A Dynamic Adaptive Acknowledgement Strategy for TCP over Multihop Wireless Networks

Ruy de Oliveira
University of Bern, IAM, RVS
November 24, 2004
Outline

- Motivation
- What the main problems for regular TCP are
- Related work
- Proposed end-to-end solution
- Simulation evaluations
- Summary and outlook
Motivation

- TCP is widely used in wired networks and so most of applications are designed to it
- Energy consumption and bandwidth utilization are of high concern in wireless networks, which demands efficient protocols
- Existing proposals are quite limited
What are the main problems for TCP?

- The scheduling scheme of 802.11 is not appropriate for multihop networks
  - Link retransmissions (may interfere with transport level rxs)
  - Hidden node
  - Spatial reuse
- The above properties degrade the performance of any transport protocol
  - The problem is even worse as the number of hops grow
- Because of the spatial reuse properties, TCP performs better if low bound for its congestion window is used (~hops/4)
Related Work

- Biaz and Vaidya [BV98] evaluated three schemes, based on RTT and throughput patterns, for predicting pkt loss reason
- Holland and Vaidya [HV99] proposed TCP-feedback which counts on ELFN messages for detecting link breakages
- Liu and Singh [LS01] proposed ATCP with ICMP messages and ECN bit to distinguish the reason of packet loss
- Li et al. [LBCLM01] concluded that in a large multihop scenario the IEEE802.11 MAC protocol allows ideally $\frac{1}{4}$ of the throughput achieved for 1-hop scenario and only $1/7$ if a greedy sender is in place
- Xu and Saadawi [XS02] showed that maximum congestion window (CWND) should be set to around 4 packets
Related Work

- Fu et al. [FZL+03] confirmed the result of [LBCLM01] in that TCP’s cwnd limit should be about hops/4. They also proposed Link RED as a MAC enhancement.

- Chen et al. [CXN+03] proposed an approach to set up the max CWND according to the round number of hops in place, and for 802.11 it should be about (hops/4).

- Altman and Jimenez [AJ03] proposed an extension for delayed acknowledgment strategy by delaying 4 packet instead of standard 2.

- Kherani and Shorey [KS04] studied analytically an extension to delayed acknowledgments. They make strong assumptions such as no timeout at the receiver.
Spatial reuse

- Nodes 2 and 3 clearly inhibit node 1 transmission
- When node 4 transmits, it prevents node 2 from transmitting a CTS
- So, node 1 only can transmit when node 4 is done with its tx to node 5
- Ideally, node 1 should have a congestion window of $h/4$ ($h=$hops)
- Also ideally, the throughput of a long connection is about $\frac{1}{4}$ of the throughput of 1-hop case
Spatial reuse (cont)

- A related work on “Capacity of ad hoc networks” found, by simulation, that the factor $\frac{1}{4}$ can be reduced to up to $1/7$ if the sender is greedy.
- The reason is that the nodes suffer interference of different number of nodes:
  - Node 1 below suffers direct interference of only 2 other nodes (2 and 3).
  - Node 2 is interfered by 3 other nodes (1, 3 and 4).
  - Node 3 is interfered by 4 other nodes (1, 2, 4 and 5).
- Every time a given node faces collision it backs off for a random interval.
The optimal bound for cwnd

- In a realistic scenario, the maximum congestion window should be at most 3 packets.
- In a short connection of up to 3 hops, even a cwnd of 1 is acceptable because the spatial reuse property does not exist.

Settings:
- amount of flows: 1, channel bandwidth: 2 Mbps, simulation time: 200 seconds
What can be done?

- To set the maximum congestion window to the correct value
  - Drawback: Channel utilization is not optimized

Viable solution

- Less medium access requests → lower collisions probability
- Less (re)transmissions at the transport layer
- Make use of short end-to-end connections in terms of hops

Then

- An adaptive delayed acknowledgement strategy may meet these requirements
  - Less ACKs allows more bandwidth for DATA pkts
  - Although small in size, an ACK causes the same overhead at the MAC layer that a DATA pkt does due to the RTS/CTS and backoff strategy

Challenge

- To handle medium losses properly
DATA and ACK packets contentions

- The limitation of using ACKs is not negligible
Our Adaptive Acknowledgment strategy

