International Journal of Applied Engineering Research
ISSN 0973-4562 Volume 10, Number 3 (2015) pp. 5581-5596
© Research India Publications
http://www.ripublication.com

A Dynamic Parameter Estimation based Binary Increase Congestion control Algorithm for Data Transfer in Satellite Network

M.Nirmala
Assistant Professor Department of Computer Applications
Hindusthan college of Engineering & Technology Coimbatore, India
nilarangan@gmail.com

Dr.Ramachandra V Pujeri
Vice Principal MIT College of Engineering
Kothrud Pune India
sriramu.vp@gmail.com

Dr.M.L.Valarmathi
Associate Professor Department of Computer Science & Engineering
Government College of Technology Coimbatore
drmlv@gct.ac.in

Abstract

In the previous work [18], it was proved that the performance of bic-tcp was good under satellite network. In the previous work an round trip time(RTT) adaptive window increment based bic-tcp algorithm (BIC_AWI) for improved data transfer in satellite network has been proved. In the previous work [18] in the congestion avoidance of standard BIC, some of the parameters such as BICTCP_B are kept constant during, while finding a optimum TCP window size.
If this constant value is changed dynamically with respect to acceleration or deceleration of rtt, then considerable improvement has been realized. In this work, we proposed a dynamic parameter estimation based binary increase congestion control algorithm (BIC_DPE) for data transfer in satellite network. The best performance is obtained and also the same mechanism to minimize packet loss during terminal handover is implemented.
We, implemented the idea on ns2's implementation of bic tcp. Proposed modifications has been made on bic-tcp, and the performance of the improved protocol "BIC_DPE" has been compared with normal bic-tcp under Iridium satellite constellation. The performance of the algorithms has been measured using suitable metrics and the proposed BIC_DPE outperformed than normal bic-tcp.

Keywords—BIC_TCP Algorithm, Iridium Satellite Constellations, BIC_DPE protocol, Acceleration, deceleration, RTT

I. INTRODUCTION

The usage of Satellite Communications for internet traffic can be taken as an attractive proposal[1]. In the near future, the satellite component is believed to be integrated with terrestrial component [2][7] and the interaction of satellite and terrestrial connections can give rise to performance problems which are actually partially unexplored and are yet to be solved [3].

II. PROBLEM SPECIFICATION

In our previous study[18], we improved perforce bic-tcp and proposed a adaptive window increment based bic-tcp algorithm (bic_AWI) for improved data transfer in satellite network. The window size with respect to bic_AWI will be incremented with respect to the acceleration and deceleration of RTT was kept as a constant factor with the parameters like BICTCP_B. If this parameter is set dynmically then considerable improvement in performance is realized. In this work a dynamic parameter estimation based binary increase congestion control algorithm (BIC_DPE) for data transfer in satellite network has been proposed to provide the best performance. The minimization of packet loss during terminal handover is also added.

The Satellite Constellations [4] [7]

Satellite constellation is a collection of working together artificial satellites. In this paper the Iridium satellite constellation has been explored.

III. TCP CONGESTION CONTROL ALGORITHMS UNDER CONSIDERATION

In satellite networks, the available bandwidth is not used by the sender due to long propagation delay [9]; TCP was basically designed to be used with low link error rates networks which means all segment losses were due to network congestion. When there is a congestion the sender decreases the transmission rate at each time a segment loss is detected. It is the satellite networks, link error often occurs instead of network congestion and causes throughput degradation.

Binary Increase Congestion control for TCP (bic)[5]

As per paper [5], a new protocol has been designed by Lisong, Hasfoush and Rhee and it has satisfied various criteria such as scalability, Fairness in RTT and TCP friendliness. The present paper has also satisfied the above mentioned criteria. The size of the congestion window is the most important parameter for TCP flow. Window parameter determines the performance of TCP flow. A protocol already presented on paper [5] called as Binary increase congestion control (BIC) has been the major algorithm used in this paper.

The 2 major aspects of BIC are

1. Binary Search increase
2. Additive increase

Binary search increase allows bandwidth to drastically improve depending on the differences of current window and target window. When the differences are large bandwidth is more aggressive compared to small differences. Binary search increase when combined with additive increase, initially increases the window size linearly and then logarithmically.

