Skype Video Responsiveness to Bandwidth Variations

L. De Cicco, S. Mascolo, V. Palmisano

Dipartimento di Elettronica ed Elettrotecnica

Politecnico di Bari – Italy
Motivation 1/2

Multimedia real-time applications – i.e. Voice/Video over IP, P2P TV, Joost - can tolerate small losses but are time sensitive, i.e. TCP is not appropriate.

But TCP/IP has congestion control that has been fundamental for preserving Internet stability. It must be used in a resource shared system such as the Internet.

TCP has been extremely successful for elastic data traffic, which is not sensitive to delays – i.e. reliable delivery is achieved through retransmissions

So what should we do with real-time traffic?
Multimedia flows “are made elastic” within a certain range using adaptive codecs (Speex, H.264, On2,...)

Congestion control is (should be) implemented at application level over UDP
Motivations
Related works
Experimental testbed and tools employed
Experimental Results
Goals of the work

• Investigate the behaviour of Skype Video to discover to what extent is able to cope with network congestion by matching network available bandwidth

• In particular we will investigate:
  • How Skype is able to adapt to available bandwidth variations
  • The degree of elasticity of the flows (i.e. minimum bitrate, maximum bitrate)
  • The dynamics of the algorithm (responsiveness, transients)
  • How Skype does throttle its sending rate
  • How Skype Video flows share the bottleneck i.e. fairness
  • Are Skype Video flows TCP friendly?
Related Works

Congestion control for multimedia applications

- Several proposals: TFRC, RAP, TEAR, ARC
- TFRC represents the only IETF standardization effort (RFC3448), however there is no evidence that it is implemented in any leading application

Skype

- **Congestion control**: Skype Audio flows implement some sort of congestion control algorithm (De Cicco et al. WWIC07, Tec. Report Submitted)
- **QoS provided by Skype**: MOS and PESQ measurements under different network conditions (Barbosa et al, NOSSDAV '07), or by defining packet level metrics (Chen et al, SIGCOMM '06)
- **Detection of Skype flows**: by using two classifiers it is possible to detect Skype calls on-line (Bonfiglio et al., SIGCOMM 07)
Experiments are performed in a controlled testbed that emulates WAN scenarios.

We measure instantaneous values (every 0.4s) of throughput, loss rate and goodput for each flow by looking at the input of the bottleneck queue.

Testbed Settings:
- RTT=50ms
- Queue size=BDP
A tool has been developed in order to generate **reproducible experiments**.

Video flows are generated by hijacking the video input (/dev/video) by using a modified version of the GStreamer plugin `gst-fakevideo`.

The Foreman YUV test sequence has been used as input to `gst-fakevideo`.
Detailed measurement of the variables shown in the Skype “Technical Call Infos” tooltip is obtained by using a modified version of QT libraries we have developed.

In this way we are able to automatically log and plot:

- RTT, Jitter, video resolution, video frame rate, estimated sent and received loss percentages:

```
ObjID: 62
Codec: SVOPC
Jitter: 113
Packet loss: 1,0% (12)
Send packet loss: 0,4%/0,6%
Recv packet loss: 1,5%/2,0%
Roundtrip: 30ms
BM: audio 1250 / 64 ms  video 4438 corr 3%
SessionOut: RELAY_UDP (3195 packets)
SessionIn: RELAY_UDP (1211 packets)
Relays: 4
UDP status local:Good remote:Good
CPU usage: 62,0% 27,9% hicc:2
Video send:FPS 8 (cam:26 bw:8 cpu:66(25) rcv:53(0 0)) cmp 9 cpu 58 320×240 PS 0.193 Kbit
```

Skype employs the *Video Codec Truemotion 7* (VP7) developed by On2.

On2 claims to adapt encoding bitrate by throttling:
- Frame quality
- Video resolution
- Frame rate (fps)

Minimum bitrate declared by On2 is 20Kbps, no information about maximum bitrate
In order to characterize Skype Video flows we have designed and carried out a set of different experiments (here we present a subset):

**Main characteristics of Skype video flows**
- Skype response to a step variation of available bandwidth
- Skype response to staircase variations of available bandwidth

**Fairness issues**
- Two Skype Video flows over a square wave available bandwidth

**TCP friendliness**
- One Skype Video flow with two concurrent TCP flows
**Skype response to a step variation of available bandwidth (1/2)**

- **Link capacity**: step-like, acts at t=50s with min. value 160 Kbps and max. value 2000 Kbps (four runs are shown).

