QTCP: An Approach for Exploring Inter and Intra Protocols Fairness

Barkatullah Qureshi, Mohamed Othman*, Shamala K. Subramaniam, and Nor Asila Wati

Department of Computer Science and Information Technology, Universiti Putra Malaysia, 43400 UPM, Serdang, Selangor D.E., Malaysia
qureshi5939@gmail.com, mothman@fsktm.upm.edu.my

Abstract. TCP provides communication service however the network congestion is one of the core issues, it occurs when a link carry so much data, and effects on delay, packet loss rate and fairness. When the available bandwidth becomes very high, TCP, as tuned today, does not perform well. Several high-speed variants of TCP have been implemented to resolve the congestion issue , such as High-Speed TCP (HSTCP), Reno and New Reno TCP, Scalable TCP (STCP), Binary Increase Congestion Control TCP (BIC TCP), CUBIC TCP, Hamilton TCP (HTCP), Compound TCP and TCP Illinois. In this research a new Quick Transport Control Protocol (QTCP) has been designed to overcome the congestion issue. This is a new congestion control mechanism based on the modifications in the slow start phase of HSTCP and AIMD. QTCP performance evaluated by the experiment and compares the QTCP average throughput ratio, bandwidth link utilization and packet loss rate in inter and intra RTTs with inter protocols. We evaluated how the link can be fairly shared and achieve end-to-end throughput for each flow. Therefore a series of experiments conducted for the evaluation of multiple high-speed flows in different configurations. The results based on NS-2 simulator shows that QTCP provides better performance compared to other transport control protocols.

Keywords: Congestion Control, Fairness, Average Throughput, Bandwidth link utilization, Packet loss rate.

1 Introduction

The Sharing of data on the Internet and other similar networks is controlled by a suite of Internet protocols also known as Transmission Control Protocol/Internet Protocol (TCP/IP). TCP, being one of the protocols of the transport layer of the Internet Protocol Suite is also considered as one of its core protocols and handles most of the Internet data. Traditionally, while handling the data, TCP's congestion control performs well in most cases where the bandwidth is not extremely high, it shares the available bandwidth and offers fairness to millions of Internet users. However, when the available bandwidth becomes very high, TCP, as tuned today, does not perform well. This is because in

* The author is also an associate researcher at the Lab of Computational Science and Informatics, Institute of Mathematical Research (INSPEM), University Putra Malaysia.

A. Selamat et al. (Eds.): ACIIDS 2013, Part I, LNAI 7802, pp. 285–294, 2013.

the congestion avoidance phase, TCP takes a long time to increase the window size and cannot fully utilize the available bandwidth [1] and [2].Thus, the doors for the investigation of an optimal solution to this problem are still open and, consequently, in this article we propose our new approach, named the Quick Transport Control Protocol (QTCP) based on the modification of HSTCP. The QTCP takes a completely different approach in terms of congestion control over high-speed networks, and, hence, achieves improved throughput performance and fairness compared to existing high-speed TCP variants. The comparison of results based on the Network Simulator (NS-2) with several experimental configurations.

2 Related Work

To ameliorate congestion control issue, several high-speed variants of TCP have been implemented, such as High-Speed TCP (HSTCP) [3], Reno and New Reno TCP [4] and [5], Scalable TCP (STCP) [6], Binary Increase Congestion Control TCP (BIC TCP) [7], CUBIC TCP [8], Hamilton TCP (HTCP) [9], Compound TCP [10] and TCP Illinois [11]. FAST TCP is a delay-based approach and uses the RTTs for congestion measure; its throughput performance is significantly affected by the reverse traffic. Although the implementation and evaluation of all of these variants of TCP are available, none of these has yet been accepted as a single gold standard owing to the relative pros and cons of each variant [12].Therefore a new Quick Transport Control Protocol (QTCP) has been developed in this study.

3 Methodology

There is a plethora of algorithms aimed at the optimization of the standard TCP in terms of its congestion control mechanism to tune the requirements of applications that share large data over high-speed networks. Each of these algorithms has its own specific set of pros and cons and; to date, there is no single algorithm with the best optimal performance in all scenarios. This situation has led us to the development of yet another algorithm named the Quick Transport Control Protocol (QTCP). This algorithm implements a new congestion control mechanism based on the modifications in the slow start phase of HSTCP and AIMD. The modification is done in such a way that it provides significantly improved convenience, throughput, fairness and efficiency compared to most of the existing popular algorithms in the community. The parameters used in QTCP described in Table 1.

