Can We Tackle Wi-Fi Rate Anomaly without Touching User Equipment?

THE UNIVERSITY *of* EDINBURGH
informatics

icsa | Institute for Computing Systems Architecture

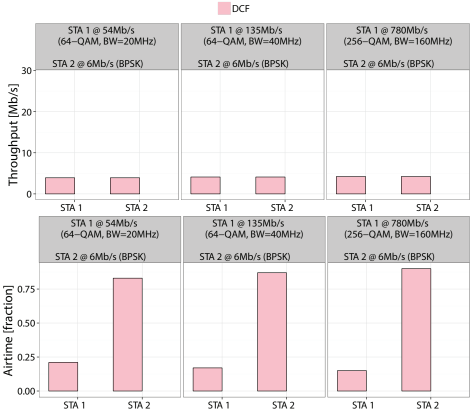
Paul Patras

w/ A. Garcia-Saavedra, D. Leith (TCD)
and D. Malone (NUIM)

3rd July 2015

802.11 performs poorly in multi-rate settings

- Decentralised channel access → users assigned equal transmission opportunities.
- Different bit rates employed to increase link reliability.
- Individual throughputs capped at that of the sluggish station → performance anomaly.



Stations transmitting at lower rates retain access to the channel for longer periods of time → network utility degrades.

ppatras@inf.ed.ac.uk

We knew this is a problem, but ...

No commercial solution exists that tackles rate anomaly!

Prior proposals suffer from at least one of the following:

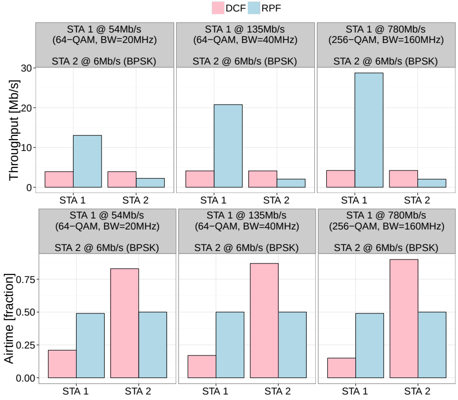
(i) Underlying analyses do not capture accurately the 802.11 details,

(ii) Naïvely assume error-free channel conditions,

(iii) Require modifications to user equipment at firmware level.

We can actually solve this

- Rigorous analysis of 802.11 operation with different packet lengths, bit rates, per-link error rates.
- Practical allocation scheme that can run on most APs* and does not require modifications to user equipment.



*OpenWrt Linux page reports 400+ supported devices – see http://wiki.openwrt.org/toh/

ppatras@inf.ed.ac.uk

Analytical model (sketch)

Setting $CW_{min,i} = CW_{max,i}$, the probability τ_i that a station i transmits in a randomly chosen slot time can be controlled:

$$\tau_i = \frac{2}{W_i + 1}. \tag{1}$$

The throughput obtained by station i:

$$S_i = \frac{p_{s,i} L_i}{P_e T_e + P_s T_s + P_u T_u}, \tag{2}$$

where

- $p_{s,i}$: probability of successful transmission of station i,
- L_i: length of the packet payloads generated by this station,
- P_e, P_s, P_u: expected probabilities that a slot is empty (idle), contains a success and an unsuccessful transmission (due to collision or channel errors),

Airtime

Denote T_{slot} the average slot duration ($T_{slot} = P_e T_e + P_s T_s + P_u T_u$).

Airtime – the fraction of time the channel is occupied by the (successful or unsuccessful) transmission of a station:

$$T_i = \frac{\tau_i}{T_{slot}} \left(\prod_{j=i+1}^{N} (1 - \tau_j) T_{s,i} + \sum_{j=i+1}^{N} \tau_j \prod_{k=j+1}^{N} (1 - \tau_k) T_{s,j} \right) \quad (3)$$

Introduce transformed variable $x_i = \tau_i / (1 - \tau_i)$ and
$X(x) = T_e + \sum_{j=1}^{N} \left(T_{s,j} x_j \prod_{k=1}^{j-1} (1 + x_k) \right)$.

ppatras@inf.ed.ac.uk

Proportional-fair Allocation

Find the solution to the following optimisation problem

$$\max_{\tilde{x}} \sum_{i=1}^{N} \tilde{S}_i \tag{4}$$

$$s.t.\ \tilde{S}_i - \tilde{z}_i - \tilde{x}_i + \log X - \log L_i \leq 0,\ \ i = 1, 2, .., N \tag{5}$$

where we use the log-transformed variables
$\tilde{x}_i = \log x_i,\ \tilde{z}_i = \log(1 - p_{n,i}),\ \tilde{S}_i = \log S_i$

Proportional-fair Allocation

The Lagrangian is

$$L = -\sum_{i=1}^{N} \tilde{S}_i + \sum_{i=1}^{N} \lambda_i \left(\tilde{S}_i - \tilde{z}_i - \tilde{x}_i + \log X - \log L_i \right)$$

