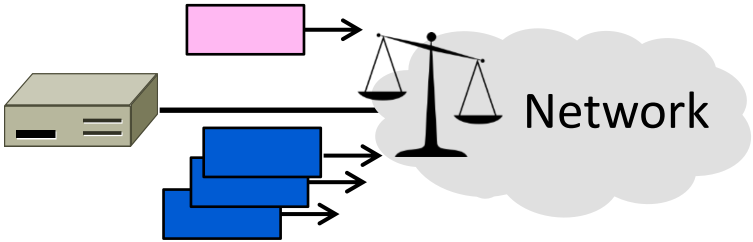
Topic

- Sharing bandwidth between flows
  - WFQ (Weighted Fair Queuing)
  - Key building block for QOS

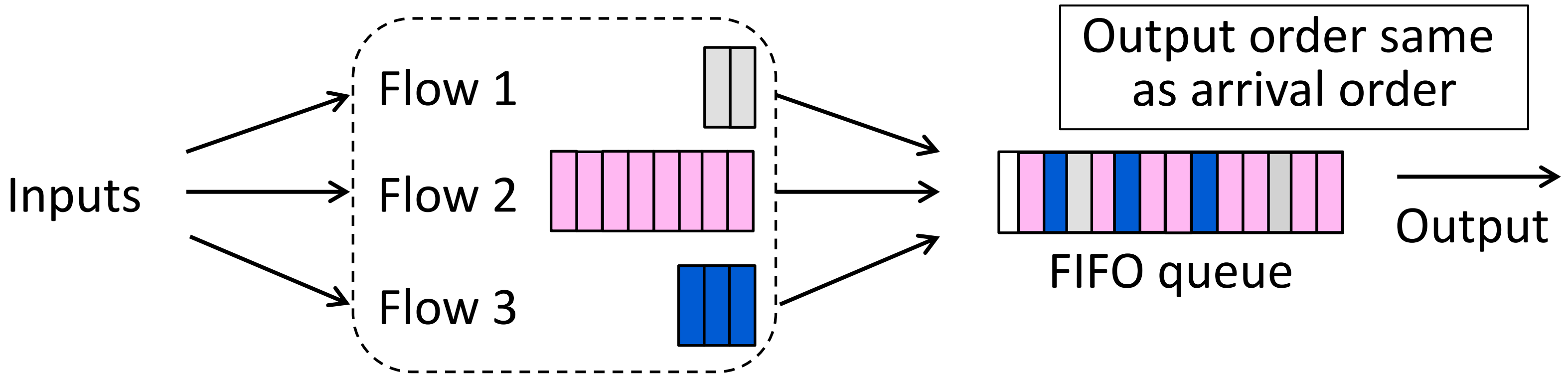


Sharing with FIFO Queuing

- FIFO "drop tail" queue:
  - Queue packets First In First Out (FIFO)
  - Discard new packets when full
  - Typical router queuing model
- Sharing with FIFO queue
  - Multiple users or flows send packets over the same (output) link
  - What will happen?

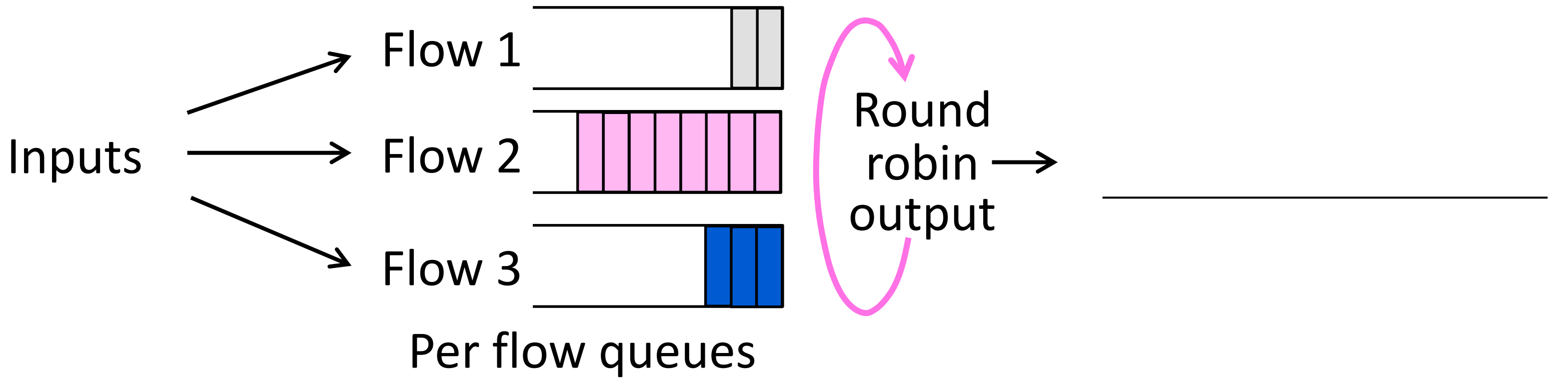
Sharing with FIFO Queuing (2)

