International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:3, No:2, 2009

The New AIMD Congestion Control Algorithm

Hayder Natiq Jasem, Zuriati Ahmad Zukarnain, Mohamed Othman, and Shamala Subramaniam

***Abstract*—**Congestion control is one of the fundamental issues in computer networks. Without proper congestion control mechanisms there is the possibility of inefficient utilization of resources, ultimately leading to network collapse. Hence congestion control is an effort to adapt the performance of a network to changes in the traffic load without adversely affecting users perceived utilities. AIMD (Additive Increase Multiplicative Decrease) is the best algorithm among the set of liner algorithms because it reflects good efficiency as well as good fairness. Our control model is based on the assumption of the original AIMD algorithm; we show that both efficiency and fairness of AIMD can be improved. We call our approach is New AIMD. We present experimental results with TCP that match the expectation of our theoretical analysis.

***Keywords*—**Congestion control, Efficiency, Fairness, TCP, AIMD.

I. Introduction

THE need for communication has always been the driving force behind the invention of technologies which is bringing the people closer day by day. The telecommunication industry began with a wired connection and then the recent advancement in different technologies has made it possible to communicate using wireless technologies. Invention of computer, software, hardware, micro chips has changed the whole concept of communication. Today we see a blend of wired and wireless communication network using heterogeneous technologies.
The heterogeneity in communication networks has not only opened the door of different formats of communication but also has induced lot of issues that need to be addressed in order to provide high quality communication. Congestion is one of the major issues among them. A modern communication network is built using a number of well connected devices or nodes, which have limited local capacity and resources. Currently, two different transport paradigms are used (1) circuit switched transport and (2) packet switched transport. Congestion, which is a sudden state of this communication network where one or more nodes reach their capacity limit and as a result they either drop the incoming packet or buffer them for a later transmission, induces delay in the arrival of packets at the receiver. None of these effects of congestion are desirable for media transport hence counter measures should be taken to prevent the occurrence of congestion in communication network.

Congestion became an issue in the 1980's on TCP/IP networks as documented by Nagle [7]. Nagle proposed that no new data is sent until an acknowledgment is received for previous data and improved use of ICMP source quench.

Authors' address: Faculty of Computer Science and Information Technology, University Putra Malaysia, 43400 UPM, Serdang, Selangor, Malaysia.

During the 1980's TCP/IP links on the Internet became increasingly congested and Van Jacobsen [8] in 1988 proposed that if 'conservation of packets' was observed then TCP flows would generally be stable. The 'conservation of packets' was implemented by a congestion window where further packets would not be sent once the congestion window was full until another was removed. This congestion window could be dynamically re-sized as the connection was established and as conditions changed. These changes are widely credited with preventing ongoing TCP collapse.

Development work continues on TCP with congestion protocols such as Vegas [23], West-wood [9] and BIC [10] being produced. There are many stream of investigation into TCP congestion carrying on such as Paganini et. al. [11] who model congestion based on provable mathematical modeling.

Floyd and Fall [12] in a paper in 1999 discuss the main danger to the stability of the Internet is now undelivered packets particularly UDP. To overcome this a number of protocols such as XCP [13] and SCTP [14] have been proposed which provided stream based reliable transport. Real time media applications do not necessarily need a reliable transport and congestion will often be worse on a reliable transport due to retransmission.

A. TCP Congestion Control

TCP uses a form of end-to-end flow control. In TCP, when a sender send a packet, the receiver acknowledges receipt of the packet. A sending source can use the acknowledgement arrival rate as a measure of network congestion. When it successfully receives an acknowledgment, a sender knows that the packet reached its destination. The sender can then send new packets on the network. Both the sender and the receiver agree on a common **window size** for packet flow [6]. The window size represents the number of bytes that the source can send at a time. The window size varies according to the condition of traffic in the network to avoid congestion [4]. Generally, a file of size f with a total transfer time of Δ on a TCP connection results in a **TCP transfer throughput** denoted by **r** and obtained from equation (a)

$$\mathbf{r} = \mathbf{f} / \Delta \qquad \text{Equation (a)}$$

We can also derive the **bandwidth utilization,** ρ_u, assuming that the link bandwidth is **B**, by equation (b)

$$\rho_u = \mathbf{r} / \mathbf{B} \qquad \text{Equation (b)}$$

TCP has three congestion-control methods: **additive increase, slow start,** and **retransmit.** [5] [8] [11].

