Can DCCP Replace UDP in Changing Network Conditions?

Yao-Nan Lien, and Yu-Chi Ding
Department of Computer Science, National Chengchi University
Taipei, Taiwan, R.O.C.
{lien, g9714}@cs.nccu.edu.tw

ABSTRACT

DCCP is proposed to replace UDP for its ability of congestion control while maintaining its promptness by ignoring lost packets as UDP does. The network would suffer less congestion. However, whether the applications that switch from UDP to DCCP can maintain their needed performance or not is a big question. This paper investigates this problem by evaluating DCCP based VoIP vs. a variety of TCPs using NS-2 simulator. Our study shows that DCCP has a disadvantage in competing network bandwidth with existing TCPs. DCCP based VoIP may perform poorly in facing the competition of TCP traffics.

Keywords: DCCP, TCP, UDP, VoIP

I. INTRODUCTION

Almost every network application is trying hard to acquire a fair share of network bandwidth to perform its own task. On the other hand, network elements including routers and terminals (hosts) are working hard to prevent the network from being collapsed by congestion. The transport protocol resides on both ends of a connection is responsible for determining the most appropriate date rate to transmit data to the network. UDP (User Datagram Protocol) and TCP (Transmission Control Protocol) are the two most popular transport protocols used by most network applications. DCCP (Datagram Congestion Control Protocol) is proposed to replace UDP [4].

UDP is an unreliable connectionless protocol. Communication is achieved by transmitting data from source to destination without verifying the condition of network or the readiness of the receiver. Therefore, it is unreliable. When a message is sent, the sender does not know if the message will reach its destination. There is no concept of acknowledgment, retransmission or timeout. As a consequence, it has no way to know the effective bandwidth available from source to destination, neither the ability to adjust its own data rate to adapt to the network conditions. Therefore, the data rate is usually determined at the beginning of a session and doesn't change during the session. Real time applications often use UDP because dropping packets is preferable to waiting for delayed packets, which may not be an option for a real time application.

On the other hand, TCP is a reliable connection oriented protocol. Since packet transfer is not reliable, a technique known as positive acknowledgment with retransmission is used to guarantee the reliability of packet transfers. This fundamental technique requires receiver to respond with an acknowledgment message as it receives the data.

The congestion control within the TCP plays a critical role in adjusting data rate (i.e., congestion window size) to avoid congestion from happening. Acknowledgments for data sent, or lack of acknowledgments, are used by senders to infer network conditions between sender and receiver. Together with timers, TCP sender and receiver cooperatively manage the congestion control behavior of a data flow.

TCP protocol is executed at the terminal nodes of a network. It doesn't have real time inside information about the network. The only indicator of network condition to the TCP protocol is packet traveling time as well as success or failure of package delivery. Therefore, most TCP versions count on these indicators to estimate the available bandwidth over the path from sender to receiver and to adjust data rate accordingly. The accuracy and the promptness of bandwidth estimation depend on many factors such as the stability of network traffic and the length of the path. Not surprisingly, most TCP versions are suffering from some shortcomings, oscillating data rate up and down making network fluctuated. More importantly, different versions of TCP may acquire unfair shares of network bandwidth. The quicker a TCP responses to network congestion, the smaller the bandwidth it may acquire. This phenomenon forms a

1550-445X/11 $26.00 © 2011 IEEE
DOI 10.1109/AINA.2011.101

paradox that a well behaved TCP may have a disadvantage in competing network bandwidth. For this reason, TCP Vegas, which can adjust its data rate more accurately and promptly, doesn't gain a wide acceptance in the network world [2].

In view of network bandwidth competition, UDP becomes the number one target to blame because it doesn't adjust its data rate to accommodate to changing network conditions. This is tolerable as UDP sessions constitute only a small portion of network traffic. However, as more and more multimedia network applications, such as VoIP and video streaming services, proliferate on the network in recent years, the selfish behavior of UDP may not be tolerable any more. Therefore, DCCP, which is an unreliable transport protocol but built-in with a congestion control mechanism, is proposed to replace UDP [4].

The question remains unanswered is: Can DCCP really replace UDP? Is there any side effect? This paper tries to provide answer to these questions by NS-2 based experiments. We use VoIP as an example to investigate two problems. First, can VoIP streams that use DCCP to transport data gain a fair share of bandwidth in the existence of other TCPs? Secondly, can VoIP maintain its end-to-end delay time if DCCP is used?