Solving the KKT conditions, we obtain

$$T_i = \frac{1}{N}, \forall i, \tag{6}$$

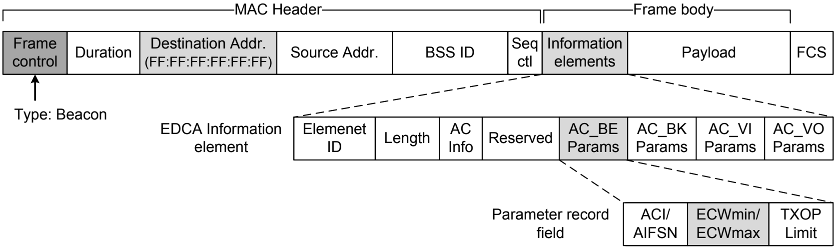
The solution to the proportional-fair allocation optimisation problem assigns equal airtime to all nodes.

Obtain τ_i by solving numerically a system of N equations and N unknowns, then compute the CW configuration of each station.

Implementation

Key idea:

- The AP broadcasts through beacons the set of contention parameters to be used by *ALL* stations.



- Setting $CW_{min} = CW_{max}$ is straightforward.
- *'UNICAST'* beacons work to configure each node with different CW to control their TX attempt rate.

Implementation

Proposed approach – AP side (*no changes to the stations!*):

- *Kernel-space modifications*[*]
  - Estimate $T_{s,i}$ by inspecting 'length' and 'rate' fields from the headers of correctly received frames.
  - Report statistics to user-space through `debugfs`.
  - Make copies of broadcast frame, update with MAC address and CW for each station.
  - Queue all 'unicast' beacons after the broadcast to avoid overwriting.
- *User-space optimisation tool*
  - Python script to parse statistics gathered by driver
  - Solve the optimisation task using GNU Octave

[*] `https://bitbucket.org/agsaaved/unicast-beacon`

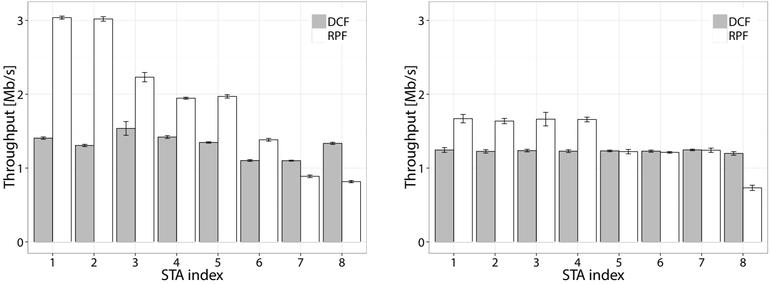
Experimental Evaluation

Test bed:

- 9x Soekris net6501-70 embedded PCs equipped with Compex WLE300NX-6B wireless cards (Atheros AR9390 chipset)
- 1 acting as AP, the others as clients
- Ubuntu 14.04 (kernel version 3.13), `mac80211` and `ath9k` driver.
- 5GHz frequency band, channel 149 (5.745GHz)

Uplink Data Traffic

- 8 backlogged stations TX at {54, 48, 36, 24, 18, 12, 9, 6} Mb/s



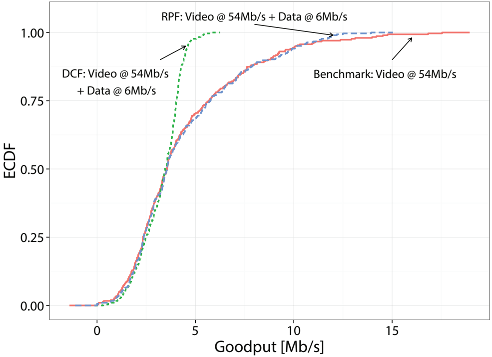
UDP upload　　TCP upload

- With RPF, faster stations improve throughput by up to 120%; clients transmitting at inferior rates only marginally affected.
- Network utility improved by 100%.

ppatras@inf.ed.ac.uk

Video Streaming

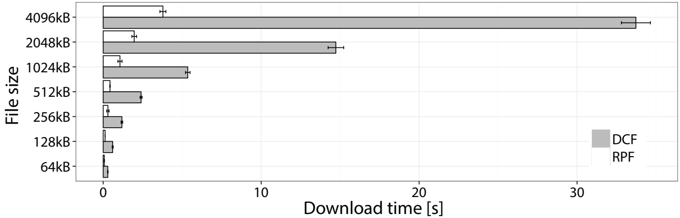
- One station performing TCP upload at a 6Mb/s.
- Second station streaming video over HTTP with 54Mb/s bit rate.



- With RPF the capacity of the video link is enhanced to ensure perfect quality of the video streaming experience.

Small File Download

- One station uploads a large file over TCP at a 6Mb/s.
- Second client downloads multiple small files over a link at 54Mb/s.
- Files with sizes between 64KB and 4096KB, retrieve with `wget`.



- Reduction of the download times by 315% for the smallest object and by 790% for the largest.

Detailed analysis and more interesting results

http://dx.doi.org/10.1016/j.adhoc.2015.06.002