B. System Model

Chiu and Jain [3] have formulated the congestion avoidance problem as a resource management problem and proposed a distributed congestion avoidance mechanism named 'additive increase/multiplicative decrease' (AIMD). In their work, as a network model they use a "binary feedback" scheme with one bottleneck router [17]. As shown in Figure 1. It consists of a set of m users each of which send data in the network at a rate[2] w_i. The data send by each user are aggregated in a single bottleneck and the network checks whether the total amount of data send by users exceeds some network or bandwidth threshold X_{goal} (we can assume that X_{goal} is a value between the knee and the cliff and is a characteristic of the network). The system sends a binary feedback to each user telling whether the flows exceed the network threshold. The system response is 1 when bandwidth is available and 0 when bandwidth is exhausted.

The feedback sent by the network arrives at the same time to all users. The signal is the same to all users and they take the same action when the signal arrives. The next signal is not send until the users have responded to the previous signal. Such a system is called synchronous feedback system or simply synchronous system. The time elapsed between the arrival of two consecutive signals is discrete and the same after every signal arrival. This time is referred also as RTT.

The system behavior can be defined the following time units:

A step (or round-trip time – RTT) is the time elapsed between the arrival of two consecutive signals.

A cycle or epoch is the time elapsed between two consecutive congestion events (i.e., the time immediately after a system response 0 and ending at the next event of congestion when the system response is again 0).

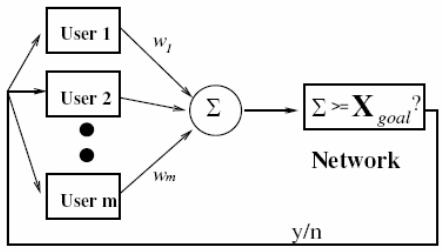

Figure 1: A control system model of m users sharing a network[3].

This network model is quite simple and its assumptions have been evaluated in the Internet for several years. In practice the parameter X_{goal} is the network capacity (i.e. the number of packets that the link and the routers' buffer can hold – or in-the-fly packets). When the aggregate flows' rate exceeds the network capacity the flows start to lose packets. If the transport protocol provides reliability mechanisms (e.g. as in TCP) it can detect the packet loss or congestion event. Since the majority of the applications use reliable transport protocols (e.g. TCP), the binary feedback mechanism has an implicit presence: a successful data transmission is interpreted as available bandwidth, and a packet loss is interpreted as congestion event [18].

Although the system had a strong impact on the evaluation of congestion avoidance mechanisms (e.g. AIMD), there are some limitations. First, the system considers the responses to be synchronous, which, in terms of real networks means that all flows have the same RTT. This assumption is not real. A second assumption and limitation is that the network response arrives at the same time to all users, even when they have the same RTT. This is disputed in [19]. The above assumption is supported by Jacobson experimentally in a low bandwidth network with congestion avoidance mechanisms (TCP-Tahoe) and where flows have the same RTT [20, 21]. Whatever the argument, this assumption is not true for a reason which is the third limitation of the system. The system has only one bottleneck. In reality a connection might go through none, one, or more than one router or bottlenecks. If a flow traverses more than one bottleneck, then it is not guaranteed that at each bottleneck congestion will happen at the same time. Nevertheless, these limitations do not prevent the mechanisms from controlling flows' data rate and avoid congestion which was the major concern in the early stages of the Internet [3].

