Study on multi-hop 802.11 ad hoc networks

He Xiaoben

Helsinki University of Technology

Email: xiaoben.he@nokia.com

MAC problems revealed by simulation

- Three problems revealed from simulation experiments:
  - Problem 1 – TCP instability
  - Problem 2 – Serious unfairness
  - Problem 3 – TCP incompatibility
- All problems rooted in 802.11 MAC
- TCP traffic enlarges the problems of MAC layer protocol.

802.11 MAC (1/2)

- Two different access methods defined in 802.11 MAC:
  - DCF (the Distributed Coordination Function)
  - PCF (the Point Coordination Function)
- The basic mechanism (idea) of DCF: CSMA/CA
- The basic basic idea of CSMA:
  - Sensing the medium before transmitting.
  - Deferring the transmission to a later time, if the medium is busy.
  - Transmitting, if the medium is sensed as free.
- However, collision still happens when:
  - Nodes sense the medium as free and send at the same time.
  - With "hidden node problem", two nodes collide due to not hearing each other.

802.11 MAC (2/2)

- 802.11 collision avoidance (CA) mechanism coupled with a positive acknowledge
  - Distributed Inter-Frame Space (DIFS) defined.
  - MAC layer Acknowledgement (M-ACK).
    Re-transmission until a M-ACK is received at the sender, or throw away after a given number of retransmissions. (7 retransmissions in current std.)
- To cope with the "hidden nodes problem", a virtual carrier sense mechanism is defined:
  - Request to Send (RTS): a short control packet sent before data packet transmission, which includes the source, destination, and duration of the intended packet and ACK transmission.
  - Clear to Send (CTS): a short control packet in response to RTS, which includes the same duration information.
  - Network allocation vector (NAV) is the virtual carrier sense indicator, which is set for the given duration when either RTS or CTS is received.
- NAV State is combined with the clear channel assessment (CCA), which is the physical carrier sense, to indicate the busy state of the medium.

TCP Overview

- TCP is a window-based ACK clocked flow control protocol. It uses an additive-increase/multiplicative-decrease strategy for changing its windows according to network conditions.
- Two phases defined for TCP window increase/decrease:
  - Slow start (SS) phase

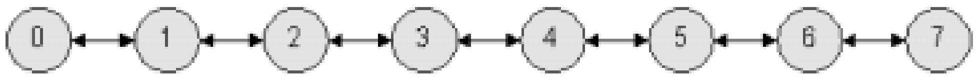
    Starting from one packet, window is increased exponentially by one packet for every non-duplicate ACK until the resource estimate of netowrk capacity is reached.
  - Congestion avoidance phase

    Window is increased by one packet for every window's worth of ACKs. The window increase will stop when it reaches the maxium TCP window size, which is defined when the connection starts. Otherwise, the window increase is interrupted when a loss is detected.

Simulation environment settings

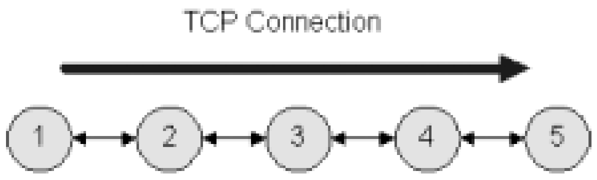
- Simulator: NS2 with extensions from MONARCH project at CMU, which includes:
  - a set of mobile ad hoc network routing protocols;
  - an implementation of BSD's ARP protocol;
  - an 802.11 MAC protocol DCF;
  - Modeled after 802.11 based WaveLan wireless radios, which is:
    - half-duplex
    - BW: 2Mbps; nominal transmission radius: 250m; reference distance *r* is 100m.
  - Two radio propagation models:
    - The free space propagation model (signal attenueates as $1/r^2$) is used when the transmitter is within the reference distance r of the receiver.
    - The two-way ground reflection model (signal attenueates as $1/r^4$) is used when the transmitter is outside the reference distance r of the receiver.

    Delay and power level of the received signal can be calculated from the propagation model.
- TCP settings:
  - Large file transfers (i.e. infinite backlog of data which the TCP sender always has to send out).
  - Fixed size TCP packet, packet index used as the TCP sequence number.
- Network topology: a string topology with eight nodes (0 ~ 7), distance between any two neighbouring nodes is 200m, which allows a node to communicate only with its neighbouring nodes.



Problem NO.1 – TCP instability

- Experiment setting:



Source: node 1, destination: node 5, TCP packet size: 1460 bytes

- We expect the TCP connection should achieve a stable throughput, as there is no background traffic, no network condition changes, no congestions …, but

TCP instability - throughput variations case 1

- Fig. 1a. illustrates the measured throughput variations during the lifetime of one simulation run.
- We could observe serious instability in the TCP throughput. In Fig. 1a, there are 20 instances when the throughput reaches or nears zero.