- Bandwidth allocation depends on behavior of all flows
  - TCP gives long-term sharing – with delay/loss, and RTT bias
  - Aggressive user/flow can crowd out the others



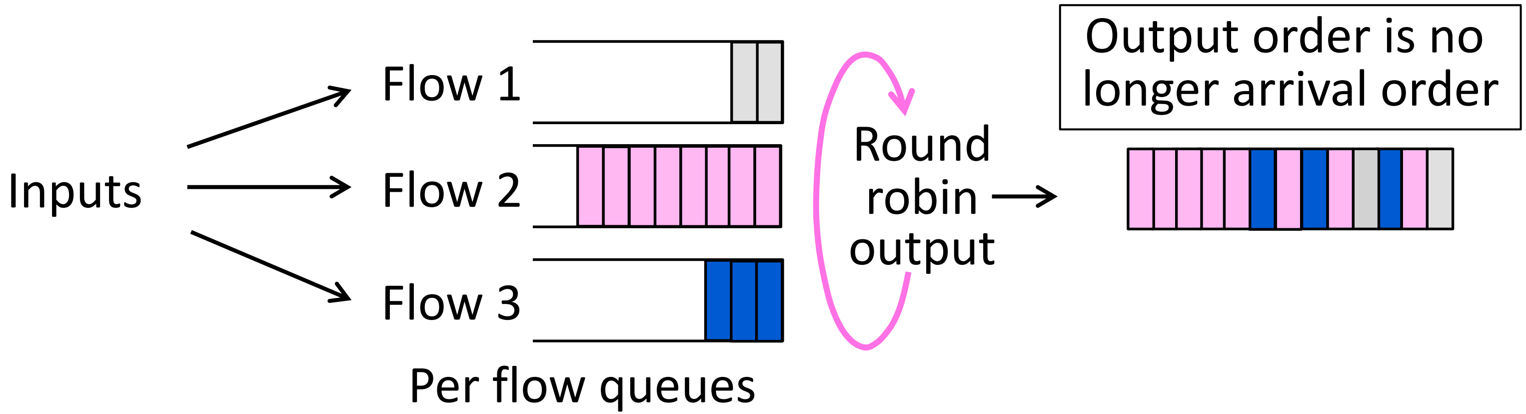
Round-Robin Queuing

- Idea to improve fairness:
  - Queue packets separately for each flow; take one packet in turn from each non-empty flow at the next output time



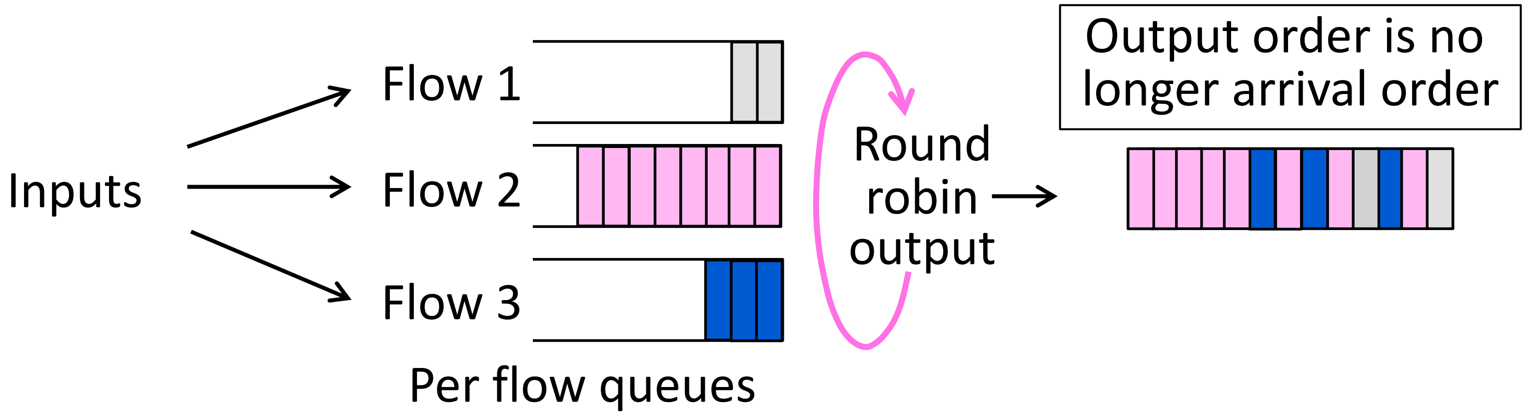
Round-Robin Queuing (2)