C. Additive Increase / Multiplicative Decrease Control algorithm (AIMD)

The Additive Increase/Multiplicative Decrease (AIMD) algorithms described in detail in [3] and are referred as "dynamic window adjustment" in [4]. The basic idea of the algorithms to reduce the sending rate/window of the flows when the system bandwidth is exhausted and to increase the sending rates/windows when bandwidth is available. As mentioned in the previous section, when bandwidth is available (i.e. the aggregate rates of the flows do not exceed the network threshold: $\Sigma\ w_i < X_{goal}$) the system attaches the signal 1 to the acknowledgment of each packet. In response, flows increase by one (packet) their windows. A continuous series of positive signals will cause a linear increase in the flows' rate. Obviously, the increase is not unlimited because the bandwidth is fixed. When flows' rate exceed the bandwidth limit (i.e. $\Sigma\ w_i \geq X_{goal}$) the system attaches the 0 signal to the acknowledgment of each packet and flows respond to congestion by a decrease in their sending rates/windows.

A. Lahanas and V. Tsaoussidis in [3] prove that a linear increase/exponential decrease policy is a condition for the increase/decrease algorithms to set (or converge) quickly the system in a fair state where the load oscillates around some equilibrium. The equilibrium state determines also the fairness and efficiency of the mechanism.

The convergence behavior of a two flow AIMD system is depicted by vectors in a 2-dimensional space oscillating around the efficiency line (or equilibrium) in Figure 2. Upon each multiplicative decrease, the two windows x_1 and

x_2 move closer to the fairness line ($x_1 = x_2$). More details on the convergence of AIMD can be found in [3].

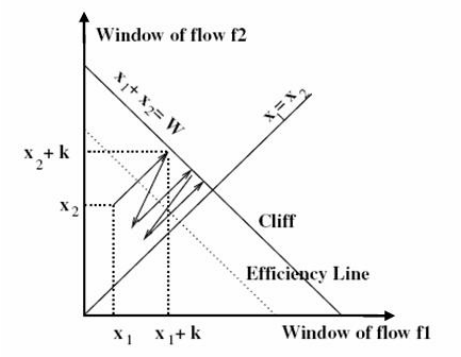

Figure 2: Vectorial representation of two-flow convergence to fairness. Figure is based on [3].

A point on the quadrant between axis represent the sum of the windows. The vectors trace the sum of the windows as they increase or decrease. 'k' denotes the length of the projection of the linear increase vector on the x and y axis. **W** is the value of X_{goal} in terms of packets.

AIMD is also designed to be responsive to fluctuations of bandwidth availability due to varying contention; this is managed by a continuous probing mechanism through additive increase of resource consumption. Chiu and Jain [1] showed that AIMD guarantees convergence to fairness: all flows eventually converge to a fair-share, i.e., an equal allocation of resources. Convergence to fairness is faster when the multiplicative decrease is larger, but then, bandwidth is further underutilized, and applications experience severe transmission gaps. Hence, although smoothness is desirable, it works against fairness: the smoother the adjustment, the longer convergence to fairness takes [16].

D. TCP Compatible AIMD Mechanisms

The TCP congestion control is classified as Additive-Increase Multiplicative-decrease (AIMD) mechanism. Following the notation AIMD(a, b) presented in [22], TCP is AIMD(1, 1/2). The parameter a represents the factor to be added to the congestion window each round trip time in absence of congestion, that is congestion_window + a.

On the other hand, the parameter b represents the complement to 1 that should be multiplied to the congestion window when congestion is detected, that is (1-b) congestion window.

In [22] Floyd, Handley and Padhye make a mathematical analysis to deduce other AIMD(a, b) mechanism that are compatible with TCP AIMD(1, 1/2) and at the same time reduce their sending rate less abruptly than TCP when a single packet is lost.

- **Compatibility:** Means that the mechanisms derived are able to compete fairly with TCP [22].
- **Efficiency:** It is average flows throughput per round trip time (RTT) when system is in equilibrium. System is said to be in equilibrium when each flow shares same window [3].
- **Smoothness:** It is magnitude of oscillations during decrease step [3].
- **Responsiveness:** It is number of RTTs required for the system to achieve equilibrium [3].
- **Fairness:** Every flow uses equal share of bandwidth [3].