II. DCCP, TCP, and VoIP

2.1 DCCP

The Datagram Congestion Control Protocol (DCCP) is a message-oriented Transport Layer protocol [4]. DCCP implements reliable connection setup, tear down, ECN, congestion control, and feature negotiation. It provides a congestion control mechanism at user's choice but without lost data retransmission. It does not provide reliable in-order delivery either.

Similar to TCP, a receiver of DCCP is required to response with acknowledgment packets upon receiving packets. Acknowledgments inform a sender whether its packets arrived, and whether they were ECN marked.

DCCP allows users to choose a congestion control mechanism. The selection is done by using Congestion Control ID (CCID) to indicate the choice of standardized congestion control mechanisms, with the connection's CCID being negotiated at connection initialization time.

2.2 TCP's Congestion Control

Most TCP versions use a network congestion avoidance algorithm that includes an additive-increase-multiplicative-decrease (AIMD) scheme, together with other schemes such as slow-start to achieve congestion avoidance [9]. TCP Tahoe and Reno are two typical examples [5].

TCP Tahoe and Reno

For each connection, the sender maintains a congestion window, limiting the total number of unacknowledged packets that may be in transit end-to-end. Upon receiving a packet, the receiver responses with a cumulative acknowledgement informing the sender the reception of the highest sequence number of the consecutive packets it had received. Upon receiving an acknowledgement packet, the sender sends out succeeding packets to keep congestion window full. Thus, acknowledgement is considered a mechanism to "clock" packet transmission. An acknowledgement packet, called *duplicate ACK*, marking a duplicated sequence number will be sent if receiver receives an out of order packet.

Sender uses a mechanism called *Slow Start* to increase the congestion window either after a connection is initialized or after a timeout [12]. It starts with a window of two times the maximum segment size (MSS). Although the initial rate is low, the rate of increase is very rapid: for every packet acknowledged, the congestion window increases by 1 MSS so that the congestion window effectively doubles for every round trip time (RTT).

When the congestion window exceeds a threshold, the algorithm enters a new state, called *Congestion Avoidance*, as well as to update the threshold. In the Congestion Avoidance phase, as long as a regular ACK is received, the congestion window is additively increased by one MSS. When a packet is lost, sender either receives a duplicate ACK or experiences a timeout if no ACK is received by a determined time limit. If it is a timeout, both protocols will reduce congestion window to 1 MSS, and reset to Slow Start state. On the other hand, Tahoe and Reno treats duplicate ACKs in different ways.

Tahoe treats triple duplicate ACKs the same as a timeout while Reno will only halve the congestion window and perform a "fast retransmit" to retransmit the lost packet right away without waiting for timeout, and enter a phase called *Fast Recovery*.

In Fast Recovery state, the sender waits for an acknowledgment of the entire transmit window before returning to Congestion Avoidance. If there is no acknowledgment, TCP Reno experiences a timeout and enters the Slow-Start state.

TCP Vegas

TCP Vegas measures round-trip delays for every packet in the transmit buffer and uses it to set the congestion window size [2]. In addition, TCP Vegas uses additive increases in the congestion window. Although the control of window size is more accurate than others, it was not widely deployed due to its disadvantage in competing bandwidth with other TCPs.

Other TCP

New Reno and SACK are developed by modifying Reno to improve its performance [5,6]. On special networks such as WiFi, the assumption of network congestion being the most likely reason of packet loss no longer holds. The congestion control mechanism built in TCP becomes a trouble maker interfering the operations of network applications. Many modifications were proposed to correct the problem.

2.3 DCCP's Congestion Control

CCID 2 and 3 are the two congestion control mechanisms that have currently been developed to support DCCP.

CCID 2: TCP-like Congestion Control

CCID 2 provides a TCP-like congestion control mechanism, including the corresponding abrupt rate changes and ability to take advantage of rapid fluctuations in available bandwidth [7]. CCID 2 acknowledgements use the Ack Vector option. Therefore, its congestion control algorithms follow those of SACK TCP.

CCID 3: TFRC Congestion Control

TFRC [8] congestion control does not use a congestion window. Instead, a TFRC sender controls its sending rate directly. Receiver feedbacks to the sender roughly once per round-trip time reporting the loss-event-rate it is currently observing. The sender uses this loss-event-rate to determine its sending rate; if no feedback is received for several round-trip times, the sender halves its sending rate.