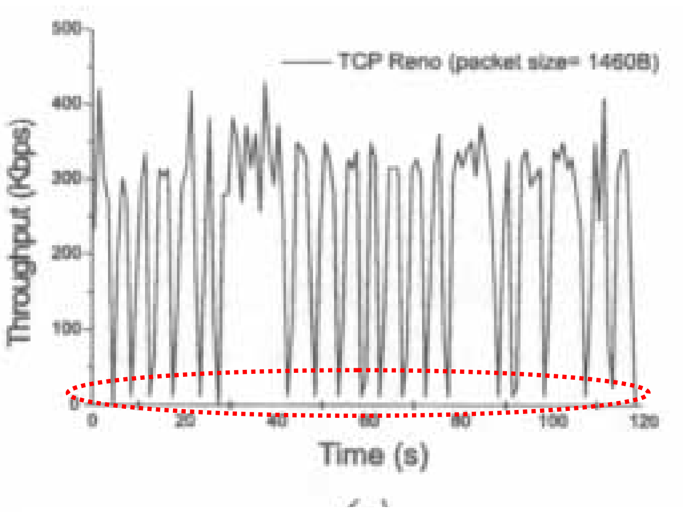


Fig. 1a. Instability problem in the four hops TCP connection.

case 1: window = 32

TCP instability - throughput variations case 2

- When the TCP window is set to 8, the instability situation improves a bit. Fig. 1b. shows the case.

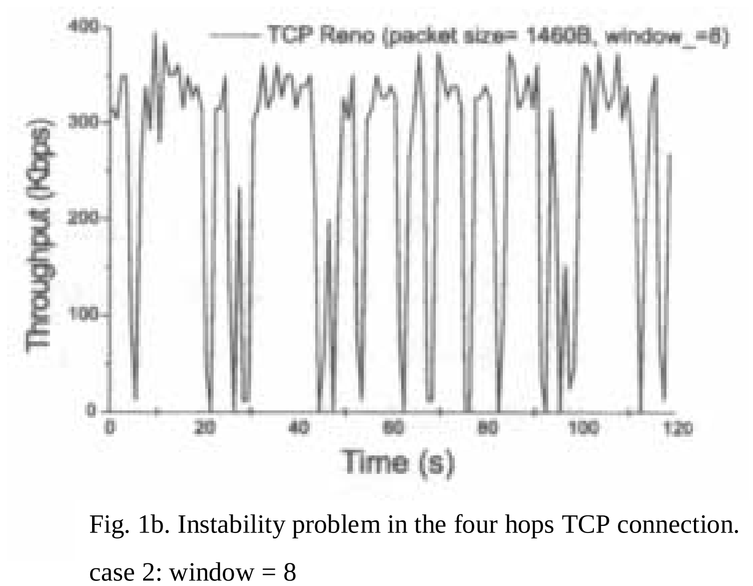


Fig. 1b. Instability problem in the four hops TCP connection.

case 2: window = 8

TCP instability - throughput variations case 3

- In Fig. 1c. the TCP window is set as 4, no serious instability problem occurs.

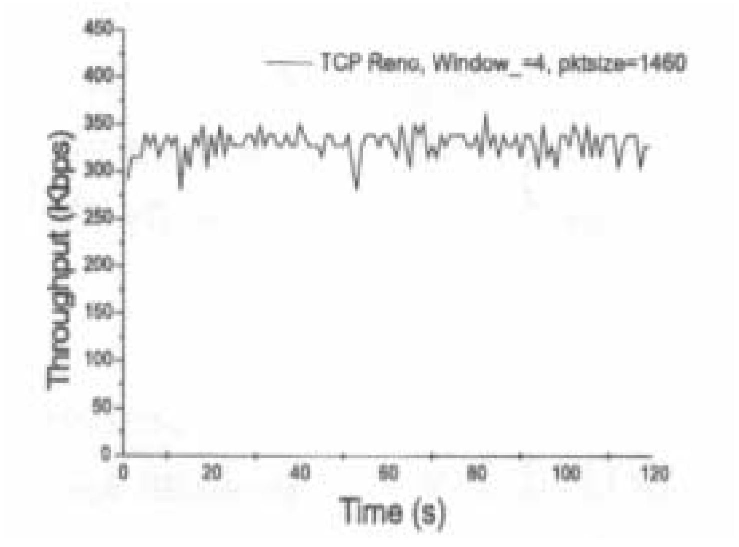


Fig. 1c. Instability problem in the four hops TCP connection.

case 1: window = 32

TCP instability – Analysis of throughput variations (1/2)

- Fig. 2. illustrates the packet events from part of the Fig. 1b. simulation.
- After the ACK drops at 4.02s, no ACK packet arrives at the TCP sender until 6.1s, when a route from node 1 to 5 is available.
- TCP Packet 111 was transmitted twice after that, it arrives at the TCP receiver (node 5) safely, but the corresponding ACK cannot be sent.