II. A Pseudocode of New AIMD

Let us assume network capacity (Window size or X_{goal}) is *W*. For Simplicity let us assume we have two flows system f1 and f2. Initially let flows f1 and f2 contain x_1 and x_2 window respectively. With out loss of generality we assume that $x_1 < x_2$ and $x_1 + x_2 <$ W furthermore, we are assuming that system converges to 'fair' in 'm' cycle. In 1st cycle Pseudocode is given by total flow is:

$$x_1 + x_2 + 2k_1 \quad (1)$$

In AIMD is $x_1 + x_2 + 2k_1$

It is clear in 1st cycle that system has k_1+1 Round Trip Time (RTTs) or steps. Let $x_1 + x_2 + 2k_1 \geq W$ then there is Congestion and system gives 0 feedback. Now we will use decrease step. In 2nd cycle Pseudocode is given by total flow is:

$$\frac{x_1}{2} + \frac{x_2}{2} + 2k_1 + 2k_2 \quad (2)$$

In AIMD is $\frac{x_1}{2} + \frac{x_2}{2} + k_1 + 2k_2$

Obviously 2nd cycle contains $k_2 + 1$ RTT. Let $\frac{x_1}{2} + \frac{x_2}{2} + 2k_1 + 2k_2 \geq W$ then system gives 0 feedback. Obviously we will use decrease step. In 3rd cycle Pseudocode is given by total flow is:

$$\frac{x_1}{2^2} + \frac{x_2}{2^2} + 2k_1 + 2k_2 + 2k_3 \quad (3)$$

In AIMD is $\frac{x_1}{2^2} + \frac{x_2}{2^2} + k_1 + k_2 + 2k_3$

Here 3rd cycle contains $k_3 + 1$ RTTs.

Let $\frac{x_1}{2^2} + \frac{x_2}{2^2} + 2k_1 + 2k_2 + 2k_3 \geq W$ then system gives 0 feedback. Obviously we will use decrease step. Similarly at m^{th} cycle we have:

Total flow is $\frac{x_1}{2^{m-1}} + \frac{x_2}{2^{m-1}} + 2k_1 + 2k_2 \ldots 2k_m$ (4)

Suppose m^{th} cycle points to equilibrium that is all flows share fair allocation of resources.

The algorithmic approach when initial window size of 2 flows and Window size are x_1, x_2, W respectively, is given by:

AIMD (x_1, x_2, W)
{
$z = x_1 + x_2$ // z denotes used Capacity of Network.
$k = 1, t = 1$ // k denotes numbers of RTTs
while (1)
{
k = k + 1
$z = x_1 + x_2 + 2t$
t = t + 1
if(z >= W)
{
$x_1 = \frac{x_1}{2}$
$x_2 = \frac{x_2}{2}$
$z = x_1 + x_2 + 2t$
k = k + 1
}
}

Total number of packets in various cycles:
In 1st cycle, total number of packets is given by:

$$(1+k_2)(\frac{x_1}{2}+\frac{x_2}{2}+2k_1+k_2)$$

But from 1st cycle we have $x_1 + x_2 + 2k_1 = W$ Therefore $x_1 + x_2 + k_1 = W - k_1$

Thus total number of packet is given by $(1+k_1)(W-k_1)$.

In 2nd cycle, total number of packets is given by:

$$(\frac{x_1}{2}+\frac{x_2}{2}+2k_1+2k_2) = W$$

Therefore $\frac{x_1}{2}+\frac{x_2}{2}+2k_1+k_2 = W - k_2$

Thus total number of packets is given by: $(1+k_2)(W-k_2)$.

Similarly in 3rd cycle, total number of packets is given by: $(1+k_3)(W-k_3)$.

Similarly in m^{th} cycle, total number of packets is given by: $(1+k_m)(W-k_m)$.

Thus total number of packets in all cycles is given by: $(1+k_1)(W-k_1) + (1+k_2)(W-k_2) + (1+k_3)(W-k_3) + \ldots + (1+k_m)(W-k_m)$.

