Advanced Computer Networks
263-3501-00

Wireless TCP, White Spaces

Patrick Stuedi
Spring Semester 2013
Outline

- Last week:
  - IP layer, Mobility: Mobile IP, NAT, SIP
- Today:
  - Wireless TCP
  - White Space Networking
Wireless TCP
Remember: TCP congestion control

- Congestion control got added to TCP to in attempt to reduce congestion inside the network
- Must rely on indirect measures of congestion
- Implemented at the sender

![Diagram showing TCP congestion control](image)
TCP Slow Start

- **Congestion window (CW)**
  - Number of bytes in TCP that can be transmitted without waiting for the ACK (CW always smaller than receiver window, flow ctrl)
  - Initially set to 1 TCP segment (e.g., 1 KB)

- **SSThresh**
  - Initially set to 64 KB

- **TCP congestion control:**
  - After each acknowledged TCP segment, increase CW by the number of bytes in the corresponding segment
    - exponential increase (slow start, CW doubles each RTT)
  - If CW >= SSThresh increase CW by 1K/CW after each ACK
    - linear increase (congestion avoidance)
  - On a timeout: Set SSThresh to half of the current CW, then set CW back to 1K
Example: Slow start
Problems of TCP in Wireless Networks

- Congestion control algorithm has been designed for wired/fixed networks
  - In fixed networks a packet loss is an indication of congestion

- But in wireless networks a packet loss is often due to wireless errors
  - Packet lost due to transmission errors or mobility

- TCP cannot distinguish between errors and congestion
  - TCP unnecessarily reduces window, resulting in low throughput and high latency
Problems of TCP in Wireless Networks (2)

- Delay is often high
  - RTT can be very long and variable
  - TCP's timeout mechanism may not work well
- Links may be asymmetric
  - Delayed ACKs in the slow direction can limit throughput in fast direction
TCP in Wireless: Solutions

- Cannot change TCP fundamentally
  - TCP congestion control keep the Internet operable
  - Improvements have to be interoperable

- Possible Solutions:
  1) End-to-End
     - Fast retransmit, Selective Acknowledgments
  2) Split Connection
     - Separate wired path and wireless hop
  3) Link Layer
     - Error-correcting codes
     - Local retransmissions
     - Snooping
End-to-End: Fast retransmit

- Note: TCP sends an acknowledgement only after receiving a packet.
- If a sender receives several acknowledgements for the same packet, this means:
  - The receiver got all packets up to the acknowledged packet in sequence.
  - The receiver is still receiving packets.
  - The gap is most likely not due to congestion, try to avoid triggering slow start.
- Fast retransmit:
  - If sender receives three duplicate ACKs for the same SeqNr he retransmits the missing packet (before the timeout occurs).
Fast Retransmit: Example

- Single packet loss (within a RTT) can be handled with fast retransmit
- Fast retransmit/fast recovery: reduce CW only to the half, and continue with linear increase
Fast Retransmit: Pros/Cons

- Advantages:
  - Simple, minor modifications in the mobile host's TCP stack (TCP Tahoe)
  - Correspondent node's stack does not have to be changed

- Disadvantages:
  - Inefficient: lost packets still have to cross the entire network between correspondent node and mobile host
Split-Connection: Indirect TCP

- Two TCP connections:
  - Fixed to Base: unmodified TCP connection
  - Base to mobile: optimized TCP connection
- Buffering at BS
- Independent flow and congestion control on the two connections
Indirect TCP: Pros/Cons

- Advantages:
  - Transmission errors on the wireless link do not propagate into the fixed network
  - The short delay on the mobile hop is known and therefore it is possible to use precise timeouts and fast retransmissions
  - It is possible to use a different transport layer protocol on the mobile hop

- Disadvantages:
  - Serious: Loss of end-to-end semantics
    ACK to the sender does not any longer mean that the receiver got a packet, FAs may crash
  - Problems during handover
Socket and State Migration during I-TCP handover

- Old proxy must forward buffered data to new proxy because it has already acknowledged the data with the CN
  - Migrate TCP buffer to new proxy
  - Migrate socket state (seqnbr, addresses, ports, etc)
Snooping TCP

- Snoop agent at the BS
  - Monitors TCP segments and ACKs
  - Cache segments until acknowledged
  - Detects packet loss
  - Retransmit lost packets if cached
Snooping TCP (2)

- Data transfer to the mobile host
  - Packet loss detected by snooping duplicated ACKs
  - Fast retransmission possible

- Data transfer from the mobile host
  - Packet loss detected by looking at sequence numbers
  - Snooping agent answers with NACK to the MH
Snooping TCP: pros/cons

- Advantages
  - End-to-end TCP semantics is preserved
  - Changes of TCP only within the FA
    - the CN does need to change
  - Handover can be more easily supported than with I-TCP
  - Interoperable with FAs that do not support the enhancement