The Midpoint algorithm calculation referred from [5] is used as a reference point in the complete below mentioned process.

The original form of BIC TCP algorithm [5]

The TCP module in NS2 has paved way to various congestion control algorithms and has helped various researchers to construct TCP in a network simulator such as NS2. The deviation in the implementation of TCP under NS2 from Linux has created greater problems such as simulation speed and coding in Linux is difficult [17].

The pseudo code of BIC TCP with respect to algorithm in fast recovery mode and not in recovery mode has been referenced from [5]. Refer to the algorithm for its working operations.

The preset parameters and the variables used for BIC TCP Algorithm are referred from [5] are used in this paper.

The Linux implementation of TCP BIC [6]

In the NS2 Linux implementation of TCP BIC various variables and parameters used are listed below. BICTCP_B is a constant under binary increase, The binary increase is reduced by a constant factor and tits name is smooth part . The variable linux_min_win is set when there occurs a loss in packet and it is set.

NS2's TCP Linux implementation of bictcp

The differences specified in the paper between Linux and NS2 [17] has been eliminated by creating a new design of NS2 TCP Linux. The NS2 TCP linux implementation uses TCP in NS2 with Linux Congestion control interface [17].

The boundary conditions and the algorithms used in the system implementation are referred from [6]

Based on the four conditions referred from [6], the value of the parameter BICTCP_B can be well understood. In the TCP Linux implementation standard version it is considered as 4, a constant value. The normal working of BIC algorithm has been explained using the flowchart **Fig. 1 : The Flow diagram Explaining Normal working of BIC TCP**

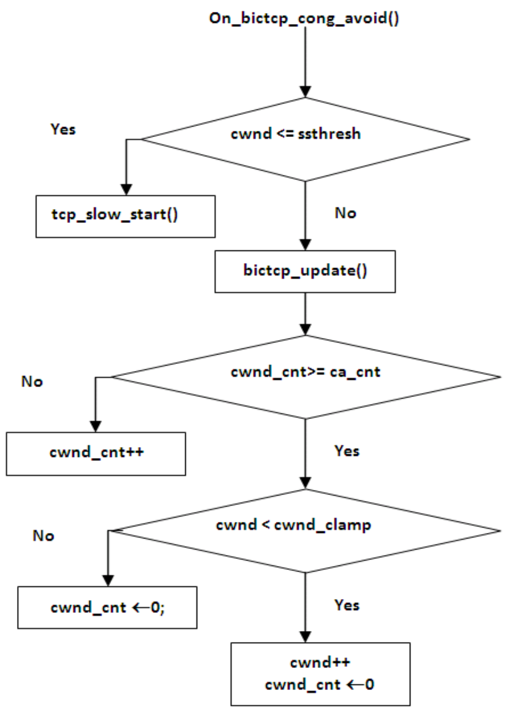

Fig. 1 : The Flow diagram Explaining Normal working of BIC TCP

The pseudo code of NS2's implementation of bictcp.

The following is the pseudo code is the congestion_avoid part of NS2's implementation of bictcp. This implementation is equivalent of the above explained Linux tcp bic implementation. The function "On_bictcp_cong_avoid" will be called during each successful acknowledgment and for each call of the function "On_bictcp_cong_avoid", the function "bictcp_update" will calculate ca_cnt according to the four boundary conditions mentioned in [6] .

```
bictcp_update(cwnd, bictcp_b)
{
  if( target_win − cwnd > bictcp_b)
            ca_cnt= (cwnd * bictcp_b)/ target_win;
   if(| target_win − cwnd | < bictcp_b)
           ca_cnt  =  (cwnd  *  Smooth_Part)/ bictcp_b;
  if( target_win − cwnd < bictcp_b)
   ca_cnt = (cwnd * (bictcp_b –1)) /(cwnd-target_win);
 return ca_cnt;
}
```