Relationship between RTTs in various cycles, from equation 1 and 2 we have $x_1 + x_2 + 2k_1 = \frac{x_1}{2}+\frac{x_2}{2}+2k_1+2k_2$

$$k_2 = \frac{1}{4}(x_1 + x_2)$$

But from equation 1 we have: $k_2 = (W - 2k_1)/4$

From equations 2 and 3 we have: $k_3 = \frac{1}{2}k_2$

From equation 3 and 4 we have: $k_4 = (\frac{1}{2^2})k_2$

Thus $k_m = (\frac{1}{2^{m-2}})k_2$ for m =3

A. Analysis

In this paper we interest to analysis for the factors of Congestion Control such as fairness, efficiency, responsiveness, and smoothness respectively.

- **Fairness:** One of the interesting properties of AIMD algorithm that we introduce in the paper is ability of a scheme to approach to fairness monotonically, i.e. the fairness during interval 'i' is given by $f_i = \frac{x_{1i}}{x_{2i}}, 0 \le f_i \le 1$, then the following conditions should be satisfied.

$\forall i : f_i + 1 \ge f_i$ and $\lim_{i\to\infty} f_i = 1$

Without loss of generality we are assuming that $x_2 = x_1 + n$. At the end of 1st cycle, fairness ratio is given by: $\frac{(x_1+k_1)}{(x_2+k_1)} = \frac{(x_1+k_1)}{(x_1+n+k_1)} = 1 - \frac{n}{(x_1+n+k_1)}$

Similarly at the end 2nd cycle, fairness ratio is given by $1-\frac{n}{2}\left(\frac{1}{(\frac{x_1}{2}+\frac{n}{2}+k_1+k_2)}\right)$. Clearly term $\frac{n}{2}\left(\frac{1}{(\frac{x_1}{2}+\frac{n}{2}+k_1+k_2)}\right)$ is smaller than $\frac{n}{(x_1+n+k_1)}$. Similarly we can find fairness ratio for remaining cycle.

According to these result we can say that our system converge to monotonic fairness. There is one interested question here how much cycles are required for fairness. We have following reasoning for it. Since every time both x_1 and x_2 are divided by 2 of its previous value and equal constant are added in both flows. Thus system can never reach equilibrium if we assume float arithmetic. In Integer arithmetic we are assuming that system reaches fairness in m cycle. It indicates that

$\frac{x_2}{2^{m-1}} + k_1 + k_2 \ldots k_m - \frac{x_1}{2^{m-1}} + k_1 + k_2 \ldots k_m \approx 1$,

$\frac{x_2}{2^{m-1}} + \frac{x_1}{2^{m-1}} \approx 1$, $\frac{x_1}{2^{m-1}} + \frac{n}{2^{m-1}} - \frac{x_1}{2^{m-1}} \approx 1$

$n \approx 2^{m-1}, m \approx 1 + \log(n)$. But in AIMD fairness is reflected as $1 + \log(x_2)$ [4].

Obviously convergence to fairness of New AIMD is faster than that of AIMD.

- **Responsiveness:** Numbers of RTTs required for equilibrium (Responsiveness) is measured as: $(1+k_1) + (1+k_2) \ldots (1+k_m) = m + (k_1 + k_2 + \ldots k_m) = m + (k_1 + \left(\frac{w-2k_1}{2}\right)\left(1-(\frac{1}{2})^{m-1}\right)$. In AIMD algorithm k is defined as $k_i = \frac{w}{4}$ for i >=2.

It means number of RTTs is fixed in each cycle. But in our approach $k_i = \frac{k_{i-1}}{2}$ for i >= 3. It means number of RTTs in each cycle are half of its previous cycle for i >=3. Obviously we have less number of RTTs.

- **Smoothness:** is reflected between i and i+1 cycle as:

$$\frac{x_1}{2^{i-1}} + \frac{x_2}{2^{i-1}} + 2k_1 + 2k_2 + \ldots 2k_i - \left(\frac{x_1}{2^i} + \frac{x_2}{2^i} + 2k_1 + 2k_2 + \ldots 2k_i\right)$$

$$= \frac{x_1}{2^i} + \frac{x_2}{2^i} = \frac{x_1 + x_2}{2^i}$$

Where $\left(\frac{x_1}{2^i} + \frac{x_2}{2^i} + 2k_1 + 2k_2 + \ldots 2k_i\right)$ is number of packets at the end of i^{th} cycle and $\left(\frac{x_1}{2^i} + \frac{x_2}{2^i} + 2k_1 + 2k_2 + \ldots 2k_i\right)$ is the number of packets at the beginning of $(i+1)^{th}$ cycle. System becomes smoother if i is increased.
It indicates that if numbers of cycle/RTTs (Responsiveness) are more, then smoothness becomes less. If Responsiveness is less then smoothness become more.