Giving up sliding window style congestion control, TFRC responses to network condition much slower, which is preferable by VoIP. On the other hand, if VoIP module insists to keep a constant inter-packet interval, it must adjust its packet size to accommodate to the changing sending rate.

2.4 Fairness in Bandwidth Competition

Although most transport protocols are trying hard to reduce network congestion, they do not response to the network congestion in the same speed due to various reasons. As a consequence, some TCPs acquire bandwidth more aggressively than others creating a severe fairness problem. Even applications that use the same version of TCP may compete to each other unfairly since the promptness of congestion control depends on the end-to-end delay time between sender and receiver too [1,3].

Even worse, real time applications that use UDP usually do not adjust their data rate to accommodate to the changing network conditions. Once they determine the initial data rate, either by detecting network condition or by users' choice, they do not change it. When the constitution of network traffic is mostly TCP, it is tolerable because network bandwidth has been overly provisioned in the past decade and TCPs can adjust themselves well to prevent the network from being collapsed. However, this may no long be true in the future when network bandwidth is gradually corroded by more and more UDP based multimedia applications such as VoIP and IPTV. DCCP is then proposed to replace UDP to solve the problem. However, killing UDP doesn't solve fairness problem at all. Fairness remains a big problem of the network. Furthermore, real time applications that use DCCP may not perform well under an unfair, if not hostile, network arena.

2.5 VoIP Stream

VoIP is one of the most popular real time multimedia applications over Internet. The speech of a voice session is converted into a steady stream of packets. Following is a typical procedure.

An input voice stream is first digitalized into an 8000 samples/sec and 8 (or 16) bits/sample PCM stream, and then chopped into a stream of 20 or 30 ms frames. Each frame is then encoded into a smaller frame using a compression codec. Each frame is then packetized into an IP packet with UDP and IP headers and then sent to the network under the control of UDP.

The entire encoding latency is then at least 20 or 30 ms. In summary, a VoIP stream acts like a constant-bit-rate source, sending a fixed number of frames per second. Users are extremely sensitive to delay. The mouth-to-ear delay time must be controlled under a limit, usually, between 300 to 400 ms. Retransmissions of lost packets are often useless because the receiver may have passed the playback point before the retransmitted packet arrives. The allowed loss rate is usually lower than 5%. The exact limits are depending on users. The stringent time constraint makes the VoIP a very time sensitive application that almost no buffer is allowed in the sender. A packet must be sent out immediately after it is converted from user's speech. Based on this property, we anticipate that any congestion control that adjust inter-packet interval will not be acceptable by VoIP.

III. EXPERIMENTS

In this section, we use NS-2 Simulator [13] with Mattsson's DCCP module [10] to carry out some experiments to evaluate the performance of DCCP in the bandwidth competition arena. This evaluation is trying to answer the question we raised in this paper: whether or not DCCP can replace UDP to support real time applications such as VoIP. The investigation is done by analyzing two things. First, can DCCP acquire a fair share of bandwidth (or even the desired bandwidth) in facing the competition of TCP traffic? Secondly, can a VoIP service maintain its quality?

3.1 Configuration of Experiments

The experimental network model is a chain topology as shown in Fig. 1. The link in the middle is first set to 0.25Mbps bandwidth with a 50ms latency simulating an intercontinental link. Every other link has a delay time of 10ms and bandwidth of 10Mbps. Therefore, the bottleneck is on the middle link. The latency of the bottleneck link was then varied to simulate shorter and longer network connections.

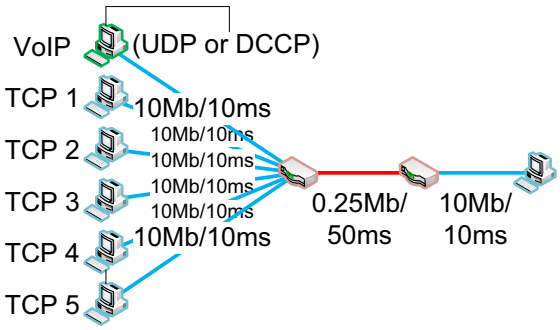


Fig. 1 Topology of Experiments

Two scenarios for each of UDP and DCCP (with CCID 3) were simulated. In the first scenario, an UDP (or DCCP) carries a VoIP stream into the network and a TCP stream was injected into the network every 10 seconds up to 5 TCP streams. The second scenario is the same as the first except that all 5 TCP streams were injected into the network at the beginning and a VoIP stream was injected at the 20th second and left at the 80th second. The experiment was repeated for 5 versions of TCP: Tahoe, Reno, NewReno, SACK, and Vegas.