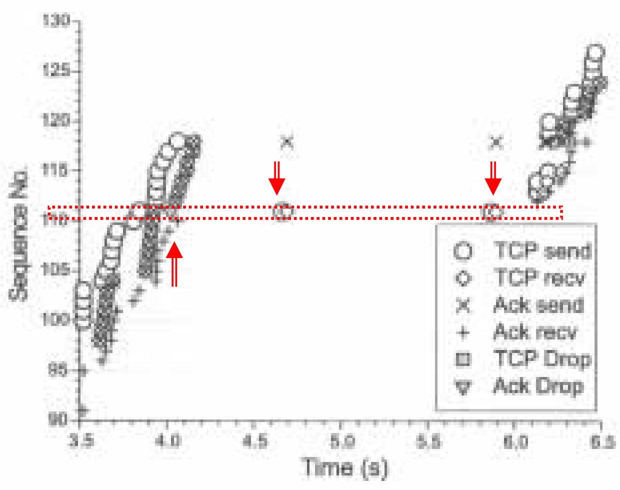


Fig. 2. Part of the packet events of TCP session shown in Fig. 1b. window = 8, packet size = 1460 bytes.

TCP instability – Analysis of throughput variations (2/2)

- TCP packet 516 cannot find a route, there is no ”TCP recv” for it, until 23s.
- This is a route failure, but why?

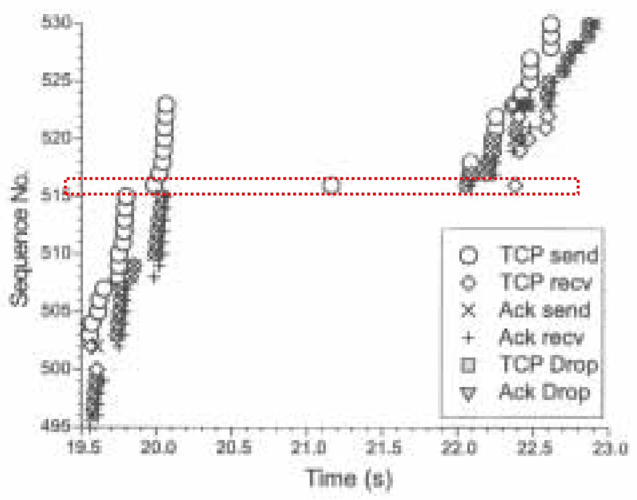


Fig. 2. Another part of the packet events of TCP session shown in Fig. 1b. window = 8, packet size = 1460 bytes.

TCP instability – a more closer look

- After the TCP packet 516 was tried to be sent and failed 7 times, a link breakage was reported.
- Note that, 7 retries is a parameter defined in IEEE 802.11. Then it is good to look at MAC layer trace.

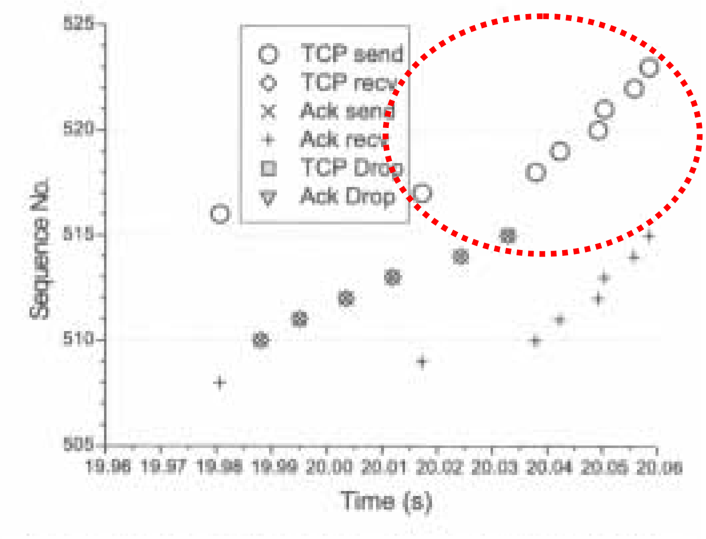


Fig. 3. Zoom of Fig. 2, for session in Fig. 1b. window = 8, packet size = 1460 bytes.