Fig. 2 The bictcp_update function

The following **Fig. 3 The On_bictcp_cong_avoid function** is the standard bic_tcp.

```
On_bictcp_cong_avoid( ){
On_bictcp_cong_avoid( )
{
if (!tcp_is_cwnd_limited())
        return;
if (cwnd <= ssthresh)
        cwnd ← tcp_slow_start();
else {
     ca_cnt      ←      bictcp_update(cwnd,
BICTCP_B);
     if (cwnd_cnt >= ca_cnt) {
        if (cwnd < cwnd_clamp) {    cwnd++;
        }
        cwnd_cnt ←0;
      } else   cwnd_cnt++;     }}
```

Fig. 3 The On_bictcp_cong_avoid function

Proposed Dynamic Parameter Estimation based BIC (BIC_DPE) Algorithm
The following function referred in
Fig. 4 The On_BIC_DPE Congestion Avoidance function is modified for the implementation of bic_AWI.

```
On_bic_DPE_tcp_cong_avoid( )
{
Increment = 0;
prev_rtt=0
diff_rtt=0;
now_rtt = rtt;
diff_seq_rtt=now_rtt - prev_rtt;
//find running average of rtt
prev_rtt=(now_rtt +prev_rtt)/2;
        if (UnderTermLinkHandoffFlag)  {
          //Under Terminal Handoff, do nothing
                return;
        }
        if (!tcp_is_cwnd_limited())
                return;
        if (cwnd <= ssthresh)
                cwnd ← tcp_slow_start();
        else {
```

```
        //set the decrement factor of BICTCP_B
//with respect to the acceleration of deceleration of
rtt
                if diff_rtt < 0
                        decB ← 1 ;
                Else
                        decB ← 0;
                bictcp_b ←  BICTCP_B – decB ;
        ca_cnt ← bictcp_update(cwnd, bictcp_b);
                if (cwnd_cnt >= ca_cnt) {
                        if (cwnd < cwnd_clamp)
                                cwnd++;
                        cwnd_cnt ←0;
                } else
                        cwnd_cnt++;
        }}
```

Fig. 4 The On_BIC_DPE Congestion Avoidance function

In the above modified function, during handoff the cwnd will not get increased. The flag UnderTermLinkHandoffFlag will be set true from another layer of the protocol stack during handoff.

The onTermHandoffTimer referred in
Fig. 5 The Timer Function Which Periodically Checks Handoff Event is a timer function which will be called periodically and check whether the connection on the terminal is going to handoff to another satellite or not. During the period of handoff, the UnderTermLinkHandoffFlag will be set as true.

```
void onTermHandoffTimer {
UnderTermLinkHandoffFlag =  IsUnderHandoff();
   if (UnderTermLinkHandoffFlag)  {
      printf("\n%f : Term link handoff");
   }
ScheuleonTermHandoffTimerAt(HandoffCheckInt
erval)}
```

Fig. 5 The Timer Function Which Periodically Checks Handoff Event

This terminal handoff flag will control the behavior of the tcp agent. So, during handoff, the congestion window size will not be increased any further (but if any loss occurs, the congestion window will get reduced from the loss event handler). In future works, we may try to set the cwnd_cnt and the cwnd as zero, exactly at the event of hand off, to avoid the risk of dropping packets during handoff.

IV. SIMULATION AND VISUALIZATION OF IRIDIUM SATELLITE CONSTELLATION

The satellite network constellation used in this paper is Iridium. The study on the basic explanation for Iridium satellite constellation has been taken from [16].

Parameters of Iridium Satellite Constellation[10]

The parameters of Iridium satellite network has been referenced from [10] [7].

Software used for Satellite Communications Simulation: [5595][12]

Network simulator (NS-2)

NS-2 is a discrete event network simulator. NS2 can be used for exact satellite network simulation with a detailed modeling of radio frequency characteristics.

Satellite Handoff Modelling in NS-2 [12]

NAM is the Network visualization tool used in NS2 and it nevertheless supports satellite network visualization. The other open source softwares used in this paper which supports visualization of satellite constellations are

- Sat-plot-scripts (perl scripts)
- SaVi (Satellite Constellations Visualization)
- Geomview

Sat-Plot-Scripts [13]

Perl scripts are used to visualize ns satellite constellation configurations, both as a complete snapshot at a given instant and the path a packet travels.

SaVi [14]

SaVi, the Satellite Visualization tool [14] , is a used for visualizing and animating the movement of satellites and their coverage.

