CS305 · Computer Networking

Final Project – Video CDN

何泽安 12011323
冷子阳 12011513
龚明道 12011204

Contents

1 Group Info

Project contribution ratio: He Zean 33%, Leng Ziyang 33%, Gong Mingdao 33%.

He Zean: Compile and Output, Log File, Basic Proxy, Bit Rate Adaption, DNS, Danmaku.

Leng Ziyang, Gong Mingdao: Report, Netsim Test.

2 Implementation

2.1 Exit the Proxy

There's a timer counting down the program's exiting. On every request of video chunk, the counter will be reset to 30 sec, and once the last video fragment was loaded, the counter will be set to 10 sec. If the client do nothing in the counting down period, it will exit. However, using the <ctrl + c> is also safe, since they all callback a function to release all the resources before the program is acturally exited.

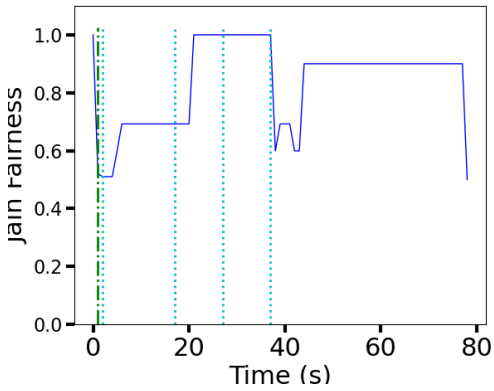
```
def suicide_daemon():
    if app.debug:
        return

    global countdown
    if countdown <= 0:
        log(f'{datetime.now()}  Proxy is inactivate: timeout 30s without request | 10s after the video is f
        os.kill(os.getpid(), signal.SIGINT)
    else:
        with countdown_lock:
            countdown -= 1
        threading.Timer(1, suicide_daemon).start()

# ...
atexit.register(release_res)
```

3 Net Simulation Result & Analysis

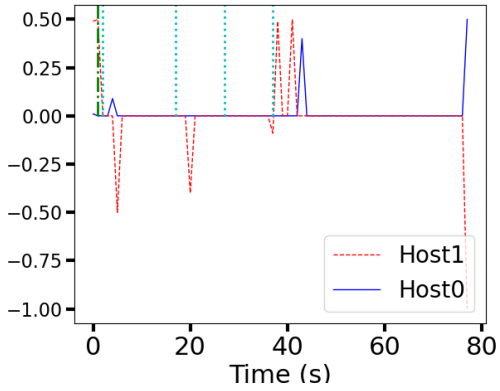
3.1 One link

3.1.1 One link with alpha 0.1 & 0.5 result

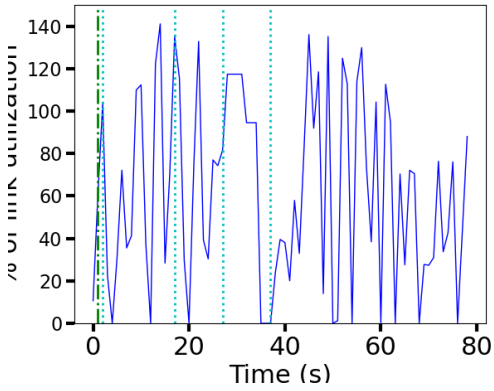
```
1654072811.089829 link_1 2000
1654072826.097545 link_1 1000
1654072841.105518 link_1 500
1654072851.112338 link_1 100
1654072861.122667 link_1 2000
```



Since the alpha of the proxy1 is much smaller than of proxy2, which means the bit rate selection of proxy2 will be more sensitive to the real time throughput of the network environment. And since all of them are competing for the same link, we can see a gap between each time the link throughput changes, representing the unblanced throughput for the proxies due to their different change scheme of bit rate. And after a short time of selection, the fairness will raise again, as all of them merged to the same bit rate.



