Modified Rate Control Algorithm for wireless Cognitive Networks

A.Askarian, M.J.Omidi,
Faculty of Electrical And Computer Engineering Isfahan University Of Technology, Isfahan, Iran
a.askarian@ec.iut.ac.ir , Omidi@cc.iut.ac.ir

***Abstract*— We have proposed a fluid flow rate allocation scheme for wireless networks. We show that this method is well suited for ephemeral web-like traffic which is proliferating as network bandwidth is increasing. This algorithm adapts itself to varies environment parameters such as bandwidth and SNR. The simulation results for both permanent and short-lived flows are promising.**

Key Words— Rate control, Cognitive Network, TCP, bandwidth sharing, short-lived traffics, Bandwidth

I. INTRODUCTION

Rate control is becoming a more demanding problem than ever as networks grow in capacity. Using conventional congestion control algorithms in these environments will never meet the objectives these algorithms are designed for. An important concern is achieving fair rate allocation. In these high speed environments, since high bandwidth is available for a flow, its life my finish before it reaches the anticipated fair rate, should conventional rate allocation scheme be deployed. The number of these ephemeral flows increases as the network capacity increases, calling for urgent need of congestion control algorithms with high convergence rate suitable for short lived flows.

Different methods of congestion control can be classified in three ways [1]: *window based* vs. *rate based*, *implicit* vs. *explicit* and *hop by hop* vs. *end to end*. In *window based* scheme, transmitting rate is controlled by error control window, while in the latter a fluid flow notion is used and transmission rate is controlled directly. TCP congestion control, as the most widely implemented algorithm in the Internet is a *window based*, *implicit* and *end to end* algorithm. Rapid increase in speed and capacity, which is a main characteristic of future networks and a necessity for high performance computational grids, cause TCP to face instability.

In this paper, after providing evidence of this performance loss, we try to pinpoint the deficiencies with current algorithms. Then we will provide necessary modifications to current flavors of TCP for future high-speed networks.

II.CLASSIFICATION OF CONGESTION CONTROL DISCIPLINES

Congestion control disciplines can be classified in three ways.

A. *Dynamic Window or Dynamic Rat*

Dynamic window scheme make use of error control window size for congestion control. Actually it adjusts the rate of transmission by adjusting window size.

$$rate = \frac{WindowSize}{RTT} \qquad (1)$$

In dynamic rate scheme, transmission rate is adjusted directly in the transmitter using timers.

Upon sending a packet, the transmitter sets a timer with timeout value equal to the inverse of the current allowable transmission rate. When timeout happens, next packet is sent.

Rate based scheme is more accurate in adjusting the rate, but it has two drawbacks. First it needs timers which are expensive; also it is not robust to loss of rate controlling information that is feed back to the transmitter, i.e. if the feedback packet for reducing rate become lost, the transmitter will continue with high rate which is dangerous for the network. In the window based scheme, after sending a window worth packets, the transmitter waits for receiving Acks, and is robust to Ack loss.

B. *Explicit vs. Implicit Schemes*

In explicit discipline, sources solicit explicit congestion information from the network. Using this information, sources can adjust their rates.

This method is accurate, but it has computational and communicational overhead.

In implicit method, sources infer the level of congestion in the network; using for example packet loss (Ack timeout) or duplicate Acks as an indication of congestion.

C. *Hop by Hop vs. End to End*

In Hop by Hop scheme, congestion control is done between every adjacent pairs of network elements. This

may lead to fast convergence but at the cost of complexity of intermediate nodes which is not desirable.

End to End schemes push the complexity to edge of the network, i.e. the complexity is in end hosts.

Various congestion control disciplines can be classified in these three ways (TABLE I). [2]

TABLE I

	Explicit	Explicit	Implicit	Implicit
	Dynamic Window	Dynamic Rate	Dynamic Window	Dynamic Rate
End to End	DECbit	ATM Forum EER	TCP Tahoe TCP Reno TCP Vegas	NETBLT Packet pair
Hop by Hop	Credit Based	Mishra Kanakia		

Classify various congestion control disciplines

III.TCP CONGESTION CONTROL

TCP congestion control is an End to End, Dynamic Window and Implicit scheme. Various kinds of TCP congestion control depends on their way of window size adjustments based on their inference about congestion level of the network.

There are some common expressions for window size adjusting in these schemes.