The QTCP growth function starts increasing the *cwnd* the same as other TCP schemes. The main function of QTCP is defined in equation (1), which has one parameter t, which represents the current time, or, we can say it represents the lifetime of a flow. QTCP depends on the parameter t that is being used in growth functions to compute the value used to increase the packet sending rate per RTT. In line one of the equation (1), it checks for the relation between *cwnd* and *ssthresh*; once the expression is satisfied it increases *cwnd* by 1, and the first packet will be sent to its destination. On successful acknowledgement of the sent packet, the process of increasing the sending rate per RTT and controlling the congestion window will be started. Hence, it will go to the second line, which increases

Table 1. Notation and Definition used in the QTCP Algorithm

Field	Description	Value
cwnd	Congestion window size.	-
ssthresh	Threshold value in TCP	-
t	Current/Elapsed Time	-
δ	Maximum increment in sending per RTT during α part.	3 seconds
Δ	Maximum increment in slow start	30 seconds
inc	A variable that holds the value, used to increase sending rate per RTT	-
incα	Increment in *inc*α in Slow Start.	4
incβ	Increment in Congestion Avoidance .	1
α	Life of a flow till second decrease in α part	
low_window	Low Window Parameter in Slow Start Phase.	-
updateInterval	Update Time Interval for *inc*α	0.5 seconds
maxInc	Maximum Increase *cwnd* size in Congestion Avoidance Phase	10
$mode_{CA}$	A flag variable that switches QTCP to Congestion Avoidance mode of *cwnd* in Slow Start.	-
tq_lastupdate	Last Update Time.	-
decrease_factor	Window Decreasing Parameter when congestion happens	0.8

the value of *cwnd* linearly to increase the packet sending rate per RTT. The variable *inc* holds the value that is used to increase the packet sending rate per RTT. While the value of *inc* is computed by the $Qupdate(t)$ function defined in equation (2).

$$cwnd = Qtcp(t) = \begin{cases} cwnd + 1, & if\ cwnd < ssthresh \\ cwnd + inc = Qupdate(t), & otherwise \end{cases} \tag{1}$$

The function $Qupdate(t)$ defined in equation (2), which has one parameter inherited from the calling function $Qtcp(t)$, in line one of the function, checks the relation between *cwnd* and *low_window* (that is set to its default value 38). If the expression is satisfied, QTCP will exponentially increase the packet sending rate on each successful acknowledgement until the *cwnd* reaches to *low_window*. The exponential function $cwnd = 1/cwnd$.

$$cwnd = Qupdate(t) = \begin{cases} \frac{1}{cwnd}\ , & \text{if } cwnd < low\_window \\ Qincrease(t), & \text{otherwise} \end{cases} \tag{2}$$

Once *cwnd* hits the *low_window*, QTCP will start using the $Qincrease(t)$ function, which is defined in equation (3); again it has one parameter t, which is used to compute the value that increases the sending rate per RTT, as shown in Figure 1.

$$Qincrease(t) = \begin{cases} \alpha increase(t), & \text{if } mode_{CA} = FALSE \\ \beta increase(t), & \text{otherwise} \end{cases} \tag{3}$$

