NICTA

Analysis of TFRC in Disconnected Scenarios and Performance Improvements with Freeze-DCCP

Olivier Mehani Roksana Boreli Thierry Ernst

Australian Government
Department of Broadband, Communications and the Digital Economy
Australian Research Council

NICTA Members
NICTA Partners

1. Context
2. Model of TFRC in Disconnected Scenarios
3. Freeze-DCCP/TFRC
4. Future Work and Discussion

- TCP-Friendly Rate Control (TFRC):
  - rate-based congestion control mechanism
    - needs packets losses p and RTT R
    - $X_{\mathrm{Bps}}(p, R) = \frac{s}{R\sqrt{\frac{4p}{3}} + t_{\mathrm{RTO}}\sqrt{\frac{27p}{8}}p(1+32p^2)}$
  - mimicks TCP's behavior
  - TCP-fair congestion control to other transports
- Datagram Congestion Control Protocol (DCCP)
  - unreliable datagrams
  - congestion control
  - multiple congestion control mechanisms (CCIDs)
    - CCID3 uses TFRC
  - interesting replacement to non-congestion aware UDP to carry real-time traffic over shared networks

Context

Motivations for Mobility Support and Issues

- Emerging mobile use-cases
  - mobiles phones and PDAs
  - intelligent transportation systems (ITS)
- Various types of wireless physical technologies
  - 802.11b/g/p (Wi-Fi)
  - 802.16 (WiMAX)
  - UMTS

Link characteristics

- Common wireless issues
  - temporary loss of signal
  - interferences
  - tunnel
- Mobility issues
  - MIPv6 Handoff times
  - disconnections during handoffs (vertical or horizontal)

- Effects at the TFRC sender
  1. feedback messages can no longer be received
  2. gradual reduction of the sending rate (X)
  3. increase of the retransmission timeout (t_{RTO})
- Effect on the connection
  1. lost packects during the disconnection
  2. lower sending rate upon reconnection
  3. additionally, poor adaptation to new network conditions (*e.g.* technology, congestion)

Model of TFRC in Disconnected Scenarios

Problems Raised by Disconnections or Handoffs

NICTA

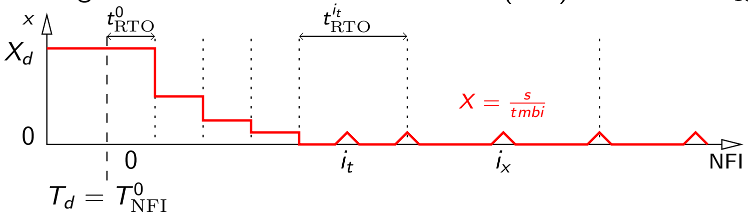
- Effects at the TFRC sender
  1. feedback messages can no longer be received
  2. gradual reduction of the sending rate (X)
  3. increase of the retransmission timeout (t_{RTO})
- Effect on the connection
  1. lost packects during the disconnection
  2. lower sending rate upon reconnection
  3. additionally, poor adaptation to new network conditions (*e.g.* technology, congestion)

$\Rightarrow$ Based on the sender observations, we want to quantify the impact of disconnections on the connection performance.

Model of TFRC in Disconnected Scenarios

Evolution of the Sending Rate and the RTO

Time segmented in No-Feedback Intervals (NFI) of duration t_{RTO}.

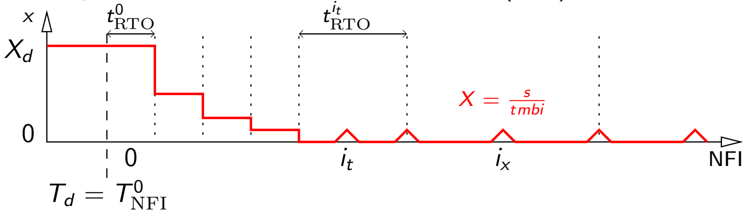


$$t^i_{\mathrm{RTO}} = \begin{cases} 4R & \text{if } i < i_t, \\ \frac{2s}{X^i} & \text{otherwise,} \end{cases}$$

$$i_t = \left\lceil \log_2 \frac{2R \cdot X_d}{s} \right\rceil \leq i_x.$$

NICTA

Time segmented in No-Feedback Intervals (NFI) of duration t_{RTO}.