TCP instability – MAC trace

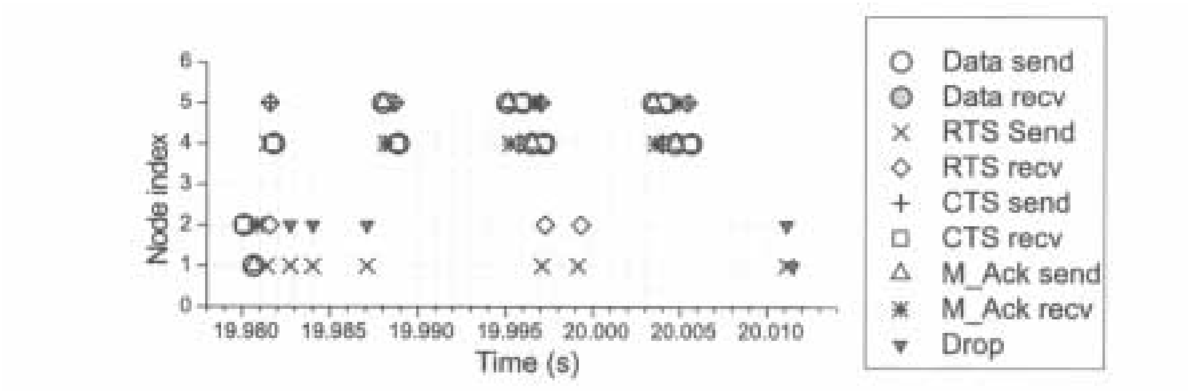
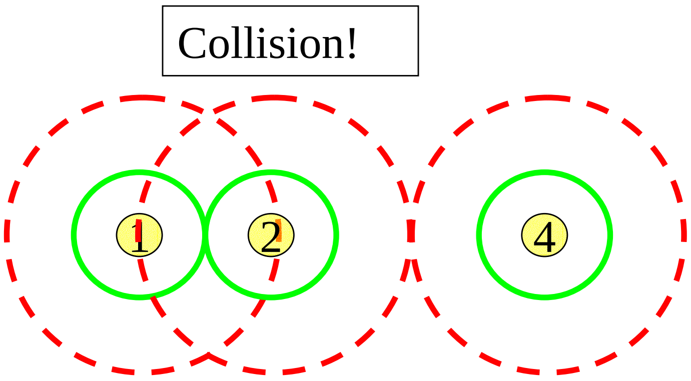


Fig. 4. Part of the MAC layer packet trace, for session in Fig. 1b, window = 8, packet size = 1460 bytes.

- In Fig. 4, ”Data” means TCP packet or TCP ACK packet. In reference to the MAC layer, they are all data from upper layer.
- The major cause of node 1’s failure to reach node 2 is: a) node 2 can’t successfully received RTS from node 1, 4 out of 7 RTS sent from node 1 was dropped at node 2; b) node 2 does not sent back a CTS to node 1 when it receives a RTS, 3 out of 7 RTS was received at node 2, but no CTS was sent.
- After 7 retries of sending RTS but without receiving CTS from node 2, node 1 also drop the MAC packet and quit the delivery. At the same time, a link breakage event is reported to the upper layer.
- The direct reason is node 4 is sending data packets to node 5 at the same time. Collision happened.

TCP instability – What happened? (1/2)

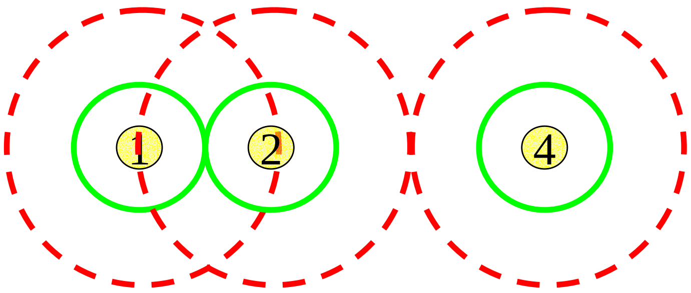


- Explain the collision:

NOTE: node 4 is out of node 1's sensing range, but node 2 is within node 4's interfering range.

  – Case 1, when node 1 sends RTS to node 2, it can't sense that node 4 is sending data at the same time; when this RTS arrives at node 2, it collides with the interference from node 4, thus node 2 drops the RTS. Although the node 1 is free to send, but the receiver node 2 is not free. This is a typical "Hidden terminal problem".

TCP instability – What happened? (2/2)



- Explain the collision:

NOTE: node 4 is out of node 1's sensing range, but node 2 is within node 4's interfering range.

