Reducing Costs in the Personal Cloud: Is BitTorrent a Better Bet?

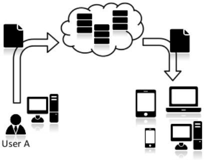
Rahma Chaabouni, Marc Dánchez-Artigas and Pedro García-López

Viktor Hangya

Wednesday 19th November, 2014

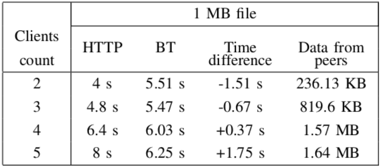
Motivation

- Cloud storage
  - Http vs. BitTorrent
  - BitTorrent is more effective for large files
  - 99% of the files are smaller than 16 MB
- Switch protocol
  - benefit from the common interest of users (devices)
  - offload the cloud server
- In the paper
  - comparison of Http and BitTorrent protocols
  - dynamic switching algorithm

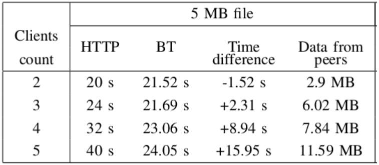


Download Times

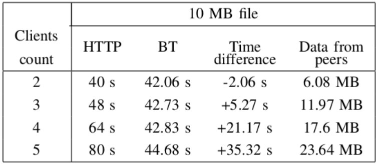
- seed (server) upload speed 5 Mbps
- client download/upload speed 2/1 Mbps



Clients count	1 MB file			
	HTTP	BT	Time difference	Data from peers
2	4 s	5.51 s	-1.51 s	236.13 KB
3	4.8 s	5.47 s	-0.67 s	819.6 KB
4	6.4 s	6.03 s	+0.37 s	1.57 MB
5	8 s	6.25 s	+1.75 s	1.64 MB



Clients count	5 MB file			
	HTTP	BT	Time difference	Data from peers
2	20 s	21.52 s	-1.52 s	2.9 MB
3	24 s	21.69 s	+2.31 s	6.02 MB
4	32 s	23.06 s	+8.94 s	7.84 MB
5	40 s	24.05 s	+15.95 s	11.59 MB



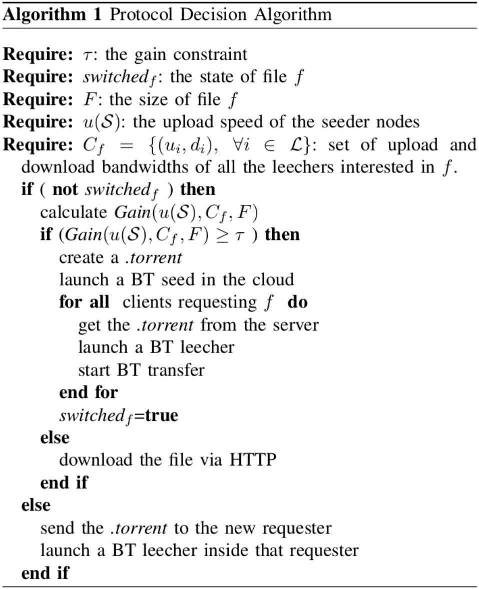
Clients count	10 MB file			
	HTTP	BT	Time difference	Data from peers
2	40 s	42.06 s	-2.06 s	6.08 MB
3	48 s	42.73 s	+5.27 s	11.97 MB
4	64 s	42.83 s	+21.17 s	17.6 MB
5	80 s	44.68 s	+35.32 s	23.64 MB

Metrics

- $Gain = \frac{T_{cs} - T_{bt}}{T_{cs}}$
  - $T_{cs} = \frac{F}{\min\{d_{min}, \frac{u(\mathtt{S})}{L}\}}$
  - $T_{bt} = \frac{F}{\min\{d_{min}, \frac{u^{'}(\mathtt{I})}{L}, u(\mathtt{S})\}} + \alpha_{bt}$

- $Offload = 1 - \frac{data\ from\ cloud}{total\ data\ sent}$

Algorithm

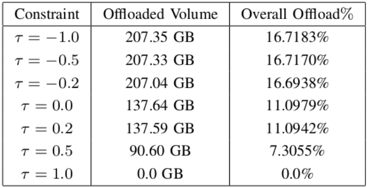


Algorithm 1 Protocol Decision Algorithm

```
Require: $\tau$: the gain constraint
Require: $switched_f$: the state of file $f$
Require: $F$: the size of file $f$
Require: $u(\mathcal{S})$: the upload speed of the seeder nodes
Require: $C_f = \{(u_i, d_i),\ \forall i \in \mathcal{L}\}$: set of upload and
   download bandwidths of all the leechers interested in $f$.
   if ( not $switched_f$ ) then
      calculate $Gain(u(\mathcal{S}), C_f, F)$
      if ($Gain(u(\mathcal{S}), C_f, F) \geq \tau$ ) then
         create a .torrent
         launch a BT seed in the cloud
         for all clients requesting $f$ do
            get the .torrent from the server
            launch a BT leecher
            start BT transfer
         end for
         $switched_f$=true
      else
         download the file via HTTP
      end if
   else
      send the .torrent to the new requester
      launch a BT leecher inside that requester
   end if
```

Results

- Ubuntu One trace
  - 30 hours of log
  - 1,887,247 files
  - 32.67% download
  - total download volume 1,240.25 GB
  - 90% of the files are less than 1 MB
- Settings:
  - seed upload speed 2 Mbps
  - client download/upload speed 1/0.5 Mbps
- this is just simulation!
- with $\tau = -1.0$, 450$ can be saved per month ($3.000 per month originally)



Constraint	Offloaded Volume	Overall Offload%
$\tau = -1.0$	207.35 GB	16.7183%
$\tau = -0.5$	207.33 GB	16.7170%
$\tau = -0.2$	207.04 GB	16.6938%
$\tau = 0.0$	137.64 GB	11.0979%
$\tau = 0.2$	137.59 GB	11.0942%
$\tau = 0.5$	90.60 GB	7.3055%
$\tau = 1.0$	0.0 GB	0.0%

Conclusion

- dynamic switching algorithm
  - server can benefit from the upload bandwidth of the clients
  - everybody can benefit if the server is not overloaded
- Sync
  - P2P file synchronization based on BitTorrent
  - unlimited space