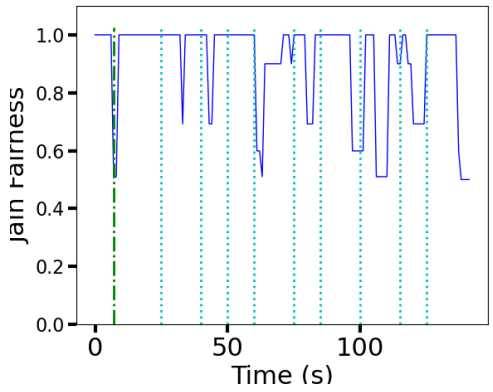
And from the smoothness graph, the proxy1's peak comes after proxy2's, since modification of bit rate is slower with proxy1's alpha.



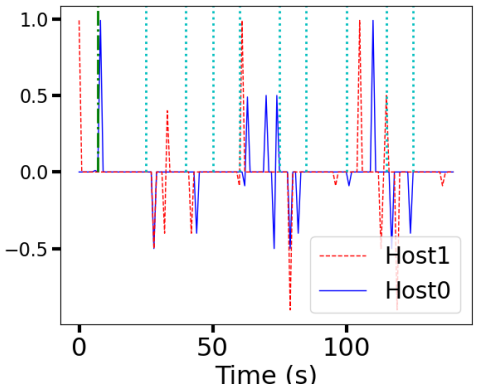
Each time the network througput changes, it will cause a turbulence to the utilization of the link, representing the two proxys competing for the same link, and as the network environment gets steady, the competition will be weaker.

3.1.2 One link with alpha 0.5 & 0.9 result

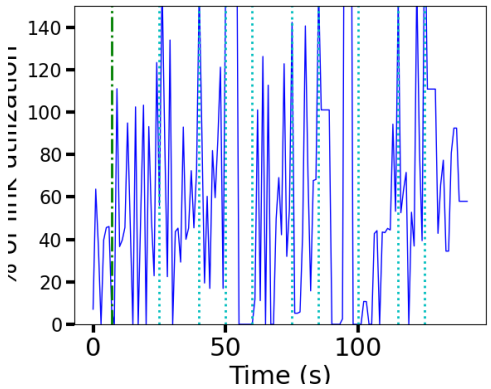
```
1654073585.137387 link_1 3000
1654073620.172610 link_1 1000
1654073635.182608 link_1 500
1654073645.192394 link_1 50
1654073655.203004 link_1 2000
1654073670.212772 link_1 500
1654073680.224222 link_1 50
1654073695.240330 link_1 2000
1654073710.252513 link_1 500
1654073720.263808 link_1 100
```



With this time, the fairness graph is much more obvious comparing the the above one. As the fairness drops to 0.5 after network environment changes and maintains 1 after gets steady. But comparing to the above situation, the recover of the curver takes much shorter time, with the alpha being large means the switch time between bit rate in the proxys be much shorter and more responsive.



It's still consistent with the above smoothness that the proxy1 mostly changes after proxy2 changes, which clearly illustrate the proxy with larger alpha changes more responsively.

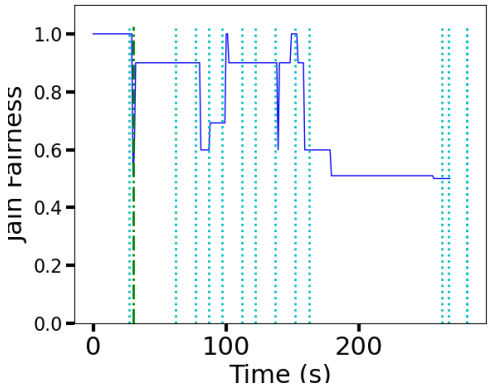


From this graph, it's more obvious that the utilizatio of the link gets higher as the proxies gets steady, which is the post period of bit rate seletion, where both proxy switch to the same bit rate. And it dropped when the network environment changed and proxys competing for the link.

