Summary of the paper:

Based on the previous work of FAST TCP, this paper designs a new TCP method to deal with the TCP incast problem in datacenter.

First of all, the author points out that the TCP incast problem is due to the long link idle time caused by the long timeout of the traditional TCP. Then the author gives two detailed reasons about why the TCP timeout always happens in datacenters, especially those with synchronized block transmission communication pattern.

The first reason is Head-TOs (head timeout). When the receiver tries to initiate a new round of synchronized data transmission, the aggregate congestion window size of all the senders, as a legacy inherited from the previous round, may be so large that the switch buffer is exhausted, causing packet losses and timeouts of the sender TCP.

The second reason is the Last-TOs (last timeout), which is caused by insufficient duplicate ACKs at the tail of synchronized blocks. At the tail of the synchronized blocks, if the buffer is exhausted and the last two or one packets of the TCP transmission is dropped, then ACKs is not enough to trigger the retransmission and a timeout happens.

The author uses a diagram to demonstrate the negative relationship between the total number of TOs (Head-TOs + Last-TOs) and the application through put, indicating that the two kinds of TOs are main cause of the TCP incast problem.

The author designs a new TCP based on FAST TCP. The new TCP involves improvement in the slow start, congestion avoidance.
1) Slow start: The author redefines *ssthreshold,* changing it from a certain congestion window value into a certain queuing delay. If the queuing delay is less than ssthreshold, then the congestion window is increased by 1 upon receiving an ACK. Otherwise it enters congestion avoidance stage. The author estimate the ssthreshold to be 20us in datacenters.
2) Congestion avoidance: The author uses the window growth function of the FAST TCP to deal with congestion avoidance. The α in the original function is a constant, means to be the expected queue length of a flow. Under the datancenter scenario, the author uses experiment to show that larger α achieves high goodput when the total number of senders is small while smaller α achieves high goodput when number of senders are large. So the author set α to be 2 when the sender number is large while 1 when it is small. As it is hard to calculate the number of the concurrent senders, the author still uses queuing delay as an indication of the number of the senders.

The experiment results of this paper shows that the proposed new TCP almost eliminates the occurrence of the timeouts, thus greatly improves the through put when the number of the senders is increased to 30s.