In old implementations of TCP, the window size was initialized to advertised window by the receiver (awnd), while it works when the two hosts are on the same LAN it is problematic when the hosts are on different networks.

A. *Slow Start*

Congestion window (cwnd) in the transmitter is initialized to one segment size (which is by default 536 or 512).

Upon receiving an ACK, cwnd size will increase by one, therefore it is doubled every RTT and has an exponential growth.

B. *Congestion Avoidance*

After cwnd passes a threshold called ssthresh, Congestion Avoidance phase will start. In Congestion Avoidance phase, upon receiving each ACK, the cwnd will increase by:

$$\frac{segsize \times segsize}{cwnd} \qquad (2)$$

bytes, where cwnd is measured in bytes. Or simply it increments by:

$$\frac{1}{cwnd} \qquad (3)$$

segments. It is clear that the cwnd would increase by one every RTT.

Now we will have a quick look to TCP Tahoe congestion control disciplines, and the closer look and other disciplines will be presented in the next report.

C. *TCP Tahoe*

Presented as the first congestion control discipline by Van Jacobson in 1988, TCP Tahoe works as follows:

Initialization of cwnd to 1

Slow Start

Congestion Avoidance (after ssthresh)

After receiving three duplicate ACKs, cwnd is set to one

Slow Start

Other conventional TCP congestion control disciplines are:

TCP Reno

TCP New Reno

TCP SACK

TCP Vegas

We could model TCP congestion control algorithms with Additive Increase Multiplicative Decrease (AIMD) that discuss in next section

.

IV. ADDITIVE INCREASE MULTIPLICATIVE DECREASE

End-to-end congestion control in packet network is based on binary feedback and the adaptation mechanism of additive increase multiplicative decrease. We describe here a motivation for this approach. It comes from the following modeling, from [3].

Assume I source, labeled $i = 1,\ldots,I$ send data at a time dependent rate $x_i(t)$, into a network constituted of one buffered link, of rate c. We assume that time is discrete and that the feedback cycle lasts exactly one time unit. During one time cycle of duration 1 unit of time, the source rate are constant, and the network generates a binary feedback signal $p[n] \in \{0,1\}$, send to all source ,sources react to the feedback by increasing the rate if $p[n] = 0$, and decreasing if $p[n] = 1$.we further assume that the feedback defined by

$$p[n]=\begin{cases}0, & \text{if } \sum_{i=1}^{I} x_i[n]\le c\\ 1, & \text{o.w.}\end{cases} \qquad (4)$$

The value c is the target rate witch we wish the system not to exceed. At the same time we wish that the total traffic be close to c as possible.

We are looking for a linear adaptation algorithm, namely, there must exist constants u_0,u_1 and v_0,v_1 such that

$$x_i[n+1]=u_{p[n]}x_i[n]+v_{p[n]} \qquad (5)$$

We want the adaptation algorithm to converge towards a fair allocation. In this simple case there is one single bottleneck and all fairness criteria are equivalent. At equilibrium,

We should have $x_i=\frac{c}{I},\forall i$. However, a simple adaptation algorithm as describe above cannot converge, but in contrast, oscillates around the ideal equilibrium.

We now derive a number of necessary conditions. First, we would like the rates to increase when the feedback is 0, and decrease otherwise. Call $y[n]=\sum_{i=1}^{I} x_i[n]$ when we have

$$y[n]=u_{p[n]}y[n]+v_{p[n]} \qquad (6)$$

Now our condition implies that, $\forall y\ge 0$:

$$\begin{aligned}u_0y+v_0>y\\ u_1y+v_1<y\end{aligned} \qquad (7)$$

This given the following necessary conditions

$$\begin{aligned}u_0y+v_0>y\\ u_1y+v_1<y\end{aligned}$$

And

$$\begin{cases}u_1<1\&v_1\le 0\\ or\\ u_1=1\&v_1<0\end{cases} \qquad (8)$$

Now we also wish to ensure fairness. A first step is to measure how much a given rate allocation deviates from fairness. We follow the sprite of [3] and use as a measure of unfairness the distance between the rate allocation $\vec{x}$ and its nearest fair allocation $\Pi(\vec{x})$, where Π is the orthogonal projection on the set of fair allocation, normalized by the length of the fair allocation (Figure 1). In other words, the measure of unfairness is

$$d(\vec{x})=\frac{\|\vec{x}-\Pi(\vec{x})\|}{\|\Pi(\vec{x})\|} \qquad (9)$$

Figure 1) illustrates that: (1) when we apply multiplicative increase or decrease, unfairness is unchanged; (2) in contrast, an additive increase decrease the unfairness, whereas an additive decrease increase the fairness [4].