The experimental parameters are summarized in Table 1. The DCCP CCID 3 throughput equation, which is the formula used by receiver to estimate effective throughput, is shown in (1) and its parameters are shown in Table 2.

Table 1 Experimental Parameters

Parameters	Value
VoIP Packet Size	80 Bytes
VoIP Inter-Packet Interval	30 ms
TCP Versions	Tahoe, Reno, NewReno, SACK, Vegas
TCP Packet Size	1460 bytes
Router Buffer Size	20 packets
Buffer Management Scheme	DropTail
Link Bandwidth	1.5~10 Mbps
Number of VoIP session	1
Number of TCP session	5

$$\langle = \frac{s}{R^* \sqrt{\frac{2^*b^*p}{\quad}} + (t\ RTO^*(3^* \sqrt{\frac{3^*b^*p}{\quad}} ^*p^*(1+32^*p^2)))} \qquad (1$$

Table 2 DCCP CCID 3 Parameters

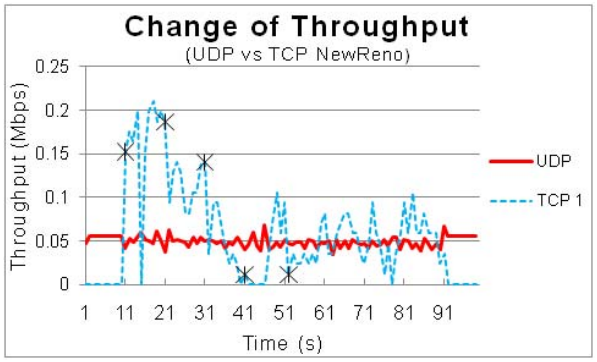
Parameters		Initial_Value
s	Packet Size	1460 Bytes
R	Round Trip Time	3 sec
P	Loss Event Rate	0-1.0
t_RTO	TCP Retransmission Timeout Value	3 sec
b	# of Packets Acknowledged by a Single TCP ACK	1

Simulation results show that varying the latency of bottleneck link doesn't affect the result significantly. Thus, only the results of 50ms latency are presented in the rest of this section.

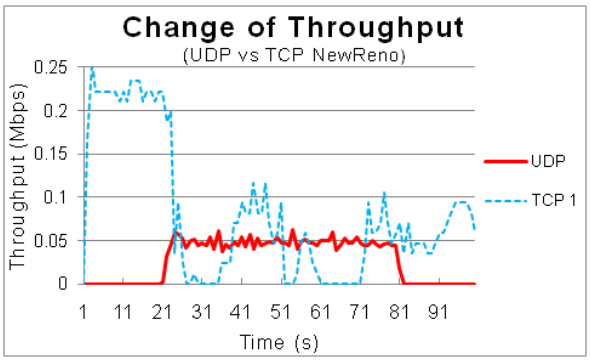
3.2 Bandwidth Competition

The throughput competition is discussed in this section. As we anticipate, UDP can maintain its throughput under the competition of TCP regardless whichever version. Fig. 2 shows the throughput of UDP and the first TCP NewReno stream. As more and more TCP was injected into the network, UDP was not affected, while TCP 1 was affected by each new injection. This demonstrates that TCP can adjust its data rate to accommodate to network congestion and UDP does not.

Fig. 3 shows the throughput of DCCP and the first TCP NewReno stream. As more and more TCP was injected into the network, both TCP 1 and DCCP UDP were affected. This demonstrates that DCCP can adjust its data rate to accommodate to network congestion too. However, as we can see from the figures, DCCP acquired much less bandwidth than TCP NewReno did. We can find similar phenomena in SACK and Tahoe. Even if there was only one TCP stream, the DCCP's throughput was reduced to approximately 30% of its original value.