Features of SaVi Includes :

- 3D visualization of satellites in orbit around the Earth
- Satellites footprints on earth's surface can be displayed
- Fraction of the Earth's surface covered by the constellation can be also computed.

Geomview [15]

Geomview is an open source Object Oriented Graphics Library used for geometry viewing.

The simulated Iridium Constellation and Terminals- Plotted using "Sat-plot-scripts" is shown in my previous paper [7]

The Iridium Constellation Map created using the "SaVi" tool has been shown in my previous paper [7]

V. SIMULATION AND ANALYSIS OF IRIDIUM SATELLITE CONSTELLATION

The TCP connection setup [8][7]

A part of the moving satellite constellation of dumbbell configuration has been referenced from [7].

The Important Common Satellite Network Parameters has been referenced from [10][7].

Important Traffic Parameters :[7]

Type	: Bic, Bic_AWI
PacketSize	: 1448
Initial Window Size	: 30000
TCP Sources	: 3
TCP Sink	: 3
Ground Terminals	: 2
Application	: FTP

Table 1 : Performance Metrics [7]

Contention Window(cwin) (Time vs cwin)	Better cwin results in better throughput and minimum EED
Round Trip Time (RTT)	Minimum RTT , better TCP performance
End to End Delay(EED)	Time taken for transfer and propagation
Delay / Packet Delay	Delay between 2 successive packets from sender to receiver
Jitter / Packet delay variation (PDV)	difference in EED between selected packets in a flow.
Packet Delivery Ratio (PDR)	Ratio between number of packets sent and received
Throughput	Number of packets arriving at the destination at a given time limit
Dropped Packets.	Number of packets failed to reach the destination while network transfer
Total Sent and Received Bytes	Number of data sent from the sender and received by the receiver

RESULTS AND DISCUSSIONS

The performance of the normal BIC congestion control algorithms and the improved BIC_DPE has been evaluated under Iridium satellite constellation. The below mentioned following bar and line graphs shows the comparative performance of the two algorithms under Iridium satellite constellation.

Contention Window Dynamics

The size of the contention window with respect to time is used as an evaluation parameter for both BIC and BIC_DPE congestion control algorithms. The simulation time is 300 seconds with a time gap of 0.5 seconds. The line graph **Fig. 6 The Contention Window Dynamics** depicts the graphical format of the evaluation metrics under the Iridium satellite constellations.

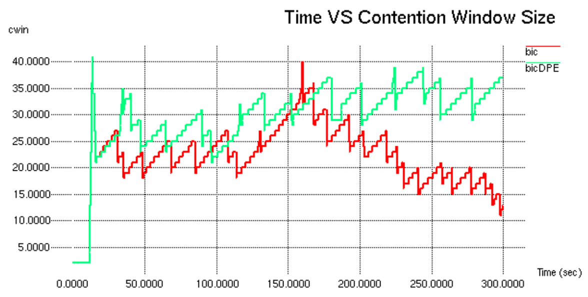

Fig. 6 The Contention Window Dynamics

Round Trip Time Estimation

Round Trip time is taken as a important parameter as it refers to the time taken for a signal to sent plus the length of the time for acknowledgement has to be considered. The graph

Fig. 7 The Round Trip Time Estimation shows the RTT of 2 congestion control algorithms.

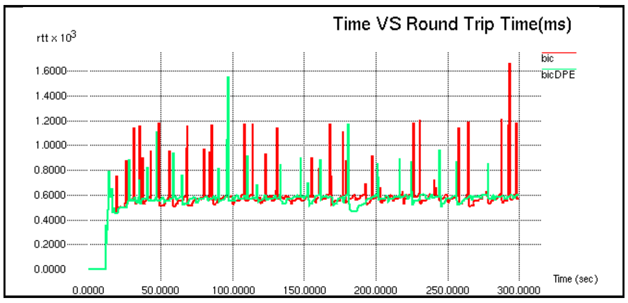

Fig. 7 The Round Trip Time Estimation

Number of Bytes Sent

The total number of bytes sent by the BIC is 17304936 and by the BIC_DPE is 20913618 is depicted in the below bar graph **Fig. 8 The Total Sent Bytes**