3.2 Two links

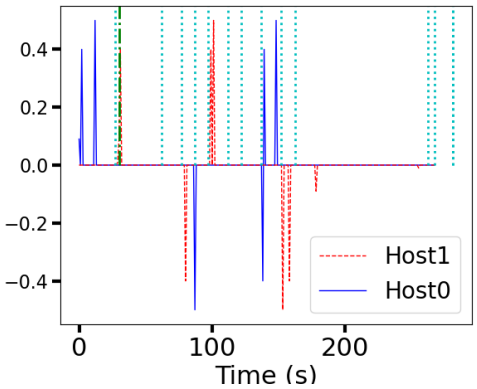
3.2.1 Two links with alpha 0.1 & 0.5 result

```
1654074380.286222 link_1 2000
1654074380.287065 link_2 2000
1654074415.322985 link_1 1000
1654074450.359054 link_2 1000
1654074465.375011 link_1 500
1654074475.385880 link_1 50
1654074485.392335 link_1 2000
1654074500.408266 link_2 500
1654074510.415779 link_2 50
1654074525.424626 link_2 2000
1654074540.432341 link_1 500
1654074550.443490 link_1 100
1654074650.542510 link_1 500
1654074655.548752 link_1 250
1654074669.872284 link_1 1000
1654074669.873276 link_2 1000
```

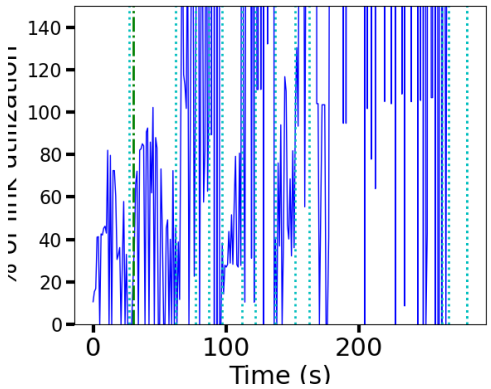


When two links are applied, the data gets much different comparing to the one link situation. The most noticeable one is that fariness no longer gets to 1 after network environment changes. This is quite obvious that changing the throughput of one link will cause no impact on the other, resulted in the fairness

graph strictly follow the actaul througput of the two links.



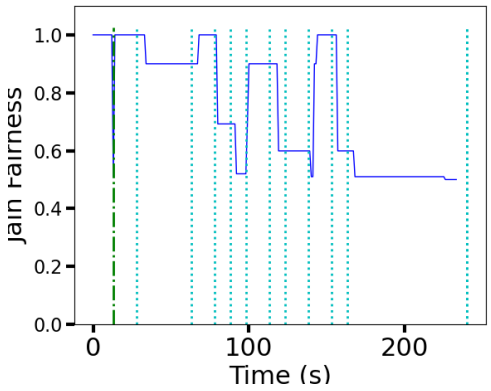
And since two links will not impact each other, the change of bit rate also strictly follows the change of throughput on each link.



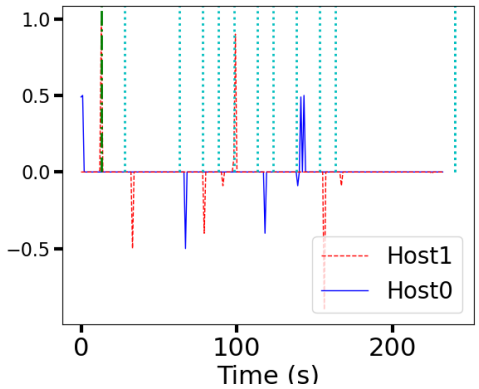
Here the utilization seems overall higher than the previous one link scenario, though with much turbulance. As the two link don't compete with each other, all together of them can reach a higher utilization without doubt.