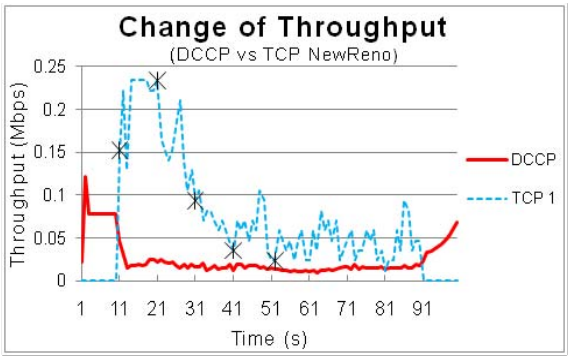


(a)

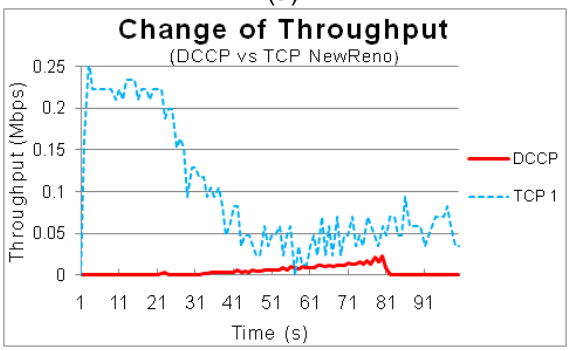


(b)

Fig. 2 Throughput Competition: UDP vs. NewReno
(a) UDP First (b) NewReno First

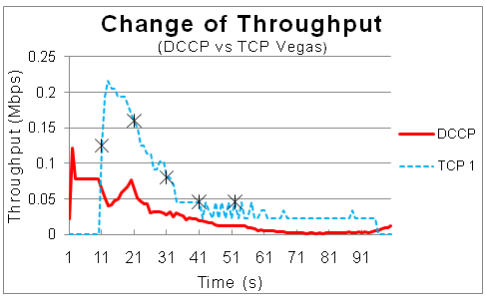


(a)

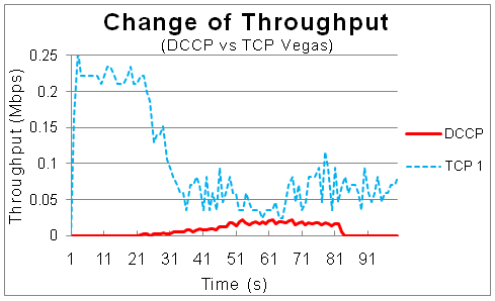


(b)

Fig. 3 Throughput Competition: DCCP vs. NewReno
(a) DCCP First (b) NewReno First



(a)



(b)

Fig. 4 Throughput Competition: DCCP vs. Vegas
(a) DCCP First (b) Vegas First

On the other hand, Vegas showed much more courtesy in competing network bandwidth if it was injected to the network after DCCP. As we can see from Fig. 4 where 5 TCP Vegas streams were injected into the network one by one and the throughput of DCCP based VoIP stream was not affected severely if the number of Vegas streams were no more than two. Although DCCP can live up with Vegas nicely, the application, VoIP, has to adjust its data rate (perhaps packet size) to maintain its quality.

In summary, DCCP has disadvantage in facing the competition of TCP NewReno, SACK, and Tahoe. The average throughput ratios of DCCP VoIP in the period 10-20s, 21-30s, 31-40s, 40-50s, and 50-60s are shown in Table 3, where throughput ratio is the ratio of the throughput at a certain time period to that of the 0-10s when no TCP existed.

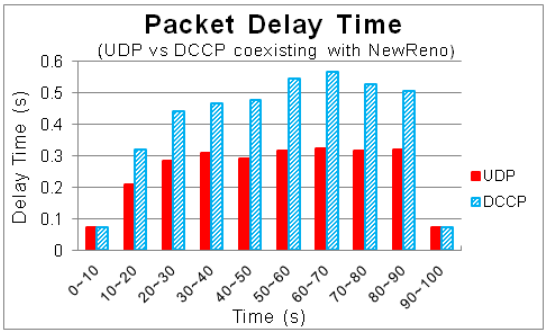
Table 3. Throughput Ratio of DCCP (%)
(DCCP runs First)

Period (sec)	10-20	20-30	30-40	40-50	50-60
vs. Tahoe	25.12	25.23	25.47	21.61	16.59
vs. Reno	29.67	32.01	23.01	16.94	20.21
vs. NewReno	29.79	25.82	20.68	21.26	15.89
vs. SACK	21.73	17.64	17.76	15.77	16.12
vs. Vegas	72.78	52.45	33.18	19.63	12.62