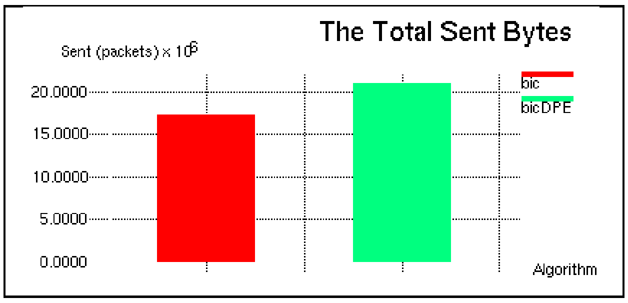

Fig. 8 The Total Sent Bytes

Number of Bytes Received

The total number of received bytes by the BIC is 17304936 and by the BIC_DPE is 20913618 is depicted in the below bar graph

Fig. 9 The Total Received Bytes.

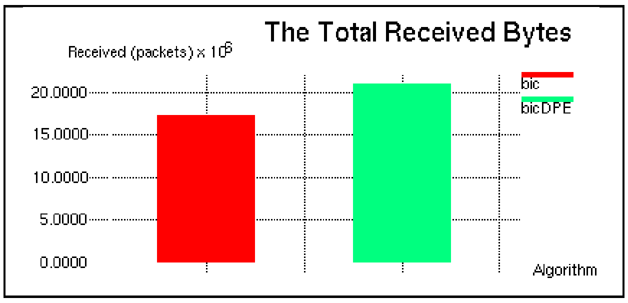

Fig. 9 The Total Received Bytes

Average Throughput

Throughput refers to the amount of bytes received by the destination. The average throughput is calculated by the throughput per unit of time by the formula (recvdSize/duration)*(8/1000). The average throughput for BIC is 461.46 kbps and for BIC_DPE is 557.70 kbps which shows that BIC_DPE throughput is improved with respect to BIC. The

Fig. 10 The Average Throughput shows an improved throughput with respect to BIC_DPE ALgorithm.

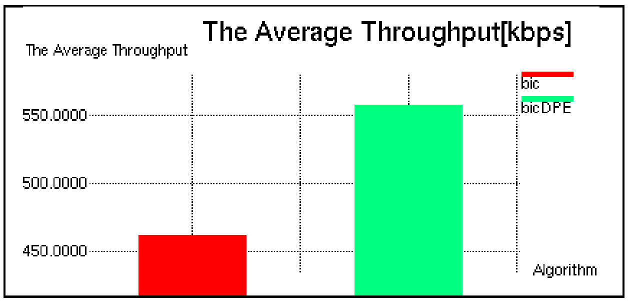

Fig. 10 The Average Throughput

Packet Delivery Ratio / Fraction

The ratio of the number of packets delivered to the destination with respect to the packets send by the source. The values are 99.38 and 99.49 with respect to BIC and BIC_DPE.

∑ Number of packet received / ∑ Number of packet send

The chart

Fig. 11 The Average PDF highlights, that there exist a slight variation with respect to BIC_DPE in the improvised angle.

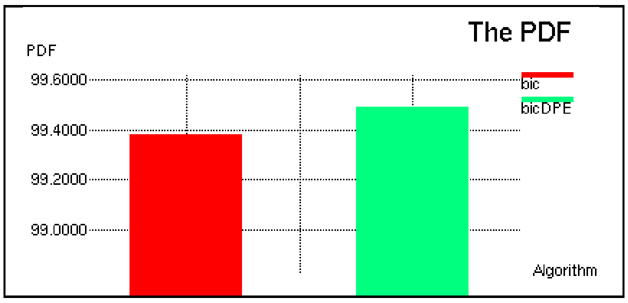

Fig. 11 The Average PDF

Total Dropped packets

The total dropped packets with respect to BIC is 181 and with respect to BIC_AWI is 195. The BIC_DPE algorithm drops just 14 packets higher than BIC and it is shown in

