1. Transport layer: UDP, TCP



Application layer: DNS



1. 2RTTx + RTT1 + RTT2+…+ RTTk + dt
2. 1. 64 RTTx + RTT1 + RTT2+…+ RTTk
   2. 8 RTTx + RTT1 + RTT2+…+ RTTk



* 1. 3 RTTx + RTT1 + RTT2+…+ RTTk

1. Total time to download for non-persistent parallel connection: (400/1600 + Dp + 400/1600 + Dp + 400/1600 + Dp + 300000/1600 + Dp) + (400/(1600/8) + Dp + 400/(1600/8) + Dp + 400/(1600/8) + Dp + 300000/(1600/8) + Dp)

= 1694.25 + 8\*Dp



Total time to download for persistent connection: (400/1600 + Dp + 400/1600 + Dp + 400/1600 + Dp + 300000/1600 + Dp) + 8 \* (400/1600 + Dp + 300000/1600 + Dp)

= 1690.25 + 20\*Dp



**Answer:**

There wasn’t a significant difference in time spent for the non-persistent and persistent connections (the persistent connection was only a little faster). Therefore, it would make sense to download via parallel instances of non-persistent HTTP.



1. 1. Yes, because James’s parallel connections utilize a greater proportion of the bandwidth. When James uses a parallel connection, the objects are loaded simultaneously and the delays overlap



* 1. No, not necessarily because all the new parallel connections will get an unfair allocation of bandwidth and might encounter bottlenecks.



1. 1. Transmission time= 0.0213

Traffic intensity = 0.9798

Access delay = 1.054



**Average response time =** 1.054 + 2.5 = 3.554 seconds

* 1. Access delay = 0.087



**Average response time =** (0.087 + 2.5) \* 0.25 = 0.8425 seconds