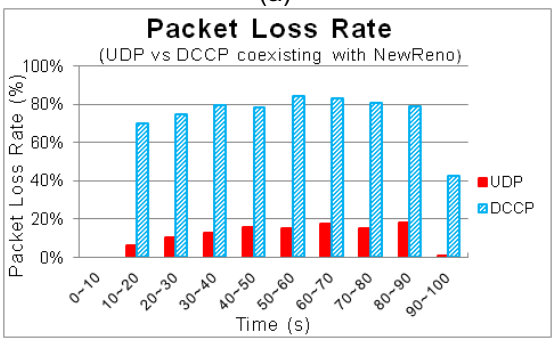
3.3 VoIP Quality

The quality of VoIP is discussed in this section. UDP based VoIP was not affected by TCP streams. However, due to the bandwidth infringement by TCP, the quality of the DCCP based VoIP stream was damaged every time a new TCP stream was injected into the network, as shown in Fig. 5 to 6. Average delay time and loss rate, which includes arrived but overdue packets, are also shown in Table 4. The delay time and packet loss rate of UDP based VoIP could be kept under 300ms and 20% respectively even if 5 TCP NewReno coexisted. On the other hand, DCCP based VoIP performed must worse. For each TCP stream injected into the network, delay time became longer and longer and packet loss rate became higher and higher. The loss rate could as high as 70% even there was only one TCP NewReno coexisting, which is not a surprise since the throughput ratio was reduced by 70%.

When facing the competition of TCP Vegas, the situation was little better. DCCP based VoIP could barely maintain its quality when there was only one TCP Vegas coexisting. The delay time could be kept under 200 ms and loss rate could be kept under 27%.

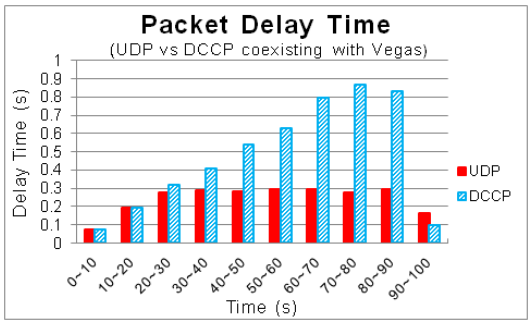


(a)

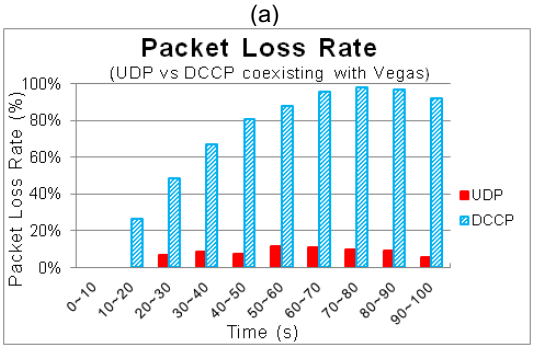


(b)

Fig. 5 Quality of VoIP: UDP and DCCP coexisting with NewReno (VoIP Runs 1st) (a) Average Delay Time (b) Average Loss Rate



(a)



(b)

Fig. 6 Quality of VoIP: UDP and DCCP coexisting with Vegas (VoIP Runs 1st) (a) Average Delay Time (b) Average Loss Rate

Table 4 Quality of VoIP (DCCP Based)

Period (sec)	0-10	10-20	20-30	30-40	40-50	50-60
vs. Tahoe						
Throughput Ratio(%)	100	25.12	25.23	25.47	21.61	16.59
Delay Time (ms)	73	395	415	424	470	548
Loss Rate	0	74.86	74.86	75.32	78.09	83.85
vs. Reno						
Throughput Ratio(%)	100	29.67	32.01	23.01	16.94	20.21
Delay Time (ms)	73	319	343	438	498	497
Loss Rate	0	70.13	68.40	77.97	82.81	79.82
vs. NewReno						
Throughput Ratio(%)	100	29.79	25.82	20.68	21.26	15.89
Delay Time (ms)	73	322	441	468	477	545
Loss Rate	0	70.01	74.97	79.58	78.32	84.54
vs. SACK						
Throughput Ratio(%)	100	21.73	17.64	17.76	15.77	16.12
Delay Time (ms)	73	443	490	513	536	554
Loss Rate	0	78.43	82.35	82.70	83.97	84.08
vs. Vegas						
Throughput Ratio(%)	100	72.78	52.45	33.18	19.63	12.62
Delay Time (ms)	73	194	321	412	543	627
Loss Rate	0	26.64	48.79	67.36	80.85	87.89