- Idea to improve fairness:
  - Queue packets separately for each flow; take one packet in turn from each non-empty flow at the next output time
  - How well does this work?



Round-Robin Queuing (3)

- Flows don’t see uncontrolled delay/loss from others!
- But different packet sizes lead to bandwidth imbalance
  - Might be significant, e.g., 40 bytes vs 1500 bytes



Fair Queuing

- Round-robin but approximate bit-level fairness:
  - Approximate by computing virtual finish time
  - Virtual clock ticks once for each bit sent from all flows
  - Send packets in order of their virtual finish times, $\text{Finish}(j)_F$
  - Not perfect – don't preempt packet being transmitted

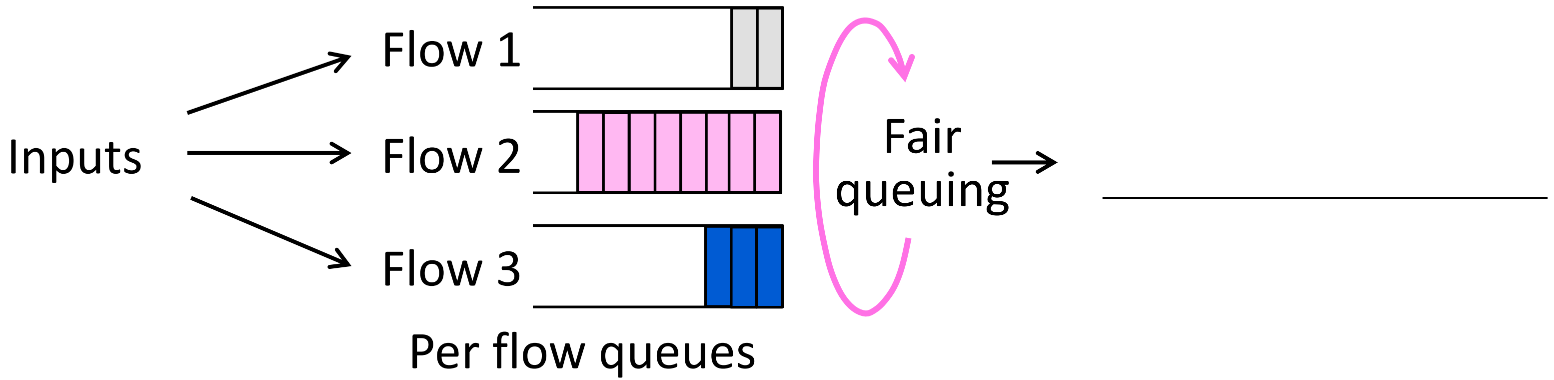
$\text{Arrive}(j)_F$ = arrival time of j-th packet of flow F

$\text{Length}(j)_F$ = length of j-th packet of flow F

$\text{Finish}(j)_F = \max(\text{Arrive}(j)_F, \text{Finish}(j-1)_F) + \text{Length}(j)_F$

Fair Queuing (2)

- Suppose:
  - Flow 1 and 3 use 1000B byte packets, flow 2 uses 300B packets
  - What will fair queuing do?