- **Experiment duration**: 500s

  **First part (0<t<50):**
  - Throughput is 80 kbps, well below 160kbps limit.
  - Frame rate starts at 15fps and decreases to 10fps

  **Second part (t>50):**
  - Throughput increases in around 100s to an avg value of ~450Kbps
  - Frame rate increases and oscillates around 15 fps
Skype response to a step variation of available bandwidth (2/2)

When loss events occur (grey line represents cumulative lost bytes) packet size (black points) doubles.

- We infer that Skype employs a FEC scheme to counteract losses that is activated after a large loss event

  - Results of the experiment
    - Skype flows react to available bandwidth variations (100s transient)
    - maximum bitrate around 450Kbps
    - FEC action activated on large loss events
**Skype response to staircase variations of available bandwidth** (increasing/decreasing [160,1000]kbps)

*Link capacity:* varies in the range [160,1000]kbps in order to show the granularity of the rate adaptation (each step is 168Kbps and lasts 100s)

*Experiment duration:* 1000s

- The steady state is reached at time $t=700s$
- Frame rate decreases until $t_A$ where the resolution is decreased to 160x120 so that the frame rate can increase using the same bandwidth

Skype Video Responsiveness to Bandwidth Variations
S. Mascolo

NOSSDAV ’08
Braunschweig, Germany

-14-
**Link capacity:** varies from 160Kbps down to 20Kbps (thin link), step size 40Kbps, step duration 50s

**Experiment duration:** 400s

- Sending rate is able to adapt to small variations (see average throughput)
- Call is dropped at $t=375s$ because a very large packet drop percentage is detected
- Minimum available bandwidth is **40Kbps** (compatible with the value declared by On2)
Two Skype Video flows over a square wave available bandwidth

**Link capacity**: square wave, min value 160Kbps (using lower values calls were dropped), max value 384Kbps (UMTS downlink capacity), period 400s.

**Experiment details**: duration 800s; second call is placed at t=50s.

First half (0<t<400):
At t=90 S1 starts to leave bandwidth to S2. S2 increases its sending rate until t=200 where link capacity is exceeded and the rate is reduced

Second half (400<t<800)
The two flows are not able to saturate the link (so quality is not the best possible)
A good fairness is obtained (JFI=0.97, see also frame rate)
Channel utilization is poor
One Skype Video flow with two concurrent TCP flows

**Link capacity:** constant 384Kbps (UMTS downlink capacity)

**Experiment details:** duration 1000s; Skype starts at t=0, TCP1 at t=200s, TCP2 at t=400s

Skype flow releases a bandwidth share to TCP1 at t=200s

Bandwidth is shared in a fair way among the three flows for t>400 except for the interval [550, 700]s

Packet size doubles (FPS remain unchanged) within that interval indicating FEC action is activated due to losses.

**Summary (t>400s)**

<table>
<thead>
<tr>
<th></th>
<th>Tput (kbps)</th>
<th>Loss rate (kbps)</th>
<th>Loss ratio</th>
<th>Channel util.</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>162.5</td>
<td>6.0</td>
<td>3.7%</td>
<td>42.3%</td>
</tr>
<tr>
<td>TCP1</td>
<td>101.6</td>
<td>12.3</td>
<td>12%</td>
<td>26.4%</td>
</tr>
<tr>
<td>TCP2</td>
<td>102.3</td>
<td>12.6</td>
<td>12%</td>
<td>26.6%</td>
</tr>
</tbody>
</table>
Conclusions

- Skype Video flows react to bandwidth variations.
- Packet size, frame rate, and video resolution are used to throttle the sending rate.
- Skype Video flows are elastics within the range [40, 450]Kbps.
- Large transient times are required to adapt to a bandwidth increment.
- Best quality is not achieved, in the sense that the encoder does not saturate when bandwidth is available (too conservative).
- Skype Video seems more aggressive than TCP due to FEC that increases bandwidth consumption even when losses are detected.