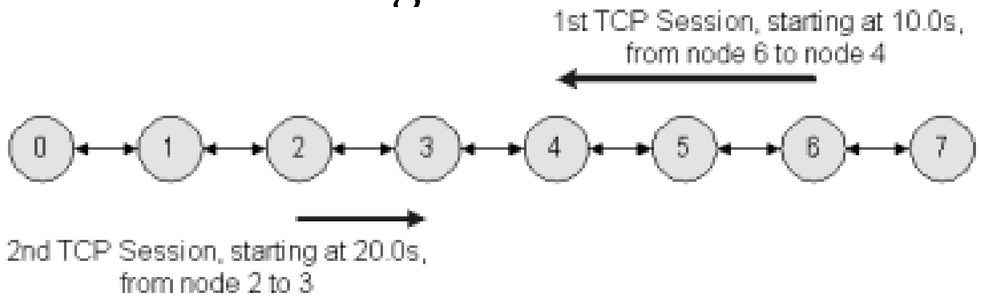
Case 2: when node 2 is about to send CTS to node 1, it senses that node 4 is sending data, so it chooses a random backoff and wait, wait and wait, as node 4 is all the time sending … Although node 1 is free to receive CTS. This is a typical "Exposed terminal problem"

TCP instability – Discussions

- Neither the PHY carrier sensing (CCA) mechnism, nor the virtual carrier sensing (NAV) helps. ”hidden terminal problem” and ”exposed terminal problem” prevent the intermediated node from reaching its next hop. Collision happened anyway.
- The random backoff scheme used in the MAC layer makes this worse, since it always favors the latest successful node. As bigger data packet sizes and sending back-to-back packets both increase the chance of blocking the intermediated nodes.
- The TCP instability problem can be lessened or eliminated by adjusting one parameter in TCP (window size).

Problem NO.2 – Unfairness

- Simulation settings:


1st TCP Session, starting at 10.0s, from node 6 to node 4
0
1
2
3
4
5
6
7
2nd TCP Session, starting at 20.0s, from node 2 to 3

- We expect the two TCP sessions share the total bandwidth, more or less equally, but …

Unfairness – Case 1

- The 1st TCP session is completely forced down after the 2nd one starts at 30.0s. After that, the throughput of the 1st TCP session is zero for most of its lifetime, and there is never a chance for it to restart.
- This is serious unfairness, the loser session is completely shut down even if it starts much earlier.
- Can this problem be eliminated by adjusting TCP parameters? (e.g. window size)

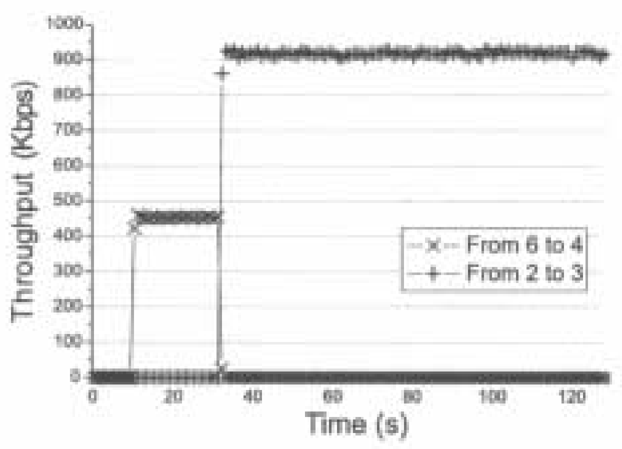


Fig. 5. Througput of two TCP connections with different sender and receiver, window_ = 4

Unfairness – Case 2

- The situation is the same with TCP window decreased to 1. The 1st session is completely shut down after the 2nd one starts at 30.0s. After that, the aggregate throughput of these two TCP connections is almost cmpletely supplied by the second session.
- We call this run ”W1 run” in the following statement.

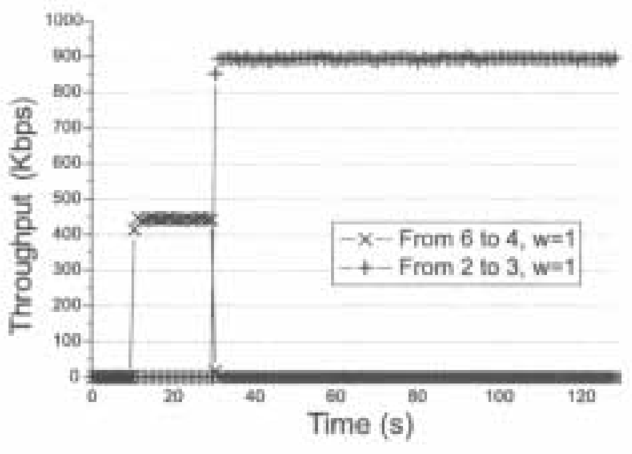


Fig. 6. Througput of two TCP connections with different sender and receiver, window_ = 1

Analysis of the unfairness

- After 30.07s, no TCP packet from the 1st TCP is delivered successfully from node 6 to 4. Packet 2164 never arrives node 4.
- By checking the TCP trace in detail, we found the drop reason as: "No Route available", and "Time out".
- The route failure seems very strange, since no node moves in this simulation. Why?