3.2.2 Two links with alpha 0.5 & 0.9 result

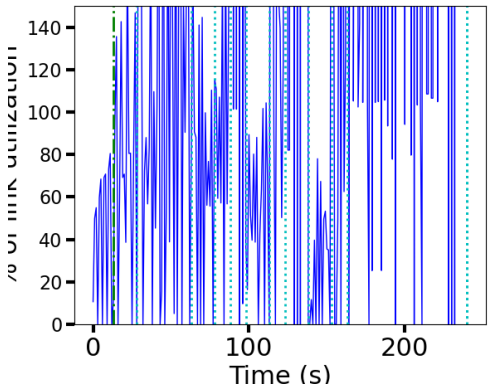
```
1654074903.749470 link_1 2000
1654074903.750093 link_2 2000
1654074938.785906 link_1 1000
1654074973.812964 link_2 1000
1654074988.822735 link_1 500
1654074998.832883 link_1 50
1654075008.839442 link_1 2000
1654075023.856416 link_2 500
1654075033.862546 link_2 50
1654075048.878889 link_2 2000
1654075063.892846 link_1 500
1654075073.902519 link_1 100
1654075150.691448 link_1 1000
1654075150.692309 link_2 1000
```



Compare with the fairness of two links which alpha of 0.1 and 0.5, the fairness of the current situation is much more responsive and changes in a short time, due to the more sensitivity of the proxies to the throughput change.



Compare with the above one, we can notice that the value of each peak gets larger and duration of peak gets smaller as the proxies changes bit rate faster to the compatible throughput.

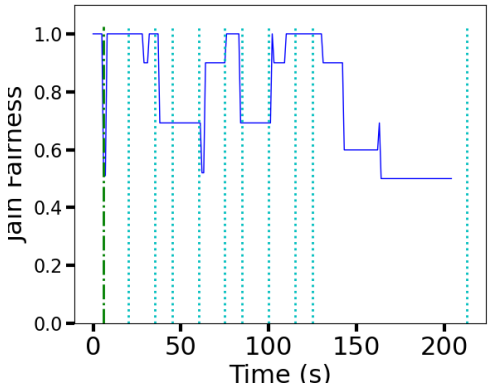


The utilization differs from time to time hugely, representing that two links have more redundency time where the links are not utilized by others. Therefore implies that the links can be further utilized if plan the communication on them correctly.

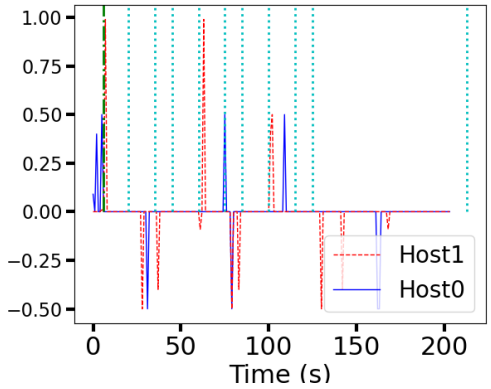
3.3 Share link

3.3.1 Share link with alpha 0.1 & 0.5 result

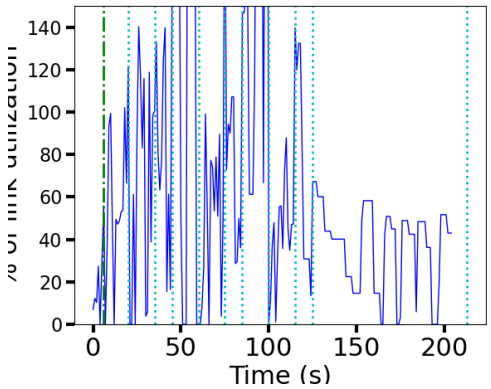
```
1654075714.351472 link_1 3000
1654075749.368361 link_1 1000
1654075764.384202 link_1 500
1654075774.388314 link_1 50
1654075789.398813 link_1 3000
1654075804.408739 link_1 500
1654075814.420254 link_1 50
1654075829.436030 link_1 3000
1654075844.441531 link_1 500
1654075854.452262 link_1 100
1654075942.891051 link_1 1000
```



As it's similar to the one link scenario where the fairness drops and recover to 1 since they are sharing the same link.