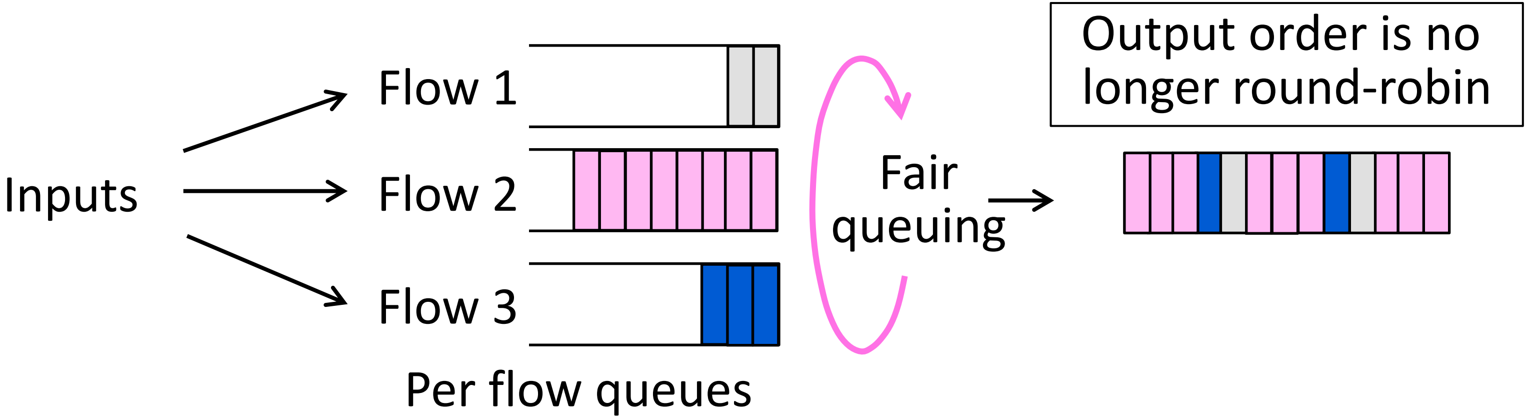
Fair Queuing (3)

- Suppose:
  - Flow 1 and 3 use 1000B packets, flow 2 uses 300B packets
  - What will fair queuing do?

Let $\text{Finish}(0)_F=0$, queues backlogged $[\text{Arrive}(j)_F < \text{Finish}(j\text{-}1)_F]$
$\text{Finish}(1)_{F1}=1000$, $\text{Finish}(2)_{F1}=2000$, …
$\text{Finish}(1)_{F2}=300$, $\text{Finish}(2)_{F2}=600$, $\text{Finish}(3)_{F2}=900$, 1200, 1500, …
$\text{Finish}(1)_{F3}=1000$, $\text{Finish}(2)_{F3}=2000$, …

Fair Queuing (4)

- Suppose:
  - Flow 1 and 3 use 1000B byte packets, flow 2 uses 300B packets
  - What will fair queuing do?



WFQ (Weighted Fair Queuing)

- WFQ is a useful generalization of Fair Queuing:
  - Assign a weight, Weight_F, to each flow
  - Higher weight gives more bandwidth, e.g., 2 is 2X bandwidth
  - Change computation of $\text{Finish}(j)_F$ to factor in Weight_F

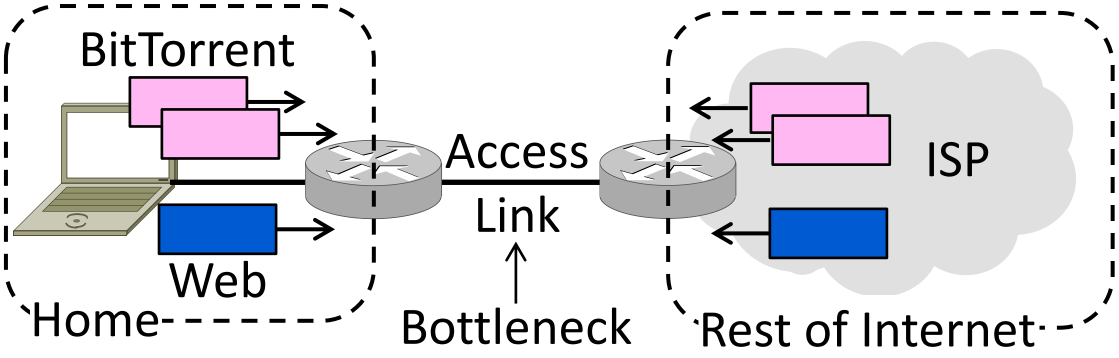
$\text{Arrive}(j)_F$ = arrival time of j-th packet of flow F

$\text{Length}(j)_F$ = length of j-th packet of flow F

$\text{Finish}(j)_F = \max(\text{Arrive}(j)_F, \text{Finish}(j-1)_F) + \text{Length}(j)_F / \text{Weight}_F$