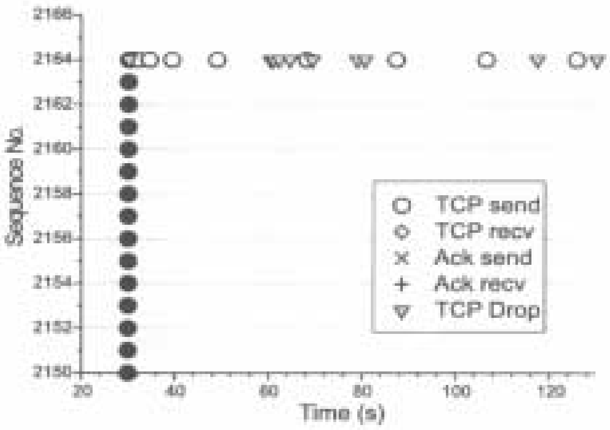


Fig. 7. Part of the packet events of the first TCP session in the "W1 run", window_ = 1

Unfairness– MAC Trace

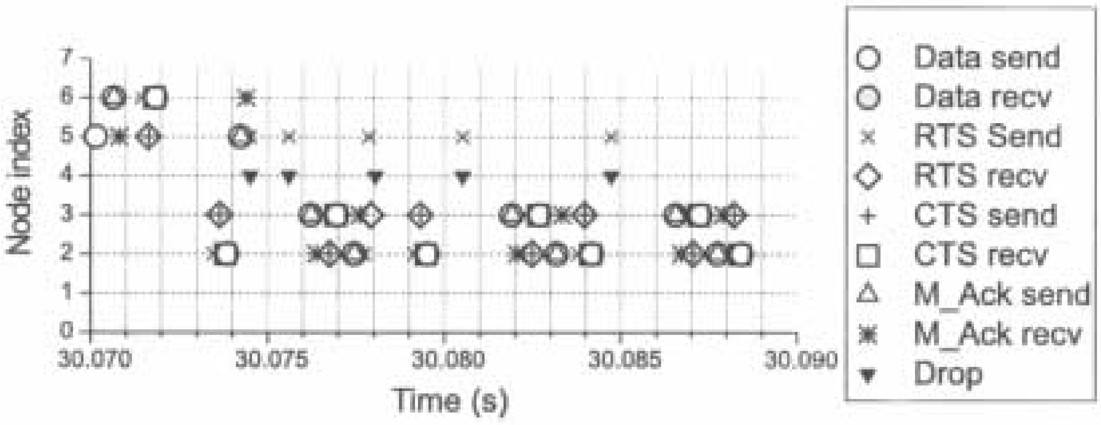


Fig. 8. Part of the MAC layer packet trace in the "W1 run", window_ = 1

- The route failure rooted in MAC layer, node 4 can't receive any RTS from node 5, as interference from node 2 collide at node 4.

Unfairness – What happened?

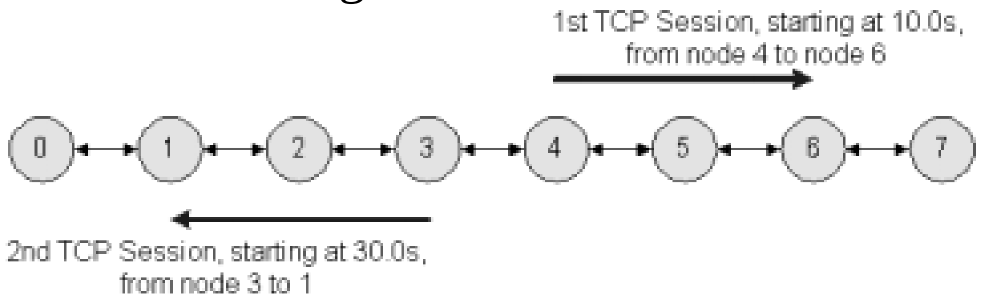
- Once again, collision happens and the virtual carrier sensing can’t help, as node 2 is out of the sensing range of node 5. This is a typical ”hidden terminal problem”
- Moreover, even if node 4 successfully receives the RTS from node 5, it still can’t send back a CTS when node 2 or 3 is sending. This is a typical ”exposed terminal problem”

Problem NO.2 – Discussions

- Now we inspect in what conditions node 5 can reach node 4 if there is a TCP session between node 2 and 3.
- We found the only chance for node 5 to access the channel to node 4 is by sending out an RTS before node 2 sends out an RTS.
- While at the moment node 3 just finished sending data packet (TCP ACK). Node 5 has little chance to do so. Moreover, the binary exponential back-off scheme in the MAC layer always favors the last succeeding station (node 2 in this case), node 5 hardly wins the contention.