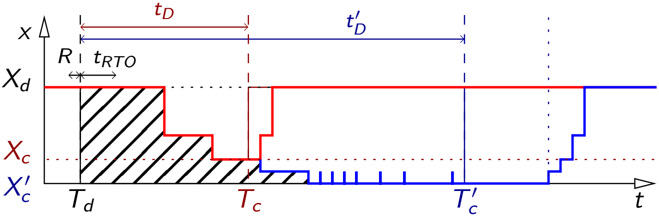


$$t^i_{\mathrm{RTO}} = \begin{cases} 4R & \text{if } i < i_t, \\ \frac{2s}{X^i} & \text{otherwise,} \end{cases} \qquad X^i = \begin{cases} \frac{X_d}{2^i} & \text{if } 0 \le i < i_x, \\ \frac{s}{t_{\mathrm{mbi}}} & \text{otherwise,} \end{cases}$$

$$i_t = \left\lceil \log_2 \frac{2R \cdot X_d}{s} \right\rceil \le i_x. \qquad i_x = \left\lceil \log_2 \frac{X_d \cdot t_{\mathrm{mbi}}}{s} \right\rceil.$$

Model of TFRC in Disconnected Scenarios

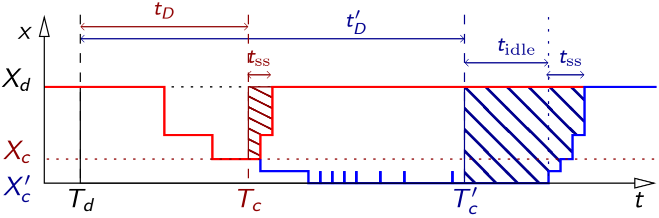
Number of Lost Packets over the Disconnected Period



$$n_{\text{lost}} = \begin{cases} \left\lfloor \frac{7}{8} \frac{t_D X^0}{s} \right\rfloor & (t_D \leq t^0_{\text{RTO}}) \\ \left\lfloor \frac{7}{8} \frac{t^0_{\text{RTO}} X^0}{s} + \sum_{i=1}^{i_D - 1} \frac{t^i_{\text{RTO}} X^i}{s} + \frac{t^{i_D}_{\text{RTO}} X^{i_D}}{2s} \right\rfloor & (\text{otherwise}) \end{cases} \tag{1}$$

Model of TFRC in Disconnected Scenarios

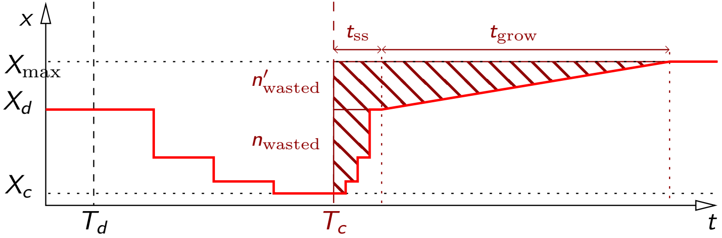
Amount of "Wasted" Bandwidth upon Reconnection



$$n_{\text{wasted}} = \frac{1}{s}\left(t_{\text{idle}} \cdot X_d + \sum_{i=0}^{n_{\text{ss}}} R_{\text{new}}\left(X_d - 2^i X_c\right)\right) \qquad (2)$$

Model of TFRC in Disconnected Scenarios

Additional "Wasted" Bandwidth on Bigger Networks



$$n'_{wasted} = \frac{1}{s}(X_{max} - X_d)(t_{idle} + t_{ss}) + \frac{R_{new}}{s} \sum_{i=0}^{n_{grow}} (X_{max} - X^i) \quad (3)$$

from \ to	UMTS	802.16	802.11 b	802.11 g
Packet losses (1)				
UMTS	306	236	226	224
802.16	2760	2614	2614	2614
802.11b	1080	1078	1078	1078
802.11g	2909	2907	2907	2907
Unused bandwidth (2) & (3) [500 B packets]				
UMTS	0	82938	263	109541
802.16	0	471	155	1029
802.11b	0	0	1085	54674
802.11g	0	0	0	4699

Link characteristics | Handoff times | Compare to simulation results

TFRC in disconnected scenarios and mobile handoffs

- more or less graceful handling of disconnections
- can be optimized by *e.g.*
  1. being given information about upcoming disconnections
  2. probing the network upon reconnection to adapt faster

TFRC in disconnected scenarios and mobile handoffs

- more or less graceful handling of disconnections
- can be optimized by *e.g.*
  1. being given information about upcoming disconnections
  2. probing the network upon reconnection to adapt faster

$\Rightarrow$ We propose such an addition to TFRC and implement it within DCCP.