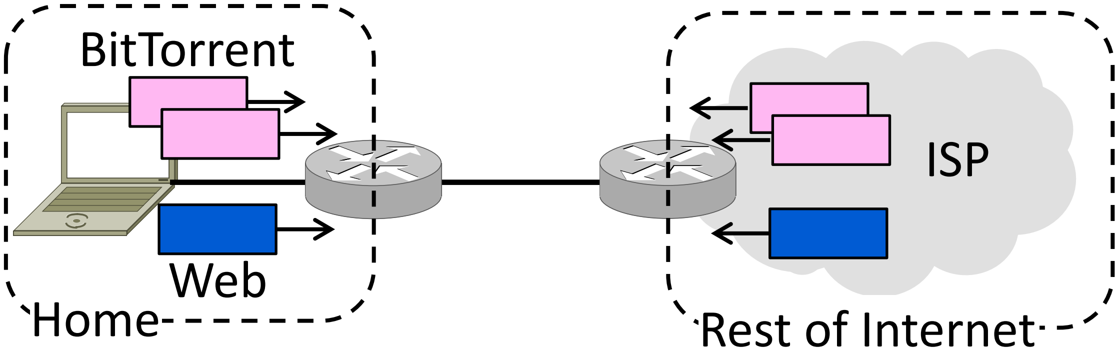
Example – Web and BitTorrent

- Home user browses the Web and runs BitTorrent at the same time
  - Assume access link is the bottleneck
  - What happens? What do we want?



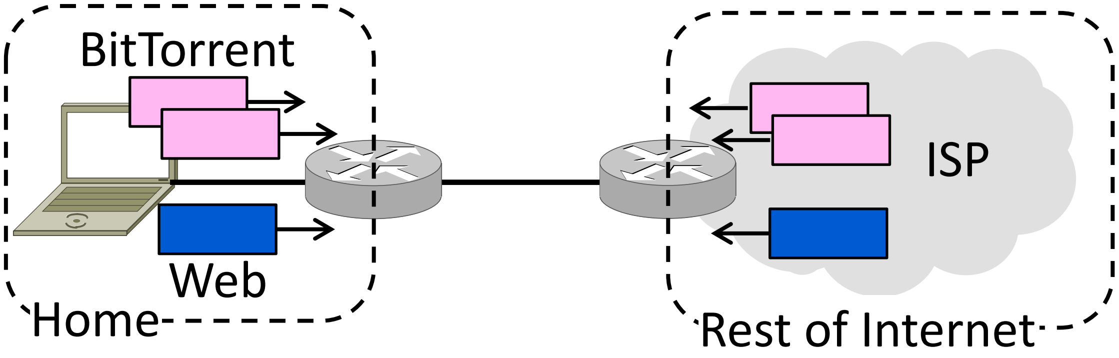
Web and BitTorrent (2)

- What happens?
  - Web and BitTorrent compete for downstream bandwidth using TCP
  - Queues build at ISP end of access …



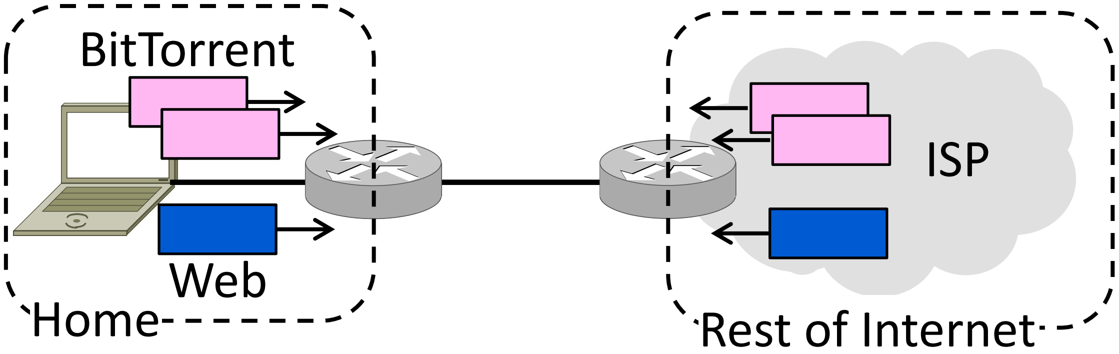
Web and BitTorrent (3)

- What happens?
  - Web PLT rises because of BitTorrent
  - Less web bandwidth, more delay/loss