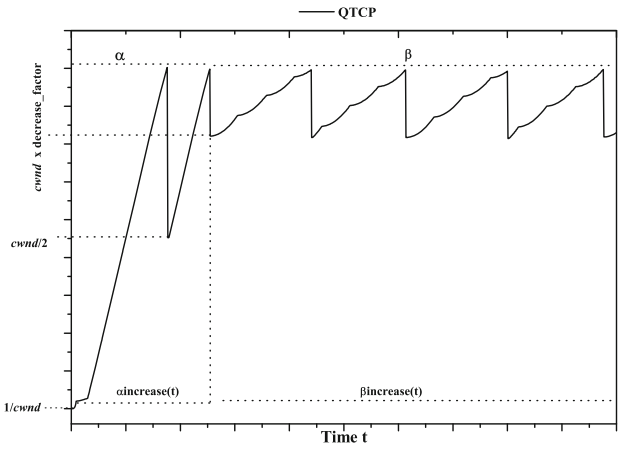

Fig. 1. Behaviour of the QTCP in slow-start and AIMD phase

QTCP divides the life of a flow into two parts, α or slow start: In this part *cwnd* must be increased faster with a certain speed so we can achieve the desired link utilization with minimum packet loss rate. β or congestion avoidance: In this part, the *cwnd* must be increased in such a way that promotes stability, utilization, fairness and minimum loss rate. Here in equation (3), $Qincrease(t)$ has two components; the first component $\alpha increase(t)$ is used to compute the value to increase the packet sending rate in α part. While $\beta increase(t)$ is used to compute the value to increase the packet sending rate in β or Congestion Avoidance part. On the α part we need to utilize the link capacity as much as we can; we need to achieve high link utilization with minimum packet loss rate. We designed a window growth function defined in equation (5), illustrated in Figure 2. It has one parameter t current time of a flow, in line one it checks the relation between t and Δ, which is set to its default value 3. If the expression is satisfied, it will return value of t as an increment in sending rate per RTT, until the value of t hits Δ. Once t hits the value of Δ, it will check the relation between *inc* and δ that is set to its default value 30. If the expression is satisfied, it will increase the value of *inc* by adding the value of $inc\alpha$ (that is set to its default value 4) to it, until *inc* hits the value of δ. Hence, although it will stop updating the value of *inc*, QTCP will continue to increase the sending rate by *inc*. The reason behind stopping the updating value of *inc* is to increase the stability and reduce the loss rate, as it will moderately increase the packet sending rate. In line three each time function saves current time t as a last update time $tq\_lastlupdate$ of *inc*.

QTCP keeps updating the sending rate until *cwnd* hits the maximum window size. Once it hits its maximum window size, the link will be congested and it will receive duplicate acknowledgements, which indicates that packets are being lost. For reducing congestion, QTCP decreases the window by half of *cwnd*, and a new threshold value will be set. This can be seen as follows in equation (4):

$$ssthresh = cwnd = cwnd/2 \tag{4}$$

Once the congestion is reduced, it stops receiving duplication acknowledgements; hence, QTCP will restart increasing the packet sending rate on each successful acknowledgement. The value of *inc* would not be updated, but its value is used to update the *cwnd* for increasing the packet sending rate, until it hits its maximum window size. Once it reaches the maximum window size, again the window will be decreased by the $decrease\_factor$, which is set to its default value 0.8, to reduce link congestion, and this time a flag variable $mode_{CA}$ will be set to true.

$$\alpha increase(t) = \begin{cases} t, & \text{if } t < \Delta \\ inc + inc\,\alpha, & \text{if } inc < \delta \\ tq\_lastupdate = t \end{cases} \tag{5}$$

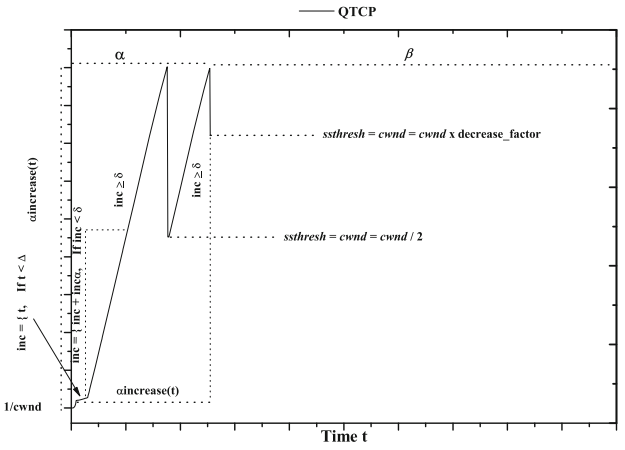

Fig. 2. The window growth function of QTCP α in slow-start phase

The flag variable $mode_{CA}$ switches QTCP to congestion avoidance phase. In congestion avoidance or β, the value for increasing packet sending rate per RTT, will be computed by β $increase(t)$, defined in equation (6), as shown in Figure 3.

$$\beta increase(t) = \begin{cases} inc + inc\,\beta, tq\_lastupdate = t, & \text{if } t - tq\_lastupdate \geq updateInterval \\ 1, & \text{if } inc > maxInc \end{cases} \tag{6}$$

