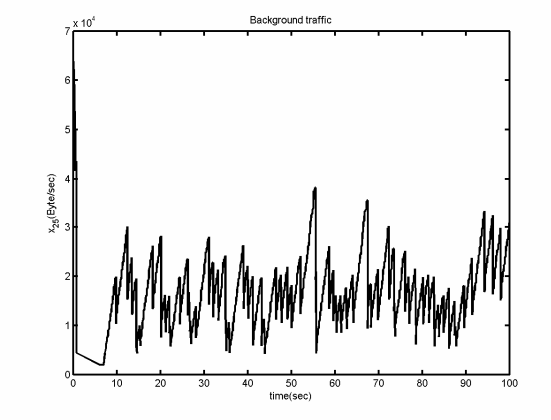

Figure 7. Background traffic 25

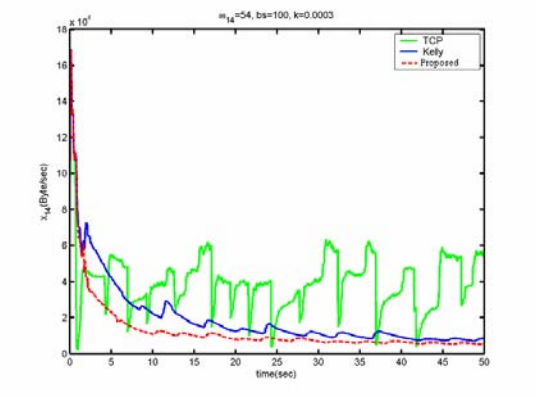

Figure 10. Rate allocated to user 14

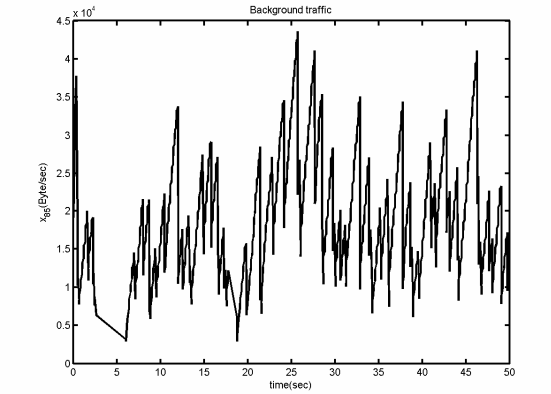

Figure 8. Background traffic 85

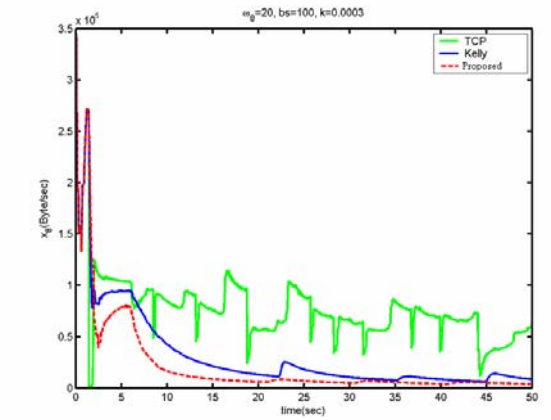

Figure 9. Rate allocated to user 8