Related work: Freeze-TCP can temporarily suspend a TCP connection

- in case of predictable disconnections on the receiving end
- rate restored to previous value when connectivity is back

Additional features: better support for mobility handoffs

sender-based freezing to account for mobile senders

slow-start-like probing for better capacity along the new path

NICTA

Freeze-DCCP/TFRC mechanism:

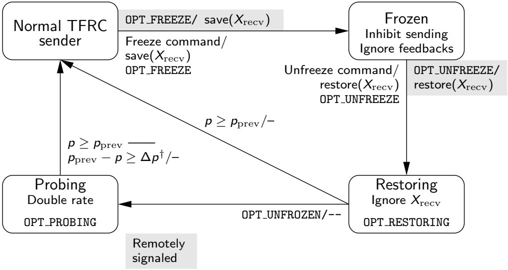
tight cooperation between the sender and the receiver using DCCP-level options

new states to support the unfreezing phase:

1. restoration of the rate or fallback to the newly computed value
2. probing the path for a higher capacity

Freeze-DCCP/TFRC

Additional states and options needed to support freezing

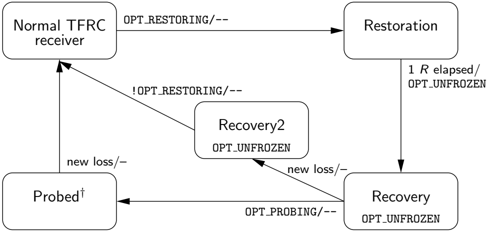


Sender

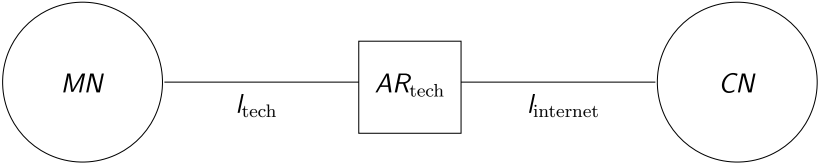
Drives the restoration process

Receiver

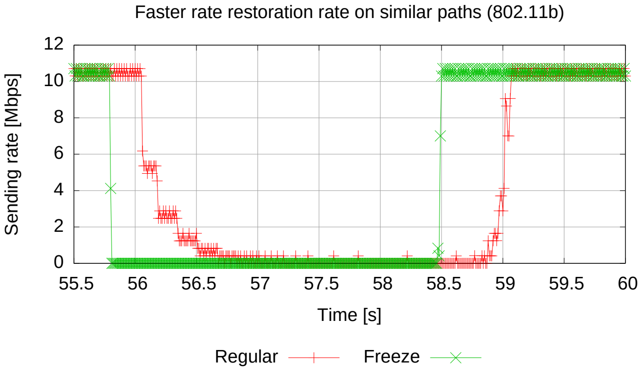
Ensures synchronisation

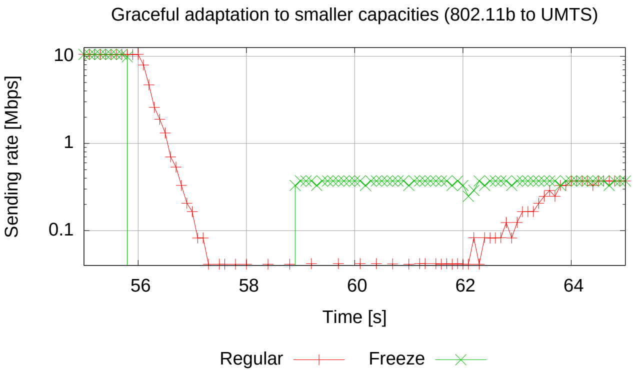


†When a packet is lost, the receiver computes and reports a p equivalent to the currently observed X_{recv}.