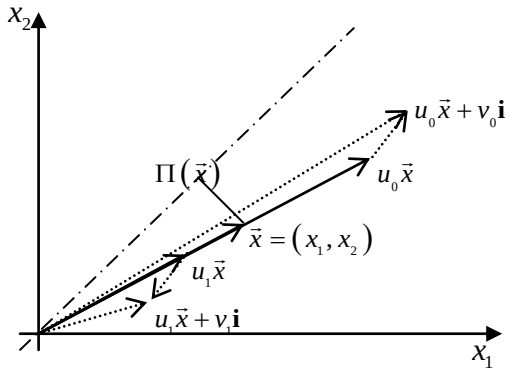

Figure 1: Measure of unfairness the distance between the rate allocation $\vec{x}$ and its nearest fair allocation

Now consider simple network below for numerical simulations of AIMD.

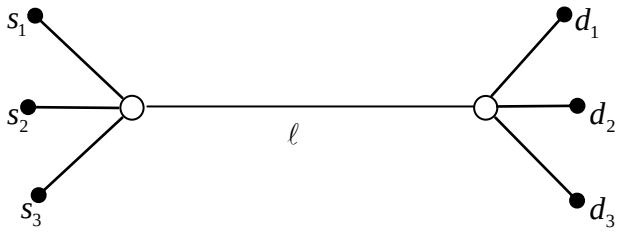

Figure 2: network topology for numerical simulations of AIMD

In figure 2 there are three sources, with initial rate of $2Mbps$, $12Mbps$ and 0. The total link rate is $10Mbps$. The third source is inactive until time unit $n=300$.

In equation (5) we have:

$$u_{p[n]}=\begin{cases}1, & p[n]=0\\ 1-s, & p[n]=1\end{cases}=1-s.p[n] \quad , 0< s <1 \qquad (10)$$

And

$$v_{p[n]} = \begin{cases} k, & p[n] = 0 \\ 0, & p[n] = 1 \end{cases} = k \cdot (1 - p[n]) \quad , \text{k>0} \tag{11}$$

Figures bellow show the rates for the three souse, as well as the aggregate rate and the measure of unfairness. The measure of unfairness is counted for two source until the time 300 [5].