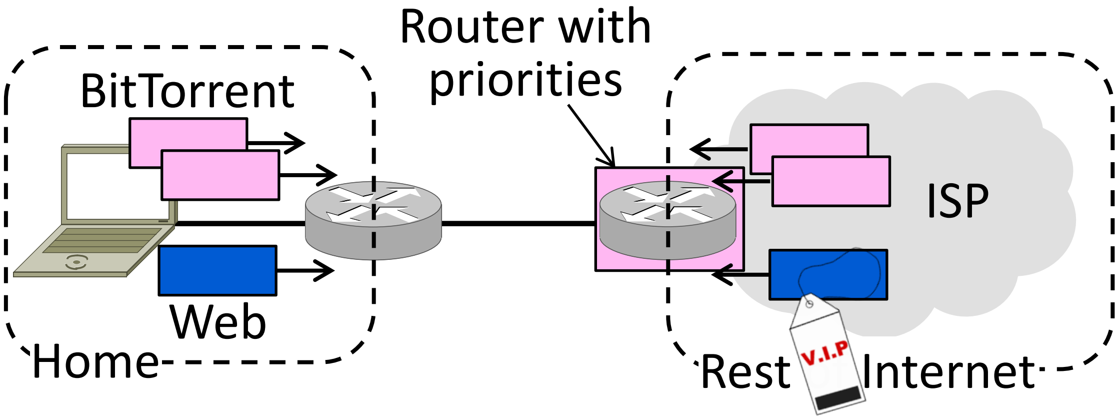
Web and BitTorrent (4)

- What do we want to happen?
  - Web is interactive, while BitTorrent runs in the background
  - Prefer to use bandwidth for Web



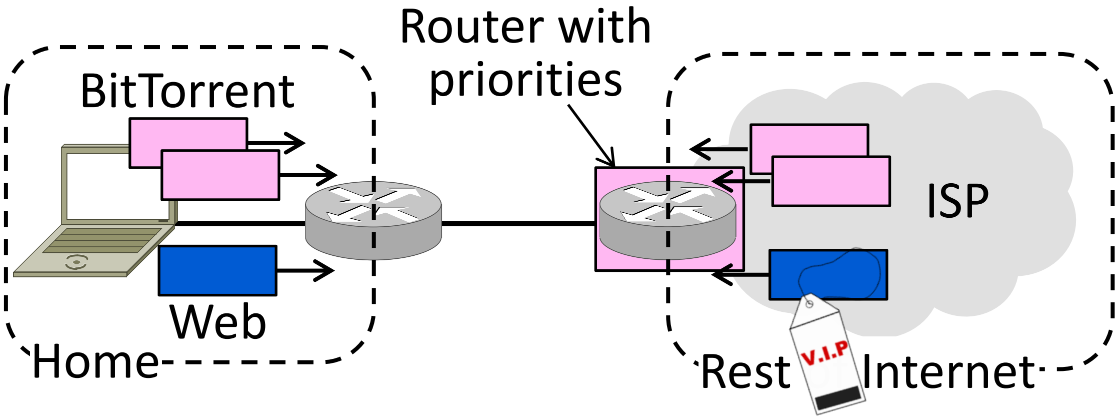
Web and BitTorrent (5)

- What do we want to happen?
  - Suppose we modify ISP router to give priority to Web packets on access link



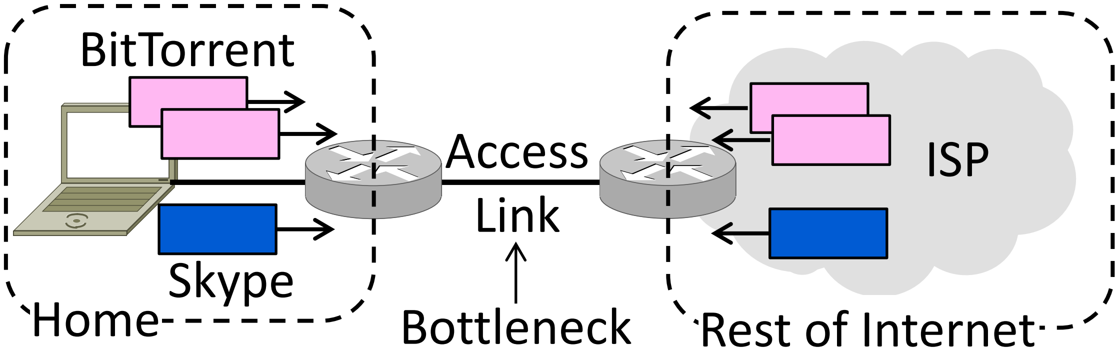
Web and BitTorrent (6)

- What do we want to happen?
  - Would minimize web PLT for user
  - BitTorrent just has less bandwidth