- **Efficiency:** The average efficiency is an interesting and important property of a Congestion Control system. It is desired that the system achieve higher efficiency. First of all we develop an expression for average efficiency of all cycles i.e. 1^{st} cycle to equilibrium cycle. We know that total numbers of packets in 1st cycle are $(1+k_1)(W-k_1)$. Since we have $(1+k_1)$ RTTs.

Now we are interested the total numbers of packets in all m cycles. This is measured as:

$$= (1+k_1)(W-k_1) + (1+k_2)(W-k_2) + \ldots$$
$$(1+k_m)(W-k_m)$$
$$= mW + (W-1)(k_1 + k_2 + \ldots k_m) - (k_1^2 + k_2^2 + \ldots k_m^2)$$

Solving this equation in term of k_1, we have:

$$= mW + (w-1)\left(k_1 + \frac{w-2k_1}{2}\right)\left(1 - \left(\frac{1}{2}\right)^{m-1}\right) - \left(k_1^2 + \frac{1}{12}(w-2k_1)^2\left(1 - \left(\frac{1}{4}\right)^{m-1}\right)\right)$$

Thus average throughput in all m cycle can be achieved dividing above equation by $W(k_1 + k_2 + \ldots k_m)$ We have:
Average throughput in the equilibrium cycle (efficiency) is given by: $1 - \left(\frac{W-2k_1}{4}\right)\left(\frac{1}{2^{m-2}W}\right)$

Example: Let the Network have W =600 and two users with initial loads of $x_1 = 10$ and $x_2 = 140$

Solution: Efficiency is given by $1 - \left(\frac{W-2k_1}{4}\right)\left(\frac{1}{2^{m-2}W}\right)$

Given W =600, $x_1 = 10$, $x_2 = 140$ $k_1 = \frac{W - x_1 - x_2}{2} = 225$

$m = 1 + \log(n) = 1 + \log(x_2 - x_1)$ (integer arithmetic)
Efficiency= 0.9973 or 99.73%

III. NCTUns Simulation Results

To evaluate the New AIMD algorithm, we conducted experiments based on NCTUns4.0 [24] simulation. The NCTUns4.0 simulation help us to evaluate the behavior of New AIMD under diverse network condition. In this section we focus on the simulation results.
Fig. 3 shows the network topology used in the simulation. The topology is a simple dumbbell topology network. The bottleneck link is set to 5 Mbps. The link that connect the senders and the receivers to the router have bandwidth of 5 Mbps. The end-to-end RTT is set to 30ms. The router queue size is 100 packets.
Fig. 4,5,6,7 shows the results between two nodes for experiments with 2,3,4,5 flows in the single bottleneck link respectively.
Table 1 shows the comparison between the throughput and average packet loss rates for different number of flow in same bottleneck link bandwidths. The throughput is calculated over a period of 300 seconds after the flow reaches steady state. Due to the large period for averaging the throughput and the buffer size of 100 packets, TCP flow seems to be able to obtain reasonable high throughput.

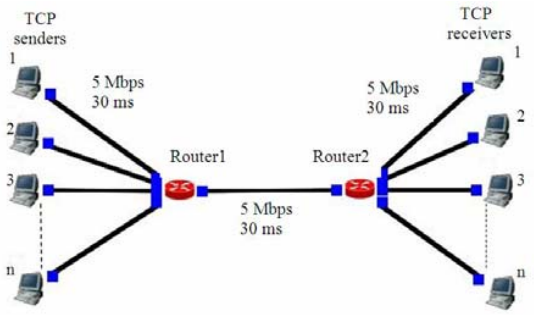

Fig. 3: Multiple flows experimental set-up for New AIMD evaluation.