In line one of $\beta increase(t)$ function, it computes the new value for increasing the sending rate, by adding *inc* and $inc\beta$ (which is set to its default value 1 after a certain interval called $updateInterval$ which is set to its default value 0.5 seconds. After every 0.5 seconds $\beta increase(t)$ updates the value of *inc* by adding $inc\beta$ to it. This gives a new value to the increased sending rate. While QTCP keeps increasing the packet sending rate on

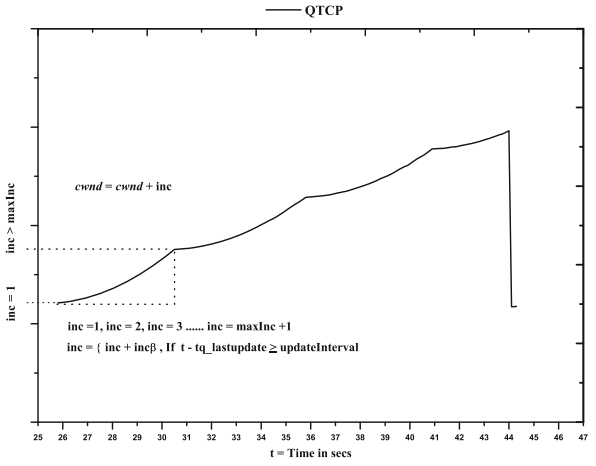

Fig. 3. The window growth function of QTCP α in slow-start phase

each successful acknowledgement by adding *inc* to *cwnd*, and waits for $updateInterval$ and again updates *inc* by adding $inc\beta$. In line two it checks if *inc* hits its maximum value *maxInc* (which is set to its default value 10) then it resets it to 1. This process will be continued until *cwnd* hits its maximum window size, when *cwnd* hits its maximum window size then the window will be decreased by the $decrease\_factor$ for reducing congestion to avoid packet loss. Once the window is decreased, again QTCP will keep increasing the sending rate by computing *inc* using $\beta increase(t)$ in the absence of congestion, QTCP continues this process of avoiding congestion, maintaining Stability, maintaining fairness and achieving utilization until the flow completes its data transfer.

4 Performance Evaluation

QTCP evaluated on the basis of the average throughput ratio, packet loss rate and link utilization between inter protocols with different RTTs in the given experiment. Simple topologies, like a dumbbell with one congested link, are sufficient to study many traffic properties.

4.1 Experimental Setup

In this section, the performance of the QTCP with different RTTs is analysed. The main attempt of the configuration is to compare the QTCP average throughput ratio, bandwidth link utilization and packet loss rate in inter and intra RTTs with inter protocols. The topology used in these experiments is shown in Figure 4.

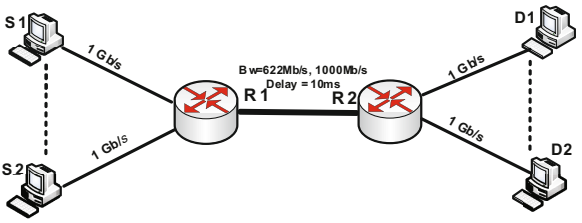

Fig. 4. Network topology used for connections with different RTTs

Dumbbell topology with two routers R1-R2 is located at the bottleneck between the two end points. The maximum bandwidth of the bottleneck routers is set in 622Mb/s and 1Gb/s, and the delay is set at 10ms. The dotted line shows the connection link between the sender/receiver nodes with the capacity of 1Gb/s. The bottleneck buffer size of the senders (S1-S2) and receivers (D1-D2) sides are fixed at 1555 packets. The start time of different flows is set between 0.1 and 0.5 seconds for each simulation, each time simulation run for 1000 seconds. In this section, inter protocols QTCP simulations are analysed with different RTTs. To determine the behaviour of different connections with different RTTs, the round trip time of connection one fixed at 160ms while the round trip time of connection to varied at 24ms, 40ms, 60ms, 80ms and 160ms. The average throughput ratios of each competing connection between 24ms RTT to 160ms RTT are shown in Figure 5(a) and Figure 5(b). The bandwidth utilization is restricted to 622Mb/s in Figure 5(a), which shows QTCP has a better average throughput ratio compared to all the inter protocols except HTCP. The average throughput ratio of HTCP slightly decreases whenever the RTTs increase. The QTCP bandwidth utilization with 1000Mb/s in Figure 5(b) reveals a better average throughput ratio with all inter protocols except HTCP and FAST. The FAST TCP has a slightly smaller average throughput ratio than HTCP.