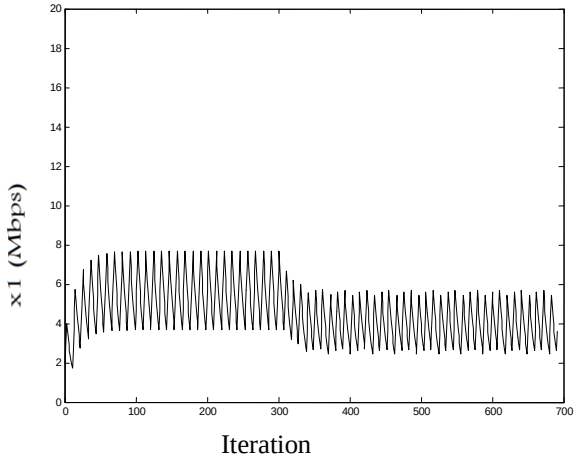

Figure 3: rate allocation for S1

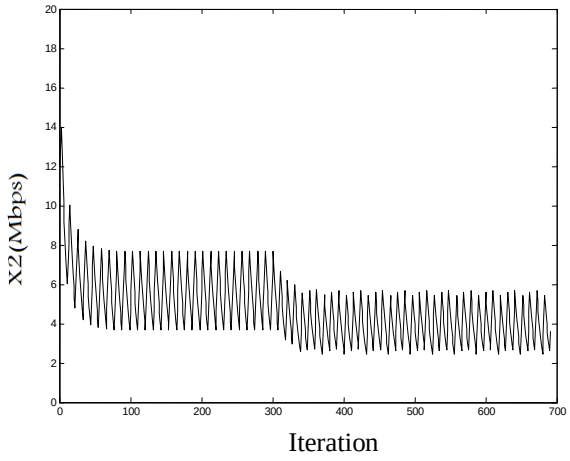

Figure 4: rate allocation for S2

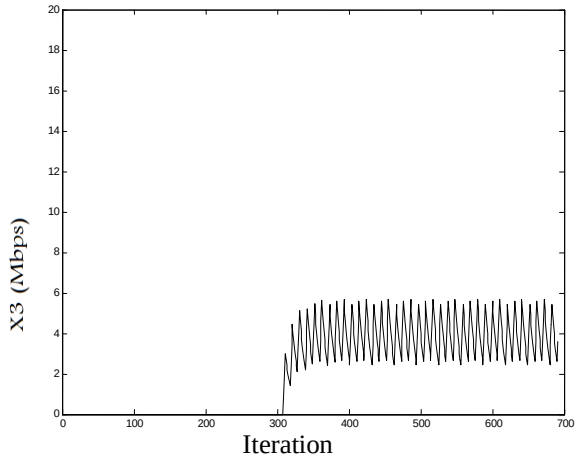

Figure 5: rate allocation for S3

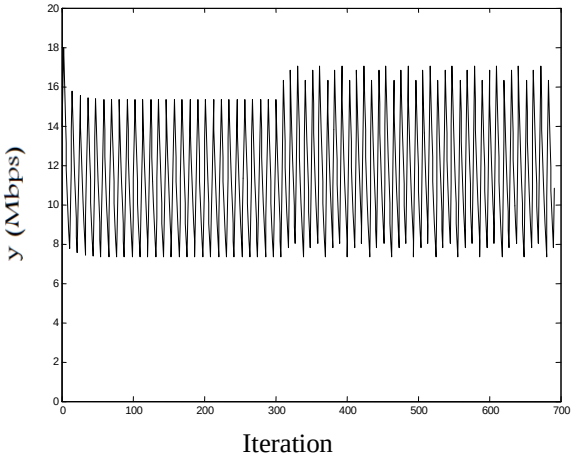

Figure 6: bottleneck rate

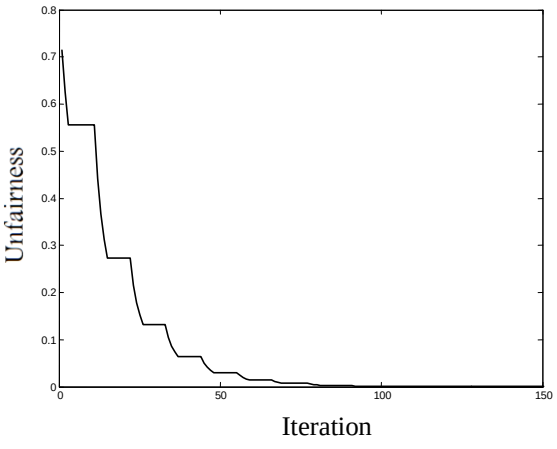

Figure 7: unfairness for S1 , S2

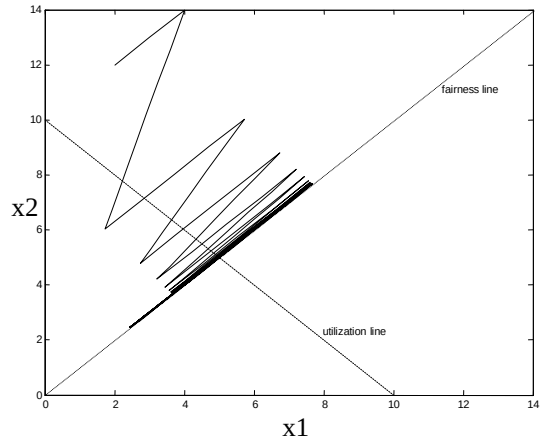

Figure 9: AIMD progress

V. MODIFY AIMD FOR HIGH COGNETIVE NETWORK

We define increase functions $f(x_i, c_i)$ and $g(x_i, c_i)$, in which x_i is the rate of source i and c_i is the source i link capacity. If $p[n] = 0$ sources increase their rate using f and if $p[n] = 1$ sources decrease their function using g, so we have:

$$u_{p[n]} = \begin{cases} 1, & p[n] = 0 \\ f(x_i, c_i), & p[n] = 1 \end{cases} \qquad (12)$$

, $f(x_i, c_i) \geq 0$

And

$$v_{p[n]} = \begin{cases} g(x_i, c_i), & p[n] = 0 \\ 0, & p[n] = 1 \end{cases} \qquad (13)$$

, $0 \leq g(x_i, c_i) \leq 1$

For simulation we use:

$$f(x_i, c_i) = 2 * x_i^{0,6} + c_i^{0.6} \qquad (14)$$

And

$$g(x_i, c_i) = 1 - 1/c_i \qquad (15)$$

Our network topology is also figure2 in which the bottleneck rate is $10Gbps$ and

$c_i = 10Mbps \quad i = 1, 2, 3$

Figures bellow compare our method with conventional AIMD.