Fig. 12 The Total Dropped Packets

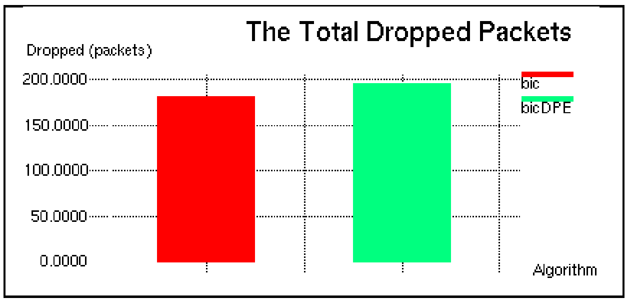

Fig. 12 The Total Dropped Packets

Delay

It is the difference between the receiving time and sending time. The AWK script processes the trace file and delay is calculated. Average delay is Total_Delay / total_packet_count

The average delay of BIC is 24.36 ms and BIC_DPE is 21.07 ms. The below depicted charts

Fig. 13 The Average Packet Delay &

Fig. 14 The Average Packet Delay shows average inter packet Delay over time. The performance in terms of delay in the case of proposed BIC_DPE is lower than the normal bic in most of the locations in this graph. That is why the proposed BIC_DPE was able to send and receive much packets and providing higher throughput than bic.

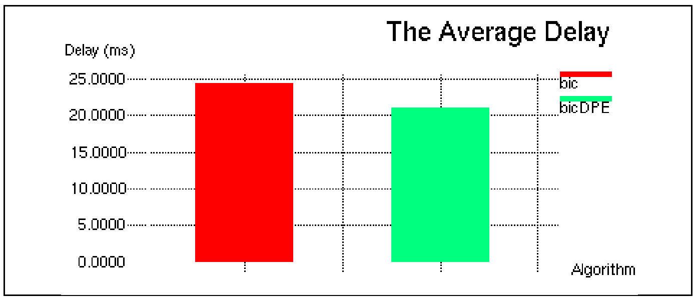

Fig. 13 The Average Packet Delay

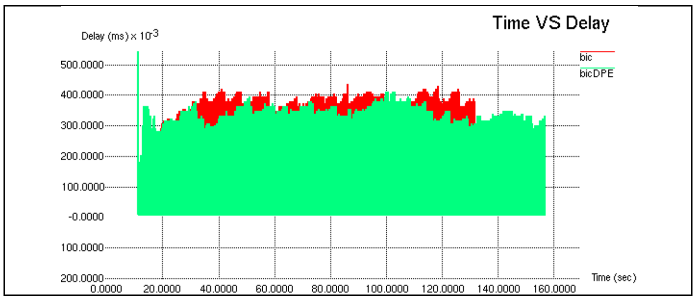

Fig. 14 The Average Packet Delay

The End to End Delay
The total time taken by the packet to reach the destination inclusive of route discovery delay, data queue is referred as end-to-end delay.

∑ (arrive time – send time) / ∑ Number of connections

The average EED of BIC is 441.30 ms and BIC_DPE is 452.57 ms

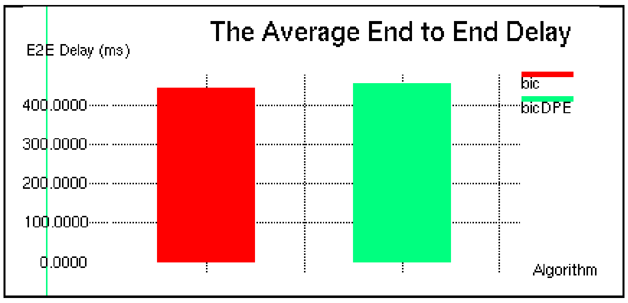

Fig. 15 The Average E2E

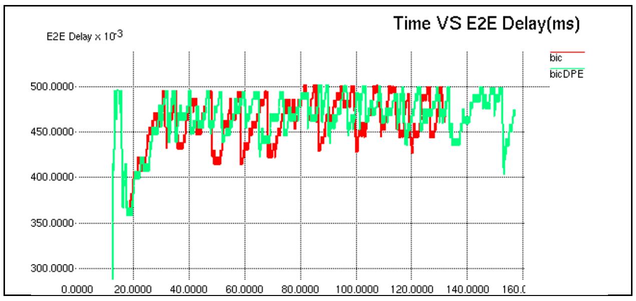

Fig. 16 The Time Vs E2E Graph

The charts

Fig. 15 The Average *E2E* &

Fig. 16 The Time Vs E2E *Graph* shows the average performance in terms of EED. With respect to end to end delay both the algorithms are almost same.