Problem NO.3 – TCP incompatibility

- Simulation settings:


1st TCP Session, starting at 10.0s, from node 4 to node 6
0
1
2
3
4
5
6
7
2nd TCP Session, starting at 30.0s, from node 3 to 1

- Can these two TCP sessions share the aggregate throughput equally, or the 2nd one shut down? What will happen?

TCP incompatibility - case 1

- When the 2nd TCP start at 30.0s, they both stay alive until 51.0s, in which the 1st session shut down.
- However, the 2nd session can't always maintain this fortunate state. It is shut down at 71.0s and the 1st session becomes alive.
- In the lifetime of the simulation, this turnover happens four times. The two TCP sessions can't keep alive at the same time.

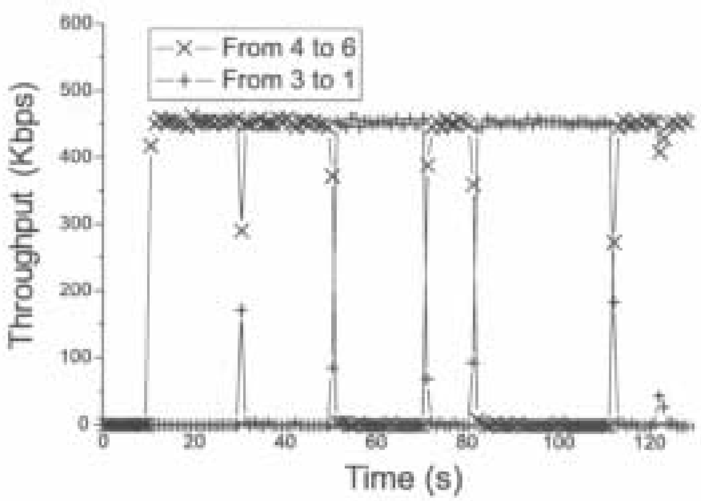


Fig. 9. Throughput of two TCP connections with the same hop numbers, window_ = 4

TCP incompatibility - case 2

- After repeating this experiment many times with different simulation seed, it is realised that these two TCP sessions can't co-exist in such a network.
- It is unpredictable which session will shut down, and when.

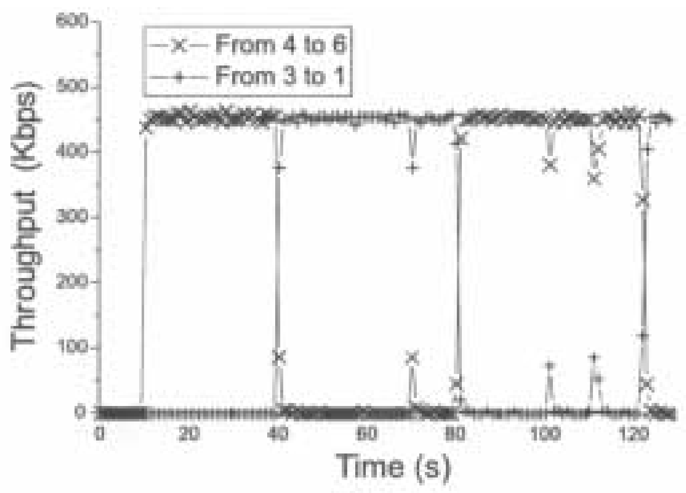


Fig. 10. Throughput of two TCP connections with the same hop numbers (another run), window_ = 4

TCP incompatibility - case 3

- But what will happen if the TCP sources are not direct neighbours? Fig. 15. shows the situation, where the 1st session is from 6 to 4, and the 2nd from 1 to 3.
- In the lifetime of simulation, 3 turnovers occur.

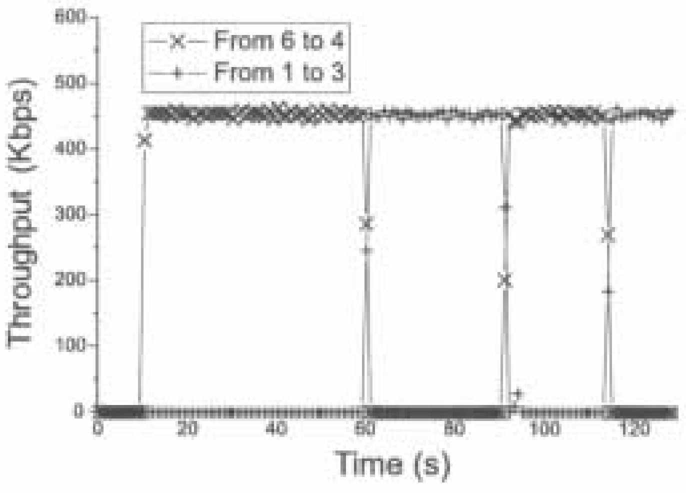


Fig. 11. Throughput of two TCP connections with the same hop numbers (another run), window_ = 4

TCP incompatibility - case 4

- Fig. 12. shows the simulation under same settings except that the TCP window is 1. We call this W1_2 run.
- The TCP incompatibility problem can’t be eliminated by adjusting TCP window size.
- In the lifetime of simulation, 3 turnovers occur.