- **Key idea:**
  - Minimize number of (re)transmissions when the channel is in good condition
  - Improve bandwidth utilization whenever it is possible and also save energy by avoiding useless retransmissions
  - Self-adjust to the channel condition when it changes

- **Target (and realistic) applications:**
  - Indoor networks where level of noisy is relatively low (recovered by MAC retransmissions) and number of nodes is small, such as faculty buildings, conferences and meetings venues, wi-fi at home, and so on.
  - Such applications will normally communicate over a few hops and the disturbances from mobility will be minimal (pedestrian movements)
The protocol design

Sender (toward minimal retransmissions by timeout)

- Sender is triggered to rx upon 2 duplicate ACKs
- Timeout interval is increased fivefold
- Maximum cwnd is set to 4 packets

Receiver

- In steady state condition, sends only 1 ACK out of 4 (d=4) DATA packets received
- An out-of-order packet or timeout triggers an immediate acknowledgement and resets d=1
- Upon every new DATA packet received, “d” increases by 1 until the limit d=4
- Timeout interval is computed adaptively on the basis of the DATA packet inter-arrival
Mechanisms

Dynamic Adaptive Acknowledgment at the receiver

Sender

Receiver

DATA

ACK

\(d=4\)

\(d=4\)

\(d=4\)

\(d=2\)

Timeout and \(d\) adaptively fit the traffic conditions

Window enlargement

\[
d = \begin{cases} 
  d + \mu, & \text{if } \max d = \text{false (startup)} \\
  d + 1, & \text{otherwise}
\end{cases}
\]

Adaptive Receiver Window for delaying ACKs

Varying Window

pkt loss

\(p_i\)

\(p_{i+1}\)

\(p_{i+3}\)

\(d=4\)

\(d=2\)

\(d=3\)

\(d=4\)
Packet Inter-arrival smoothing

Low-pass filter:

\[ t_i = (1 + \kappa) \times \overline{\delta}_i \]

\[ \overline{\delta}_i = \alpha \times \overline{\delta}_{i-1} + (1 - \alpha) \times \delta_i \]

- \( t_i \): timeout interval at arrival of pkt \( i \) to be delayed
- \( \overline{\delta}_i \): smoothed pkt inter-arrival
- \( \delta_i \): sampled pkt inter-arrival (from delayed ACKs only)
- \( \alpha \): smoothing factor
- \( \kappa \): timeout tolerance factor
Performance evaluation

Throughput:

- Only losses due to collisions by the MAC protocol
- Our protocol (TCP-DAA) outperforms the other flavors in most cases, mainly for large number of flows (results: average of 5 runs)
Performance evaluation

Throughput:
- Our protocol (TCP-DAA) outperforms the other flavors in most cases, mainly for large number of flows
- Up to about 50% of gain
Performance evaluation

Energy Efficiency:
- Energy spent by the sender
- Regular TCP spent about 26% more energy than TCP-DAA
Performance evaluation

Short-lived flows:
- Scenario: chain topology without hidden node problem
- TCP-DAA performs as effective as the other flavors
Performance evaluation

Short-lived flows:
- Scenario: chain topology with hidden node problem
- TCP-DAA may not have enough time to reach equilibrium for very short files !!! → too conservative
Performance evaluation

**RTT computation:**
- The adaptive mechanism induces considerable inaccuracy in the conventional RTT computation
  
  **RTT estimation:**
  - Too small → needless retransmissions
  - Too large → slow reactions

  **A more robust computation scheme is needed**
Performance evaluation

Delay compensation:

- Receiver informs the sender how much it has delayed the ACKs
- The sender takes the echoed feedback into account when computing the smoothed RTT

\[
SRTT_{i+1} = (\alpha \cdot SRTT_i) + (1-\alpha) \cdot (s_i + td)
\]


**Performance evaluation**

**Delay compensation:**

- Sender reacts more promptly to lost packets
- There is a clear tradeoff between responsiveness and bandwidth utilization
Summary

- The protocol works very well under moderate level of losses even with congestion
- It is easy to deploy (end-to-end)
- Good tradeoff between throughput and energy saving is achieved
- Benefits for very short-lived flows, in large scenarios with various concurrent flows, to be improved
- More accurate timeout computation at the sender is needed
Problems to be investigated

- Robustness under higher wireless losses
- Higher number of rx at MAC layer
- Better RTT computation at the sender