IV. RELATED WORK

Many papers propose the use of DCCP for VoIP instead of UDP. The research done by Nor, Hassan, and Almomani is the one most close to our study [11]. They did a similar study using both CCID 2 and CCID 3 congestion control mechanisms. They concluded that DCCP is more TCP-friendly as compared with UDP. Furthermore, they found that CCID-3 performs poorly in competing network bandwidth, which is the same as what we found in this study. However, they didn't study the quality of VoIP, in terms of delay time and packet loss. As we have known that mouth-to-ear delay of a VoIP session must be controlled under 300ms to 400ms. Our study shows that the quality of VoIP that uses DCCP is way below demanded criteria. Further studies are needed to evaluate other time sensitive applications over DCCP in order to make replacement decision.

V. CONCLUDING REMARKS

Whether or not DCCP can replace UDP is a big question. This paper investigates this problem by evaluating DCCP based VoIP against various TCPs using NS-2 simulator. Our study shows that DCCP has a disadvantage in competing network bandwidth with existing popular TCPs. As a result, a DCCP based VoIP cannot maintain its quality even under the competition of multiple streams of TCP Vegas. Although our experiment cannot represent all the situations, it represents a very typical intercontinental VoIP environment, which is the most critical service many VoIP operators intent to offer.

VoIP prefers constant packet rate to variable one. It requires application's cooperation to change packet size in order to change effective data rate. The cooperation between a VoIP service and DCCP in adjusting its effective data rate remains a research issue.

In conclusion, designing a transport layer protocol with congestion control for real time network services is much more difficult than that for non-real-time services. It must take timing factor into account, in addition to the average throughput. The cooperation of applications may be essential in adjusting packet size.

REFERENCES

[1] C. Albuquerque, B.J.Vickers, and T. Suda, “Network border patrol: Preventing congestion collapse and promoting fairness in the Internet”, *IEEE-ACM Transactions on Networking*, vol. 12, no. 4, Dec. 2004, pp. 173-186.

[2] L. S. Brakmo, S. W. O’Malley, and Larry L. Peterson. "TCP Vegas: New Techniques for Congestion Detection and Avoidance, " *Proc. Of ACM SIGCOMM*, pp. 24-35, Aug. 1994.

[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.1, pp. 1-14, 1989.

[4] Eddie Kohler, Mark Handley, and Sally Floyd, "Designing DCCP: Congestion Control Without Reliability," *SIGCOMM’06,* Sep., 2006, Pisa, Italy, pp. 27-38.

[5] K. Fall and S. Floyd, "Simulation-based Comparisons of Tahoe, Reno, and SACK TCP," *ACM Computer Communication Review*, vol. 26, no.3, pp. 5-21, 1996.

[6] S. Floyd and T. Henderson, "The NewReno Modification to TCP's Fast Recovery Algorithm," *IETF RFC 2582*, 1999.

[7] S. Floyd and E. Kohler, “Profile for DCCP Congestion Control ID 2: TCP-like Congestion Control”, *IETF draft-ietf-dccp-ccid2-08*, http://www.ietf.org/internet-drafts/draft-ietfdccp-ccid2-10.txt, Mar. 2005.

[8] S. Floyd, E. Kohler, and J. Padhye, “Profile for DCCP Congestion Control ID 3: TFRC Congestion Control”, *IETF draft-ietf-dccp-ccid3-11*, http://www.ietf.org/internetdrafts/draft-ietf-dccp-ccid3-11.txt, Mar. 2005.

[9] V. Jacobson, "Congestion Avoidance and Control," *Proc. of ACM SIGCOMM*, pp. 314-329, Aug. 1988.

[10] N. E. Mattsson, “A DCCP module for ns-2”, http://epubl.luth.se/1402-1617/2004/175/LTU-EX-04175-SE.pdf, Sep. 2004.

[11] S. A. Nor, S. Hassan, O. Almomani, "Simulated Performance of VoIP using DCCP CCID2 over Large Delay Link Networks," *Proc. Of Int’l Conf. on Network Applications, Protocols and Services*, Nov, 2008.

[12] W. Stevens, "TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms," *IETF RFC 2001*, 1997.

[13] "The Network Simulator - ns-2", http://www.isi.edu/nsnam/ns/.