Figure 6(a) and Figure 6(b) shows the results of QTCP inter protocol packet loss rate with different RTTs, and 622Mb/s and 1000Mb/s bandwidths, respectively. In Figure 6(b) it reveals that the FAST TCP packet loss rate is slightly higher at 80ms RTT. STCP has high packet loss rate, as shown in Figure 6(a) and Figure 6(b). In this evaluation, the QTCP shows the lower packet loss rate with all other inter protocols except FAST TCP and STCP, therefore it is found that QTCP is more compatible with all other high-speed TCP protocols.Figure 7(a) and Figure 7(b) shows the results of QTCP inter protocol link utilization with different RTTs, and 622Mb/s and 1000Mb/s bandwidths, respectively. Figure 7(a) confirms that QTCP has best link utilization, which shows the better share of resources with all inter protocols. As compared to the large RTTs, the short RTTs provide better link utilization results by up to 99.88%, and 76.53% in large RTTs. Figure 7(b) indicates that QTCP better link utilization with all the inter protocols except HTCP and FAST TCP. The HTCP link utilization gradually decreased, when the RTTs increased. The FAST TCP has slightly smaller link utilization at 60ms RTT.

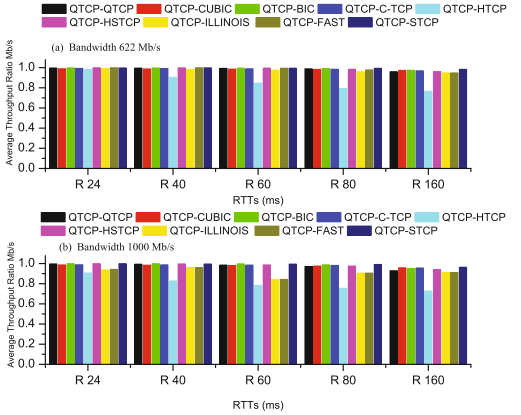

Fig. 5. Inter protocols throughput ratio with different RTTs

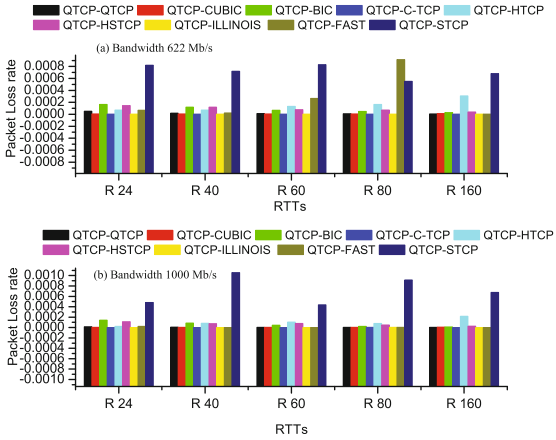

Fig. 6. Inter protocols packet loss rate with different RTTs

4.2 Fairness

We computed the fair share link metric by using equation (7), which describes the Jain fairness index [13]. A series of experiments for the evaluation of multiple high-speed flows in different configurations is shown in Table 2.

$$F = \frac{(\sum_{i=1}^{n} \bar{x}_i)^2}{n \sum_{i=1}^{n} \bar{x}_i{}^2} \tag{7}$$