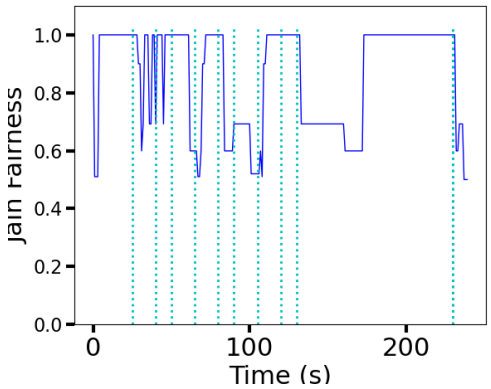
And it gets more obvious that the proxy with smaller alpha follows the change of the proxy with large alpha.



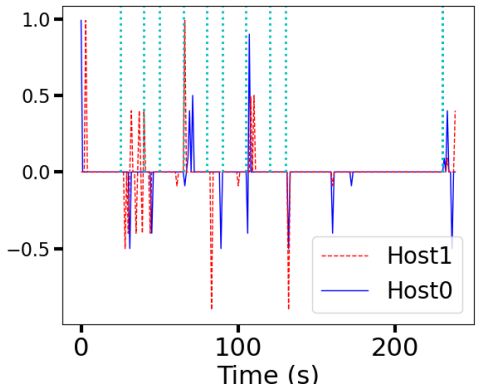
Though there still exists competition, the overall utilization of the link is higher than one link version, due to the systemetic plan for the sending and recieving of packets.

3.3.2 Share link with alpha 0.5 & 0.9 result

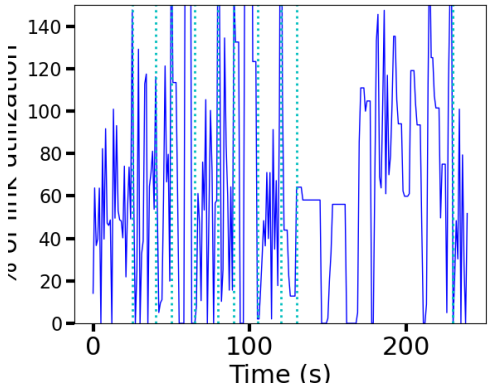
```
1654076170.605882 link_1 3000
1654076205.618374 link_1 1000
1654076220.634422 link_1 500
1654076230.645694 link_1 50
1654076245.661917 link_1 3000
1654076260.668412 link_1 500
1654076270.679468 link_1 50
1654076285.695649 link_1 3000
1654076300.711663 link_1 500
1654076310.718316 link_1 100
1654076410.819303 link_1 500
1654076410.820030 link_1 1000
```



This time the fairness gets even more fair, with two of the proxies adapting to the throughput quickly and reduced competition and congestion with the share link.



It is noticeable that the distance between the two peaks gets smaller with the alpha overall larger, representing responsiveness.



The utilization reveals less 0 time, and overall increase in the utilization after the rapid changing of bit rate and share link control.

4 Conclusion

As it's obvious that with larger alpha the proxy gets more responsive and sensitive to the throughput, and lower alpha means less changing during the small network turbulations. On one side, we hope the proxies response to the througput change more quickly so that it won't cause congestion to the network and user won't experience buffering, also helps better to adapt to the link situation with more utilization. But on the other side as the proxies get more responsive, it may cause more turbulance to the overall network, with proxies competing links with each other, some noise loop through the system and may gets amplified and effect the steadyness of the system. Therefore choosing the right alpha among the 0 and 1 may help proxies brings more stablity and higher utilization to the link, also better user experience to the client who are using them.

Forwarding Strategies and Queueing PoliciesIn addition to keeping the delivery predictability, a node may also keep an exponential weighted moving average (EWMA) of the delivery predictability. The EWMA is then used to make forwarding decisions and to report to peering nodes, but the value calculated is still used as input to the calculations of new delivery predictabilities. The EWMA is calculated according to

$$P_{ewma} = P_{ewma\_old} * (1 - \alpha) + P * \alpha$$

where $0 \leq \alpha \leq 1$ is the weight of the most current value.

The appropriate choice of alpha may vary depending on application scenario circumstances. Unless prior knowledge of the scenario is available, it is suggested that alpha is set to 0.5.