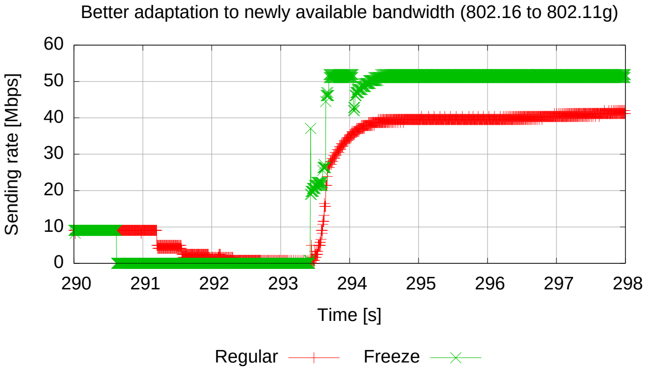


- ns-2 simulations for realistic networks
- l_{tech}, AR_{tech}: wireless network side
  - simulated using a wired link ▸ Link characteristics
- l_{internet}: wired internet
- disconnections using `$ns_ rtmodel-at $discotime_ down $ar_ $cn_` ▸ Handoff times


Faster rate restoration rate on similar paths (802.11b)
Sending rate [Mbps]
12
10
8
6
4
2
0
55.5
56
56.5
57
57.5
58
58.5
59
59.5
60
Time [s]
Regular
Freeze



- Note: logarithmic scale



- Though: the probing phase can still be improved.

Freeze-DCCP/TFRC

Performance Improvement of Freeze-DCCP over DCCP

NICTA

from \ to	UMTS	802.16	802.11 b	802.11 g
Packet losses (DCCP/TFRC only)				
UMTS	253.3	269.8	273.6	275.4
802.16	1732.3	1734.6	1734.6	1734.6
802.11b	856	855.5	855.3	855.3
802.11g	2470.9	2470.4	2470.2	2470.1
Unused bandwidth [500 B packets]				
UMTS	50.5	54018.05	2209.5	92156.1
	13.4	3607.9	9342.75	89328.6
802.16	12.45	1827.95	603.05	4185.75
	5	591.15	150.9	1520.35
802.11b	150.45	28314	2101.75	57970.65
	0	15278	47.45	1045.05
802.11g	42.5	2104.3	943.4	4313
	0	7172.75	46.5	188.45

▸ Link characteristics ▸ Handoff times ◂ Compare to analytical predictions

NICTA

- Single TCP flow from *AR* to *CN*
- Wait for settlement of rate upon reconnection
- 100 s samples afterwards

from \ to	UMTS	802.16	802.11 b	802.11 g
UMTS	0.6	0.3	0.2	0.1
802.16	1.6	1.3	1.1	0.9
802.11b	1.3	1	0.9	0.7
802.11g	1.5	1.2	1	1.1

- Values in $[0.5, 2]$ considered "reasonably fair"
- Closely similar to DCCP/TFRC in the same conditions

NICTA

Freeze-DCCP/TFRC

Better network usage when/as soon as it is available;

More flexible than Freeze-TCP:

- can accomodate a mobile sender;
- adapted to multiple network paths and technologies;

Mobility-aware transport protocol well suited for real-time traffic (*e.g.* VoIP or video streaming).

TCP fairness similar to regular TFRC

Future Work and Discussion

- Conclusion
  - model of TFRC in disconnected/mobility scenarios
  - Freeze-DCCP/TFRC
    - suspend the connection to avoid losses
    - restores the parameters to keep the previous rate
    - probes the new network to adapt faster
    - needs cross-layer information
    - reasonably TCP-fair
- Future work
  - Linux 2.6 implementation of Freeze-DCCP
    - experimentation over real wireless links
    - more thorough fairness evaluation
  - Cross-layer framework

Questions?

Thanks

olivier.mehani@nicta.com.au
http://www.nicta.com.au/people/mehanio/freezedccp

NICTA

Technology	Bandwidth [bps]	Delay [s]
UMTS	384 k	125 m
802.11b/g	11 M/54 M	10 m
802.16	9.5 M	40 m

◂ Mobility Requirements ◂ Scenario ◂ Simulation Results

NICTA

$$T_{\text{handoff}} = 2.5 + RTT_{\text{wireless}} + RTT_{\text{wired}}$$
$$= 2.6 + 2Delay_{\text{wireless}}$$

Destination network	T_{handoff} [s]
UMTS	2.85
802.16	2.68
802.11b/g	2.62

◂ Mobility Requirements ◂ Analytical Results ◂ Scenario ◂ Simulation Results