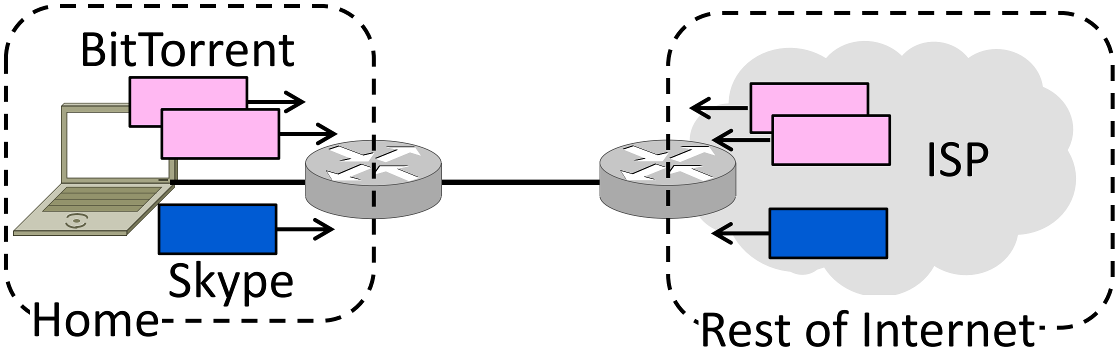
Example – Skype and BitTorrent

- Home user skypes (VoIP only) and runs BitTorrent at the same time
  - Assume access link is the bottleneck
  - What happens? What do we want?



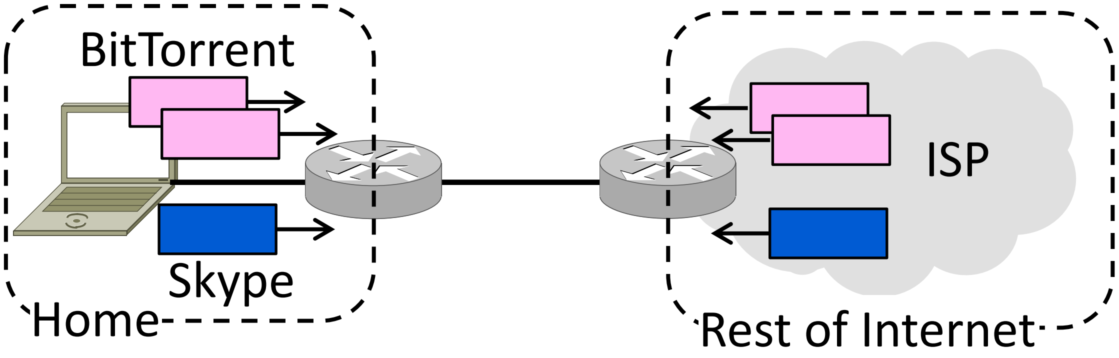
Skype and BitTorrent (2)

- What happens?
  - Skype and BitTorrent compete as before, though not with TCP
  - Queues build at ISP end of access …



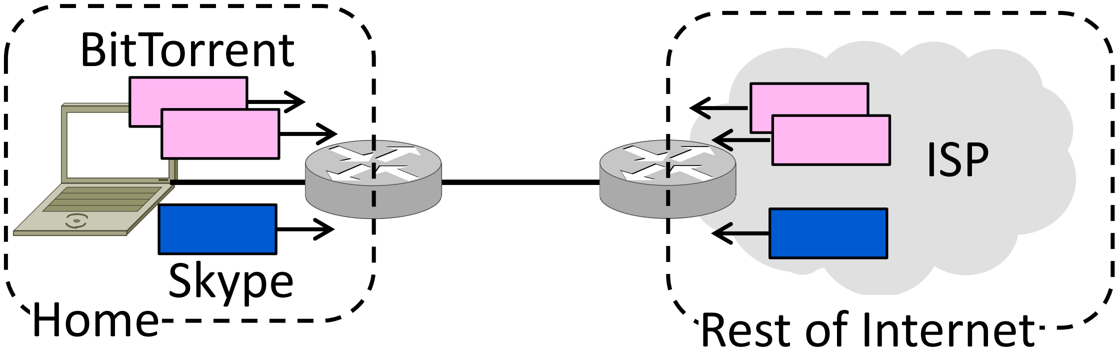
Skype and BitTorrent (3)

- What happens?
  - Skype call quality falls due to BitTorrent
  - More delay/loss; little bandwidth issue