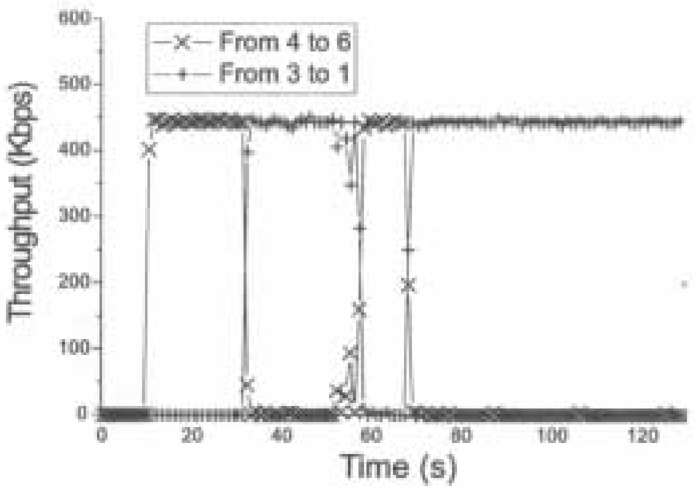


Fig. 12. Throughput of two TCP connections with the same hop numbers (another run), window_ = 1, ”W1_2 run”

TCP incompatibility –Analysis of MAC Trace

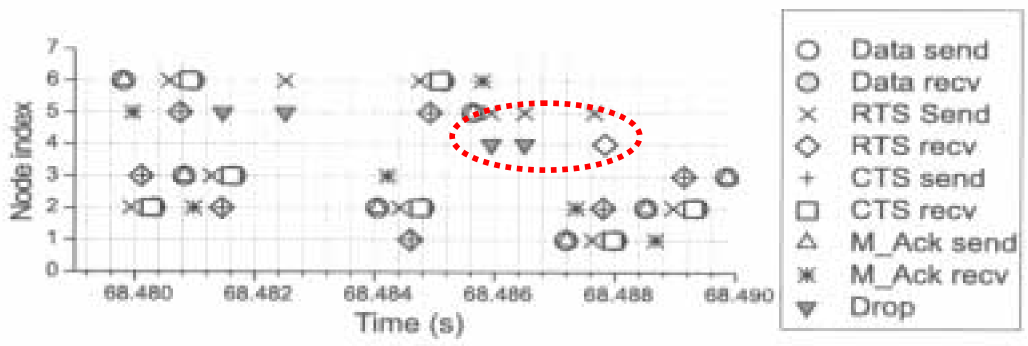


Fig. 13. Part of the MAC layer packet trace in the ”W1_2 run”, window_ = 1

- Look at node 5 at 68.486s, after it receives the data from node 6, it sends RTS to node 4. But at the same time, node 2 is sending data packet to node 1, collision happens at node 4. This is a typical ”hidden terminal problem”
- In 68.4878s, node 4 successfully received a RTS from node 5, but it can’t send a CTS to node 5, as node 2 and 3 is sending data packet. This is a typical “exposed terminal problem”

Conclusions

- The current 802.11 MAC protocol supports some kind of ad hoc network architecture, which only means a distributed networking as opposed to a centralised one, it is not intended to support the wireless mobile ad hoc network, in which multi-hop connectivity is one of the most prominent features.
- TCP can hardly work well in 802.11-based multi-hop network.
- More efforts on the MAC layer are expected to design a usable multi-hop wireless network.

References

- S. Xu, T. Saadawi, ”Revealing the problems with 802.11 medium access control protocol in multi-hop wireless ad hoc networks”, Computer Networks, vol.38, 2002
- S. Xu, T. Saadawi, ”Does the IEEE 802.11 MAC Protocol work well in multihop wireless ad hoc networks?”, IEEE Com. Mag., Jun. 2001

Homework

- NO.1, Please elaborate following concepts:

  a) Hidden terminal problem.
  b) Exposed terminal problem.

- NO.2, What are the proposed solutions for the above mentioned problems in wireless networks? What are the corresponding short-comings of each solutions?