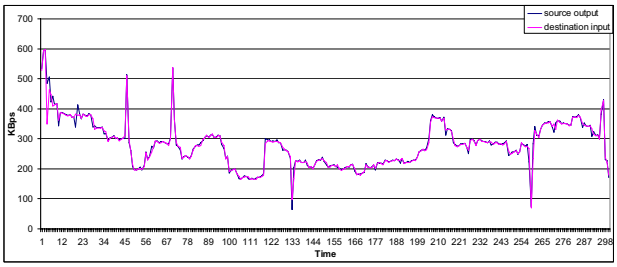

Fig. 4: The result between two nodes in experiment with 2 flows.

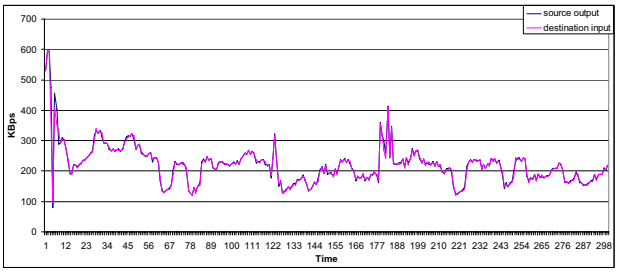

Fig. 5: The result between two nodes in experiment with 3 flows.

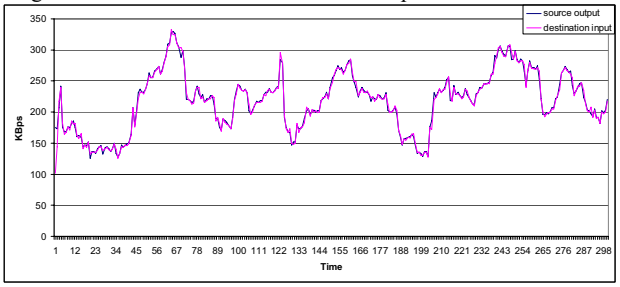

Fig. 6: The result between two nodes in experiment with 4 flows.

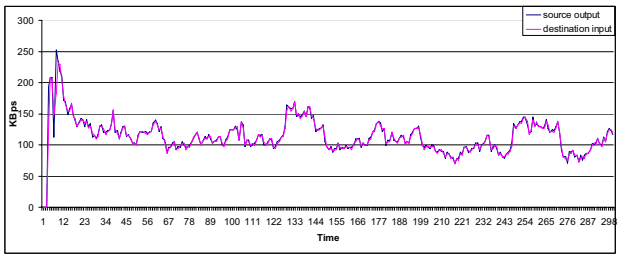

Fig. 7: The result between two nodes in experiment with 5 flows.

TABLE I
THROUGHPUT FOR DIFFERENT NUMBER OF FLOWS IN SINGLE BOTTLENECK LINK WITH NEW AIMD ALGORITHM

No. of flows	TCP		
	Total Throughput per-flow sent KB	Total Throughput per-flow received KB	Rate of Throughput received %
2	84477.76	84236.644	99.71
3	66250.012	65862.738	99.41
4	65368.328	65196.956	99.73
5	34138.592	33924.476	99.37

IV. CONCLUSION

In this paper we presented and evaluated a new algorithm of AIMD family of congestion management, called the New AIMD. It generalizes during increasing step x = x + k and on decreasing step x = x + k/2. It converges to fairness in 1+log (n) approximately. This is the best result in AIMD family. Responsiveness is reflected as very good because $k_i = \frac{k_{i-1}}{2}$ for i >= 3. It gives smoothness $\frac{x_1 + x_2}{2^i}$. Efficiency in equilibrium cycle is given by $1 - \left(\frac{W - 2k_1}{4}\right)\left(\frac{1}{2^{m-2}W}\right)$. From above numerical figure it gives more than 99% efficiency. It is compare to the (AIMD) [3].