Jitter Analysis

Packet delay variation (PDV) is the difference in end-to-end delay between selected packets in a flow with any lost packets being ignored (RFC 3393). The effect is referred to as jitter1.

Jitter1 = jitter1 + e2eDelay - prev_e2eDelay. The average jitter1 of BIC is 7.63 ms jitter and BIC_DPE is 7.66 ms jitter

The following graphs

Fig. 17 The Time vs Jitter1(PDV*)* shows the Time vs Jitter1 (packet delay variation or PDV) and

Fig. 18 The Time vs Jitter2(rate of change of PDV*)* shows the analysis between time Vs jitter1 and jitter2

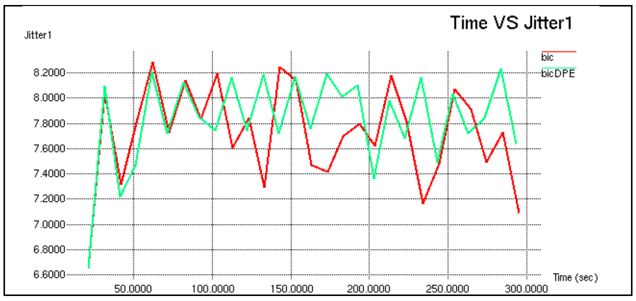

Fig. 17 The Time vs Jitter1(PDV)

Jitter difference in delay between selected packets in a flow with any lost packets being ignored. This effect is referred to as jitter2.

jitter2 += abs(delay- prev_delay)

average_jitter2 = jitter2*1000/rcvd_count

The average jitter2 value of 2 algorithms are 32.61 and 26.05 jitter respectively. If we carefully note the variation over time, it is obvious that the proposed BIC_DPE performed better and provided low Jitter1 and Jitter2

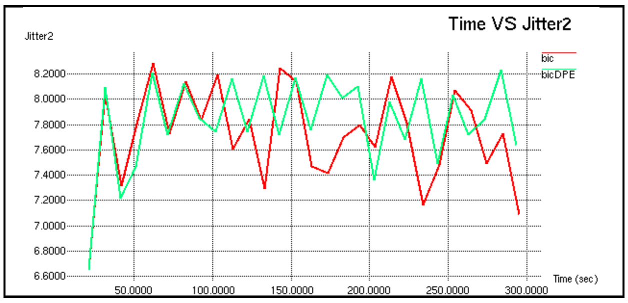

Fig. 18 **The Time vs Jitter2(rate of change of PDV)**

The following bar chart **Fig. 19 The Average Jitter1(PDV) and Jitter2(rate of change of PDV) Graph** shows the average jitter1 and jitter2. As shown in the charts, it is obvious that the proposed BIC_DPE performed better than bic.

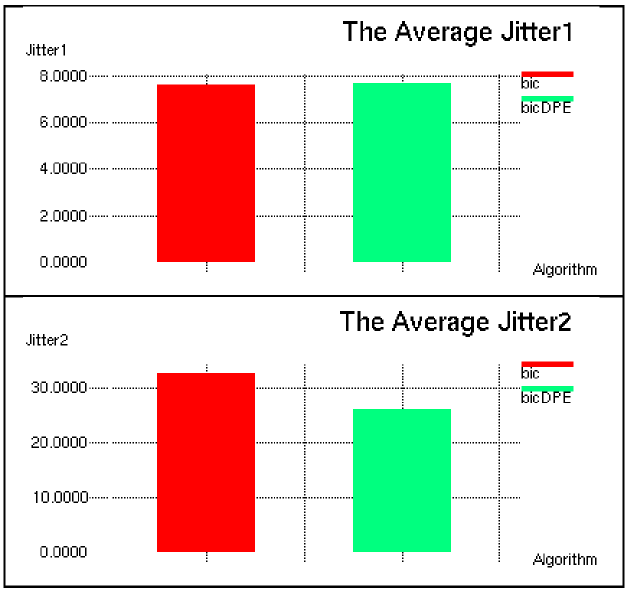

Fig. 19 The Average Jitter1(PDV) and Jitter2(rate of change of PDV) Graph

The above results shows that the proposed BIC_DPE performed very well under Iridium satellite constellation.