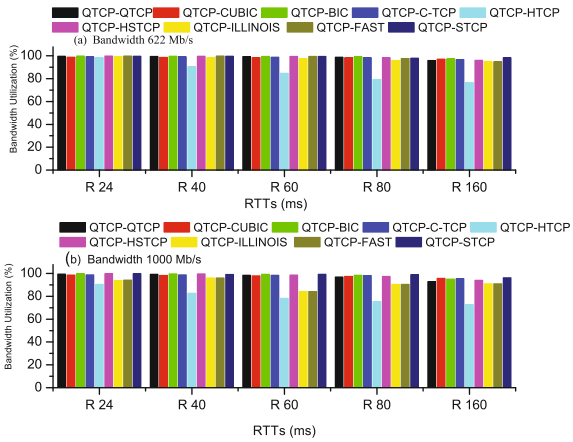

Fig. 7. Inter protocols bandwidth utilization with different RTTs

Table 2. QTCP fairness evaluated with other high-speed protocols

	Bandwidth Mb/s				
Protocols	200	400	600	800	1000
QTCP-QTCP	1.00	1.00	1.00	0.99	0.98
QTCP-CUBIC	0.99	0.99	0.98	0.99	0.99
QTCP-BIC	1.00	1.00	1.00	0.99	0.99
QTCP- C-TCP	0.98	0.99	0.99	0.99	0.98
QTCP-HTCP	0.99	0.99	0.99	0.99	0.99
QTCP-HS	0.99	0.99	1.00	0.99	0.98
QTCP-ILLINOIS	1.00	1.00	1.00	0.99	0.98
QTCP-FAST	1.00	1.00	1.00	0.99	0.98
QTCP-STCP	0.90	0.90	0.95	0.96	0.95

5 Conclusion

The QTCP, a new variant of high-speed TCP, was proposed and studied in this article. The most significant finding that emerged from this study is that the QTCP achieves improved fairness in terms of the multiple flows sharing the link and it also offers high throughput in terms of complex network resources compared to most of the existing high-speed TCP variants. The inter and intra protocol fairness evaluation over different RTTs, verified that the QTCP can effectively utilize the link capacity with low packet losses and excellent throughput ratio. Consequently, it is recommended that the future trials should analyse the parallel flows that use multiple effective TCP connections. It is assumed that as the parallel TCP would aggregate, the throughput would increase, the loss rate would decrease and, hence, even better throughput would be achieved.

Acknowledgements. The research was supported by the Malaysian Ministry of Higher Education, Fundamental Research Grant (FRGS) 01-11-09-734FR.

References

1. Xue-zeng, P., Fan-jun, S., Yong, L., Ling-di, P.: CW-HSTCP: fair TCP in high-speed networks. Journal of Zhejiang University Science 7(2), 172–178 (2006)
2. Fan-jun, S., Xue-Zeng, P., Jie-Bing, W., Zhengi, W.: An algorithm for reducing loss rate of high-speed TCP. Journal of Zhejiang University Science 7 (supp.II), 245–251 (2006)
3. Floyd, S., Ratnasamy, S., Shenker, S.: Modifying TCP's congestion control for high speeds, Technical note (May 2002), http://www.icir.org/floyd/papers/hstcp.pdf
4. Zhang, Y., Lemin, L., Wang, S.: Improving Reno and New-Reno's performances over OBS networks without SACK. International Journal of Electronics and Communications 63(4), 294–303 (2009)
5. Floyd, S., Henderson, T., Gurtov, A.: The NewReno modification to TCP's fast recovery algorithm (April 2004), http://www.faqs.org/rfcs/rfc3782.html
6. Kelly, T.: Scalable TCP: Improving performance in high-Speed wide area networks. ACM SIGCOMM Computer Communication Review 33(2), 83–91 (2003)
7. Xu, L., Harfoush, K., Rhee, I.: Binary increase congestion control for fast, long distance networks. In: Proceeding of NETWORKING 2006, pp. 476–487 (2006)
8. Rhee, I., Xu, L.: CUBIC: A new TCP-friendly high-speed TCP variant. In: Proceeding of PFLDnet (2005)
9. Shorten, R., Leith, D.: H-TCP: TCP for high-speed and long-distance networks. In: Proceeding of Second International Workshop on Protocols for Fast-long Distance Networks (2004)
10. Tan, K., Song, J., Zhang, Q., Sridharan, M.: A compound TCP approach for high-speed and long distance networks. In: Proceeding of 25th IEEE INFOCOM International Conference on Computer Communications (2006)
11. Liu, S., Basar, T., Srikant, R.: TCP-Illinois: A loss- and delay-based congestion control algorithms for highspeed networks. In: Proceeding of 25th IEEE INFOCOM International Conference on Computer Communications, Performance Evaluation, vol. 65, pp. 417–440 (2008)
12. Weigle, M.C., Sharma, P., Freeman, J.: Performance of competing high-speed TCP flows. In: Proceedings of the IFIP Networking, Coimbra, Portugal (2006)
13. Chiu, D., Jain, R.: Analysis of the increase/decrease algorithm for congestion avoidance in computer networks. Computer Networks and ISDN 17(1) (June 1989)