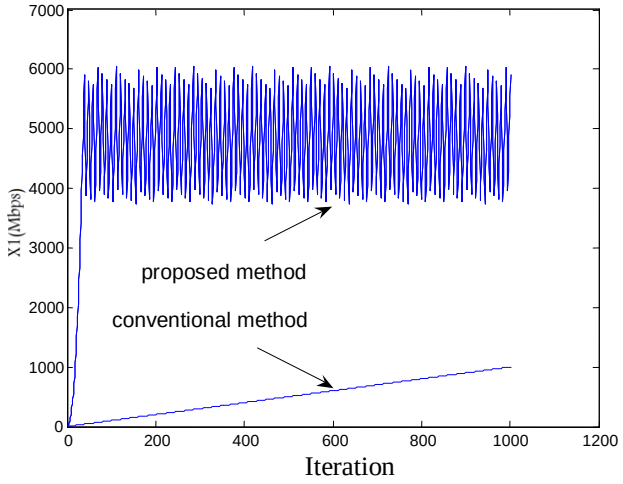

Figure 10: rate allocation for S1

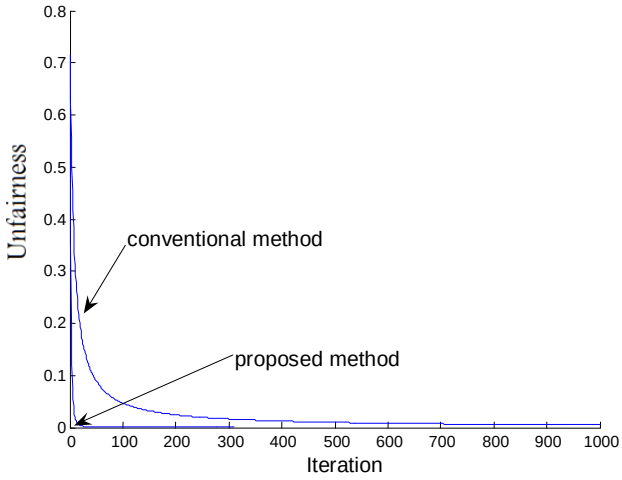

Figure 11: unfairness of source 1

Suppose that we have short-live traffic between the time n=1500 and 6000. Our simulation show that AIMD allocates an unfair rate to this traffic figure 12.

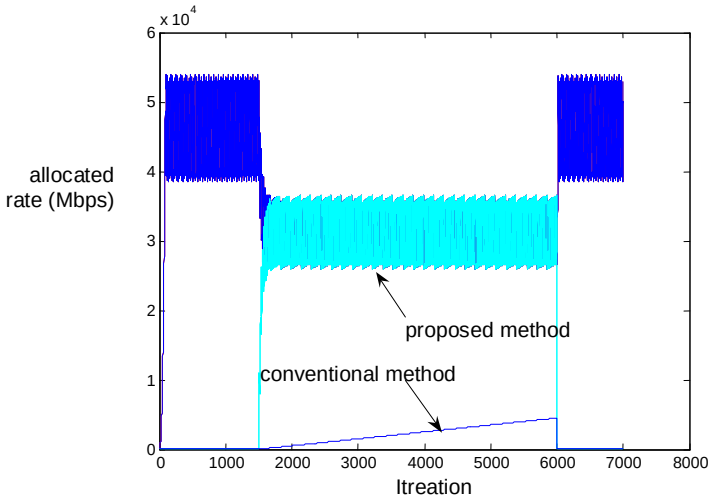

Figure 12: short-live traffic

VI. NETWORK OPTIMIZATION

Rate allocation in data networks is studied as a distributed optimization problem [6]-[7]. This optimization problem establishes a well defined framework for studying network characteristics, designing new protocols or improving the existent ones. It is known that the currently implemented TCP. congestion control also is a special case of this general framework, by which its performance is evaluated and enhancements are proposed. At this section we review rate allocation as the optimization problem.