- Disadvantages
  - Won't work if TCP connection is encrypted
  - Does not isolate the behaviour of the wireless link like I-TCP
    - As long as packets are not acknowledged end-to-end the corresponding node has a timer ready waiting to retransmit/slow start
Snooping TCP Add-on: Explicit Loss Notification (ELN)

- Works with mobile host sources (first link on path is wireless)
- BS keeps track of missing packets from mobile
- When duplicated ACKs are received, BS sets “ELN” bit in ACKs
- When mobile receives duplicated ACKs with ELN bit set, it does not back off, simply retransmit
Other improvements: Selective retransmission

- TCP acknowledgements are cumulative
  - ACK N = correct and in-sequence receipt of packets up to N
  - If single packets are missing a whole packet sequence has to be retransmitted (go-back-n), thus wasting bandwidth

- Selective retransmission
  - RFC 2018 allows for ACKs of single packets (SACKs)
  - Sender can now retransmit only the missing packets

- Advantage
  - Much higher efficiency

- Disadvantage
  - More complex software and more buffer needed at the receiver
    - Trade-off memory, complexity vs performance
TCP over 3G: Best Practice

- Selective ACKs (TCP SACK)
- Increase the TCP’s initial window (from 1 to 2-4 segments)
- Larger TCP receive window (typically only 64KB)
  - Windows scale option (window gets shifted, allows for almost 1G size)
- Explicit Notification Schemes (congestion: ECN bit, loss: ELN bit)
  - Requires support from ECN capable routers
- Fast Retransmit
## Summary TCP

<table>
<thead>
<tr>
<th>Approach</th>
<th>Mechanism</th>
<th>Advantage</th>
<th>Disadvantage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Indirect TCP</td>
<td>Split TCP connection into two connections</td>
<td>Good isolation of wireless link</td>
<td>Loss of TCP end-to-end semantics</td>
</tr>
<tr>
<td>Snooping TCP</td>
<td>“snoop” TCP segments and ACKs and issue local retransmissions if necessary</td>
<td>Transparent for end-to-end connection</td>
<td>Does not work with encryption</td>
</tr>
<tr>
<td>Fast retransmit</td>
<td>Avoids slow start after single packet loss</td>
<td>Simple</td>
<td>No optimal isolation of wireless link</td>
</tr>
<tr>
<td>Selective retransmissions</td>
<td>Retransmit only lost data</td>
<td>Efficient</td>
<td>More complex, requires larger receive buffer</td>
</tr>
</tbody>
</table>
MAUI

- Research Prototype, Microsoft Research
- Code offload, uses Microsoft .Net
  - Profiling
  - Code annotation
  - Reflection
  - Automatic partitioning
- Energy argument: can we use code offloading to reduce the energy consumption on Smartphones?
How energy efficient is 3G code offload?

- 3G consumes significantly more energy to upload a fixed size byte buffer than WiFi
- Energy cost mainly caused by longer RTT

Keep server close to Smartphone to save energy
TCP and WiFi Power Save Mode (PSM)

- Experiment: transfer 500 KB over TCP using Wifi PSM
- First part of the transfer
  - TCP window is small: allow sender to sleep after finishing window
- Second part of transfer:
  - First Ack is received while still sending: sender won't sleep
- PSM may pro-long total transfer time and increase energy consumption
Where are wireless networks going?

- Exponential data growth + limited spectrum/capacity gains
Where are wireless networks going?

- Increasing density
  - Small cells: higher SINR/user → higher capacity

- Increasing spectrum
  - Can we find more spectrum for these networks?
Dense Deployments

- Hard to manage
  - Inter-cell interference: not enough spectrum to avoid interference
  - Variable load
Spectrum Allocation in the US

- Cellular
- Broadcast TV
- Wi-Fi (ISM)

Higher Frequency
In Contrast...

- Large portions of the spectrum are underutilized
  - Free analog TV bands (Migration to digital TV)
  - Guard bands between reserved bands
Idea: Dynamic Use of Spectrum

- **Determine** available spectrum
- **Transmit** in “available frequencies”
- **Detect** if primary user appears
- **Move** to new frequencies
- **Adapt** bandwidth and power levels
Smart Antennas

1) Dynamically identify currently unused portions of spectrum
2) Configure radios to operate in available spectrum band
TV White Spaces

- Unused TV band
  - In the US primarily the upper 700 Mhz band (698-806 Mhz)
  - TV Channels 52-69, each channel 6 MHz wide
Challenges: Fragmentation

- Spectrum is fragmented
- Combine multiple channels to increase bandwidth
- Allocated ranges are of different width
Challenges: Spatial Variation

- Location impacts spectrum availability: cannot assume channel to be free everywhere
Challenges: Temporal Variation

- Same channel will not always be free
- Any connection can be disrupted at any time
- Must re-configure after disruption
Centralized Spectrum Management

- In the US, spectrum managed by the Federal Communications Commission (FCC)
- FCC decided to abandon the “sensing” based approach and instead requires devices to consult a centralized database together with geo-location to learn the spectrum allocation at a given location
References

- KNOWS Project: http://research.microsoft.com/knows