VI. CONCLUSION

According to the results arrived, the overall performance of the proposed BIC_DPE was good under Iridium constellations compared to BIC algorithm. If we see the cwin and rtt dynamics, we can realize the better performance of BIC_DPE. The performance in terms of delay in the case of proposed BIC_DPE was lower than the normal bic. That is why the proposed BIC_DPE was able to send and receive much data and providing higher throughput than normal bic.

In the design of the update method of BIC, BICTCP_B parameter is kept constant during the optimum TCP window size findings. In this model, that paramter has been dynamically changed with respect to acclerationa dndeceleration of RTT and the results had considerable improvements compared to BIC. Our future work will address this possibility and try to provide another improved congestion control algorithm for satellite networks.

VII. REFERENCES

[1] Zhang, Yongguang (Editor.), Internetworking and Computing over Satellite Networks, Hardcover, ISBN 978-1-4020-7424-0, Kluwer Academic Publishers.

[2] D.O’Mahony, UMTS: The fusion of fixed and mobile networking, IEEE internet computing, Vol 21 Jan – Feb 1998

[3] G.Neglia, V.Mancuso, F.Saitta, I.Tinnirello, A Simulation study of TCP performance over satellite channels, Department of Electrical engineering, University of Palerino Italy

[4] Hu, Y; Li, VOK Satellite-based Internet: a tutorial, IEEE Communications Magazine, 2001, v. 39 n. 3, p. 154-162 .

[5] Lisong Xu, Khaled Harfoush, and Injong Rhee, "Binary Increase Congestion Control for Fast Long-Distance Networks", Proceedings of IEEE INFOCOM 2004, pp. 2514-2524, HongKong, March, 2004

[6] Wu Hua and Gong Jian, Analysis of TCP BIC Congestion Control Implementation, CSSS '12: Proceedings of the 2012 International Conference on Computer Science and Service System.

[7] M.Nirmala,Ramachandra and V.Pujeri, "Performance of TCP Vegas, Bic and Reno Congestion Control Algorithms on Iridium Satellite Constellations", IJCNIS Vol.4, No.12, November 2012.

[8] Evaluation of TCP Congestion Control Algorithms on Different Satellite Constellations, 2013 International Conference on Advanced Computing and Communication Systems (ICACCS -2013), Dec. 19 – 21, 2013, Coimbatore, INDIA, [978-1-4799-3506-2/13/©2013 IEEE]

[9] Ian F. Akyildiz, Fellow, IEEE, Giacomo Morabito, and Sergio Palazzo, "TCP-Peach: A New Congestion Control Scheme for Satellite IP Networks", IEEE/ACM Transactions On Networking, Vol. 9, No. 3, June 2001.

[10] The ns Manual, Chapter 17 - Satellite Networking in ns, http://isi.edu/nsnam/ns/ns-documentation.html.

[11] Marian GRega, Lubomir Copjan, Stanislav Marchevsky, Stanislav Benco, Possibility Of Using Network Simulator (Ns-2) For Modeling Satellite Networks, Acta Electrotechnica et Informatica No. 4, Vol. 5, 2005

[12] Dr. Dharma P. Agrawal and Dr. Qing-An Zeng,, Satellite Systems, www.cs.gsu.edu/~cscyip/csc8221/Chapt-11.pdf

[13] http://personal.ee.surrey.ac.uk/Personal/L.Wood/ns/sat-plot-scripts/

[14] L. Wood, Examples of use of SaVi, list of papers and articles, page last retrieved May 2011. http://savi.sf.net/lw/examples.html

[15] Geomview 1.9.4, Sourceforge project site, August 2007, http://www.geomview.org/

[16] https://www.globalsatellitecommunications.com/iridium/network.html

[17] Wei, David X., and Pei Cao. "NS-2 TCP-Linux: an NS-2 TCP implementation with congestion control algorithms from Linux." Proceeding from the 2006 workshop on ns-2: the IP network simulator. ACM, 2006.

[18] An adaptive Window increment based BIC TCP algorithm for improved data transfer in Satellite Network (ICACCS - 2015), January 5 - 7, 2015, Coimbatore, INDIA.