REFERENCES

[1] D. Chiu and R. Jain. Analysis of the Increase/Decrease Algorithms for Congestion Avoidance in Computer Networks. Journal of Computer Networks and ISDN, 17(1), June 1989.
[2] A. Lahanas and V. Tsaoussidis. Additive Increase Multiplicative Decrease - Fast Convergence (AIMD-FC). In Proc. Networks 2002, Atlanta, Georgia.
[3] A. Lahanas and V. Tsaoussidis. Exploiting the Efficiency and Fairness Potential of AIMD-based Congestion Avoidance and Control. Journal of Computer Networks, 2003.
[4] Nader F. Mir, computer and communication networks, Prentice Hall, 2007.
[5] James F. Kurose, Keith W. Ross, computer networking, third edition, Addison Wesley, 2004.
[6] William Stallings, Data and computer communications, Prentice Hall, 2004.
[7] John Nagle. Congestion control in ip/tcp internetworks. SIGCOMM Comput. Commune. Rev., 14(4):11–17, October 1984.
[8] Van Jacobson. Congestion avoidance and control. In ACM SIGCOMM '88, pages 314–329, Stanford, CA, August 1988.
[9] Saverio Mascolo, Claudio Casetti, Mario Gerla, M. Y. Sanadidi, and Ren Wang. Tcp westwood: Bandwidth estimation for enhanced transport over wireless links. In Mobi-Com '01: Proceedings of the 7th annual international conference on Mobile computing and networking, pages 287–297, New York, NY, USA, 2001. ACM Press.
[10] Lisong Xu, Khaled Harfoush, and Injong Rhee. Binary increase congestion control (bic) for fast long-distance networks. In IEEE Infocom 2004, 2004.
[11] Fernando Paganini, Zhikui Wang, John C. Doyle, and Steven H. Low. Congestion control for high performance, stability, and fairness in general networks. IEEE/ACM Trans. Netw., 13(1):43–56, February 2005.
[12] Sally Floyd and Kevin Fall. Promoting the use of end-to-end congestion control in the internet. IEEE ACM Transactions on Networking, 7(4):458–472, 1999.
[13] D. Katabi, M. Handley, and C. Rohrs. Internet congestion control for future high bandwidth-delay product environments, 2002.
[14] R. Stewart, Q. Xie, K. Morneault, C. Sharp, H. Schwarzbauer, T. Taylor, T. Rytina, M. Kalla, L. Zhang, and V. Paxson. Stream control transmission protocol, 2000.
[15] Jean-Yves Le Boudec (EPFL Lausanne) "Rate adaptation, Congestion Control and Fairness: A Tutorial" Nov 2005.
[16] Comer, Douglas E. Internetworking with TCP/IP, 5E, Prentice Hall: Upper Saddle River, NJ. (2006).
[17] K. Ramakrishnan and R. Jain. A Binary Feedback Scheme for Congestion Avoidance in Computer Networks with a Connectionless Network Layer. ACM Transactions on Computer Systems, 8(2):158–181,May 1990.
[18] V. Jacobson. Congestion Avoidance and Control. In Proceedings of the ACM SIGCOMM '88, pages 314–329, August 1988.
[19] S. Shenker. A Theoretical Analysis of Feedback Flow Control. In ACM SIGCOM Symposium, September 1990.
[20] A. Tang, J. Wang, S. Hedge and S. H. Low. Equilibrium and fairness of networks shared by TCP Reno and Vegas/FAST. Telecommunication Systems, 30(4):417-439, December 2005.
[21] L. Wang, L. Cai, X. Liu and X. Shen. AIMD Congestion Control: Stability, TCP-friendliness, Delay Performance, Tech. Rep., Mar. 2006.
[22] Sally Floyd, Mark Handley, Jitendra Padhye. A Comparison of Equation-Based and AIMD Congestion Control; ACIRI; May 2000.
[23] Lawrence S. Brakmo and Larry L. Peterson. Tcp Vegas: End to end congestion avoidance on a global internet. IEEE Journal on Selected Areas in Communications, 13(8):1465-1480, 1995.
[24] S. Y. Wang, C. L. Chou, C. C. Lin. The design and implementation of the NCTUns network simulation engine. Science Direct, Simulation Modeling Practice and Theory, 2007, 57-81.