A. Minimum Cost Flow Control

Minimum cost flow control or MCFC proposed by S. J. Golestani [7,8] formulates end to end congestion control as a global optimization problem, based on which a class of algorithms for adjusting end host rates are proposed. This theoretical approach leads to a class of congestion control algorithms that describes the common algorithms like TCP as its special case. We consider a network consisting of a set $\mathbf{L} = \{l=1,\ldots,L\}$ of links and a set $\mathbf{S} = \{s=1,\ldots, S\}$ of sessions. Let x_s denote the average rate of session s traffic and f_l denote the average traffic of link l. The fraction of traffic of session s carried over link l is denoted by ϕ_{sl}. This defines routing matrix $\Phi = (\phi_{sl}, s\in\mathbf{S}, l\in\mathbf{L})$. In single path case as is in the Internet, $\phi_{sl} \in \{0,1\}$. It follows that

$$f_l = \sum_{s=1}^{S} \varphi_{sl} \cdot x_s, \qquad l \in \mathbf{L} \qquad (16)$$

The rates to be allocated to the sessions should satisfy

$$0 \le x_s \le x_s^d \qquad (17)$$

where x_s^d is the desired rate of session s. To formulate the congestion control as a resource allocation and optimization problem, two cost functions are considered in this framework. First there is a user dissatisfaction cost function $e_s(x_s)$, which demonstrates the cost of limiting the rate of session s to x_s. Based on the previous assumptions, the network congestion control problem is formulized based on the following optimization problem:

$$\min_{\vec{r}} J(\vec{r}) \triangleq \sum_{s=1}^{S} e_s(r_s) + \sum_{l=1}^{L} g_l(f_l) \qquad (18)$$

Subject to:

$$0 \le x_s \le x_s^d$$

$$\vec{f} = \Phi\vec{x}$$

Incremental reward function of session s is defined as

$$h_s(x_s) \triangleq -\frac{\partial}{\partial x_s} e_s(x_s) \qquad (19)$$

Since $e_s(x_s)$ is decreasing and convex, $h_s(x_s)$ is positive and decreasing. *Congestion measure* of a session s is defined as the increase of network congestion due to a unit increase in x_s, as follows:

$$\gamma_s(\vec{f}) \triangleq \frac{\partial}{\partial x_s} \sum_{l=1}^{L} g_l(f_l) = \sum_{l=1}^{L} \varphi_{sl} \cdot g_l'(f_l) \qquad (20)$$

It is always a positive quantity. In the single path routing scenario (5) reduces to:

$$\gamma_s(\vec{f}) = \sum_{l\in\rho_s} g_l'(f_l) \qquad (21)$$

where ρ_s is the path used by session s.

It is shown in [7] that using Kuhn-Tucker theory [10] the optimality condition holds for the optimization problem (18) is as follows. Assuming that $g_l(\cdot)$ and $e_s(\cdot)$ have the first and second derivatives satisfying $g_l' > 0, g_l'' > 0, e_s' < 0$ and $e_s'' > 0$, the following is the set of necessary and sufficient conditions for the session rate vector $\vec{x}^*$ to be the solution for the convex optimization problem (18):

$$h_s\left(x_s^*\right)\begin{cases}\le \gamma_s\left(\vec{f}^*\right), & \text{if } x_s^* = 0\\ = \gamma_s\left(\vec{f}^*\right), & \text{if } 0 < x_s^* < x_s^d\\ \ge \gamma_s\left(\vec{f}^*\right), & \text{if } x_s^* = x_s^d\end{cases} \quad (22)$$

For $s \in \mathbf{S}$ and $\vec{f}^* = \Phi\vec{x}^*$.

Accordingly, the iteration of gradient projection algorithm for solving (18) would be:

$$x_s[n+1] = \begin{cases}0, & \text{if } x_s[n] + k\left(h_s\left(x_s[n]\right) - \gamma_s\left(\vec{f}\right)\right) \le 0\\ x_s^d, & \text{if } x_s[n] + k\left(h_s\left(x_s[n]\right) - \gamma_s\left(\vec{f}\right)\right) \ge x_s^d\\ x_s[n] + k\left(h_s\left(x_s[n]\right) - \gamma_s\left(\vec{f}\right)\right)\end{cases} \quad (22)$$

k in above is the step size and has a critical role in determining the convergence speed of the algorithm. As we will see in the next section, by using adaptable step size, i.e., changing k according to the current transmission rate we can achieve high convergence rate well suited for ephemeral flows copious in high speed networks. But now we see how to relax the constraint imposed by x_s^d.

We assume that the users are greedy, i.e., they use up all available rate. Regarding this we can rewrite iterations in (22) as follows[7]:

$$x_s[n+1] = \max\left\{0, x_s[n] + k\left(h_s\left(x_s[n]\right) - \gamma_s\left(\vec{f}\right)\right)\right\} \quad (23)$$

VII.MODIFIED ALGORITHM FOR COGNITIVE NETWORK

The iterative equation (23) is shown to have desired equilibrium properties in term of fairness and stability. But the convergence rate of this algorithm should be enhanced in order to be suitable for high speed environments where short-lived flows are abundant. We consider an adaptable variable k, that updates itself in each iteration of the algorithm according to the following:

$$k_s\left(x_s[n]\right) = \frac{\upsilon}{\beta + \max\limits_{i\in\rho_s}\left\{\left(\frac{x_s[n]}{c_i}\right)^\theta\right\}} = \frac{\upsilon}{\beta + \left(\frac{x_s[n]}{\min\limits_{i\in\rho_s}\{c_i\}}\right)^\theta} \quad (24)$$

The max or min term is over the entire links session s passes. In simulations we observed that this can be relaxed and considering just one link would suffice.

Now we consider example of Figure 13, where multiple sessions interact with each other and there is multiple bottleneck links. Also in this case we consider user arrival and departure or ephemeral flows.

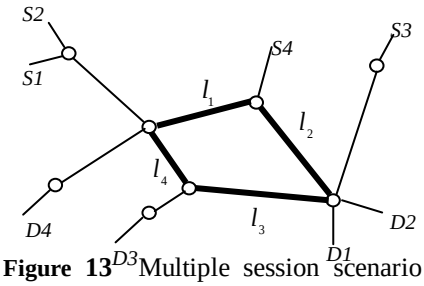

Figure 13[D3]Multiple session scenario sharing the same backbone

In this scenario, sessions share the same backbone and bottleneck link. The bottleneck link capacities are: $c_1 = 50Mbps, c_2 = 30Mbps, c_3 = 60Mbps, c_4 = 70Mbps$. The path sessions use is shown in Table 1

Table1 Session paths

Session1	l_3, l_4
Session2	l_1, l_2
Session3	l_4
Session4	l_1

Figure 14 shows the overall increase in convergence rates of all sessions. As can be seen, the equilibrium fairness is intact in high speed algorithm. Si in the figure stands for session *i*.

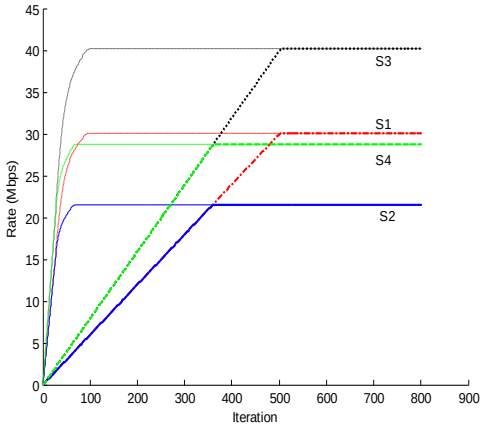

Figure 14. Convergence rate enhancement for stable flows in scenario of Figure 13

VI.CONCLUSION

We propose a modification to conventional TCP congestion control. This algorithm enjoys a high convergence speed, making it suitable for cognitive

networks. We show that new method is suited for short-live traffic.

REFERENCE

[1] S. Keshav "An Engineering Approach to Computer Networking" Adisson-Wesley, 1997

[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] D. Chiu and R. Jain, "Analysis of the Increase and Decrease Algorithms for Congestion Avoidance in Computer Networks," *Computer Networks and ISDN Systems*, vol. 17, pp. 1-14, 1989.

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

[5] Lapsley, D.E. and Low, S.H., "An optimization approach to ABR control." In Proceedings of the ICC, June 1998.

[6] Kelly F.P., Maulloo A.K. and Tan D.K.H., "Rate control communication networks: shadow prices, proportional fairness and stability," J. Oper. Res. Soc., vol. 49, no. 3, pp. 237-252, Mar. 1998.

[7]Golestani, S.J. and Bhattacharyya, S., "End-to-End Congestion Control for the Internet: A global optimization framework." In Proc. Of International Conf. On Network Protocols(ICNP), Oct. 1998.