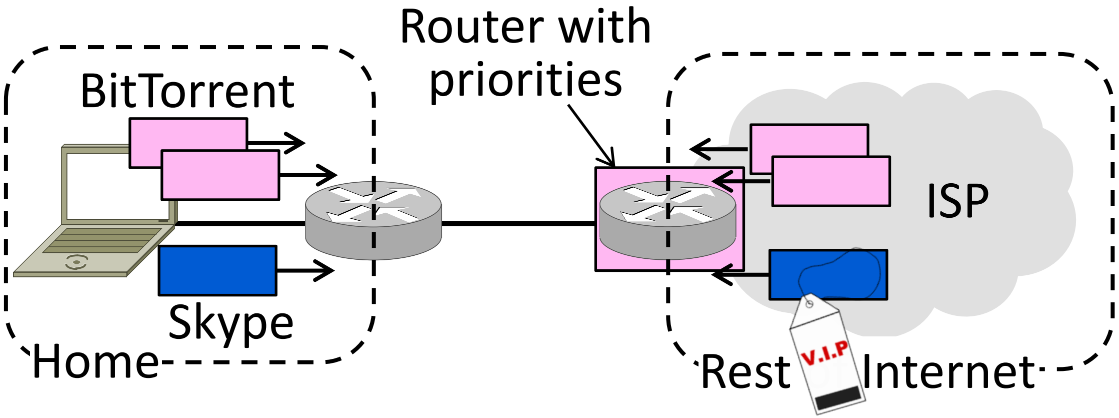
Skype and BitTorrent (4)

- What do we want to happen?
  - Skype real-time, BitTorrent background
  - Prefer low-delay for Skype and high-bandwidth for BitTorrent



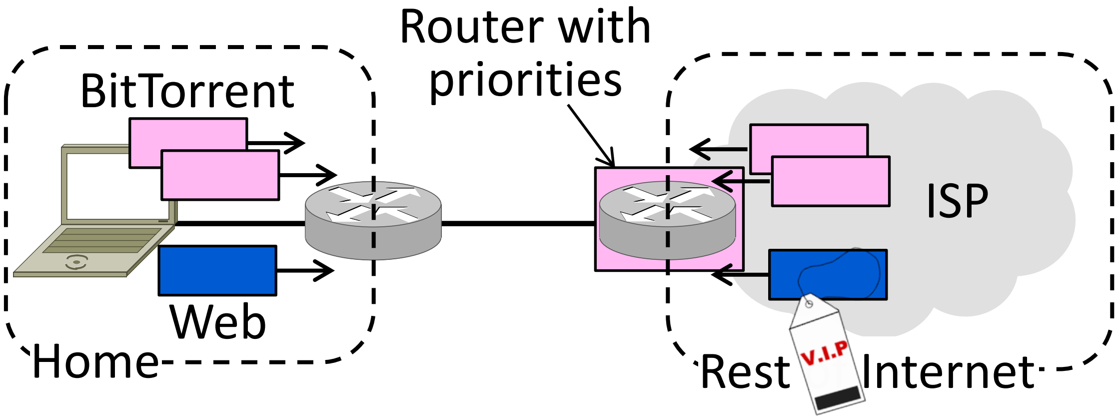
Skype and BitTorrent (5)

- What do we want to happen?
  - Modify ISP router to give priority to Skype packets on access link



Web and BitTorrent (6)

- What do we want to happen?
  - Maximizes skype call quality without slowing BitTorrent – both win!



QOS Motivation (2)

- Opportunity to allocate bandwidth to improve app/user performance
  - Guarantee bandwidth to an app
  - Satisfy multiple apps at once

- To provide QOS, we need to know what apps require of the network
  - Need for bandwidth, delay, loss

Application Requirements

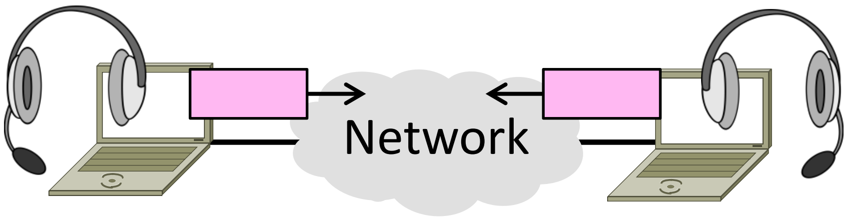
- HIGH stringency means high bandwidth, low delay/loss

Application	Bandwidth	Delay	Jitter	Loss
Email	Low	Low	Low	Medium
File sharing	High	Low	Low	Medium
Web access	Medium	Medium	Low	Medium
Remote login	Low	Medium	Medium	Medium
Audio on demand	Low	Low	High	Low
Video on demand	High	Low	High	Low
Telephony	Low	High	High	Low
Videoconferencing	High	High	High	Low

Variation in delay

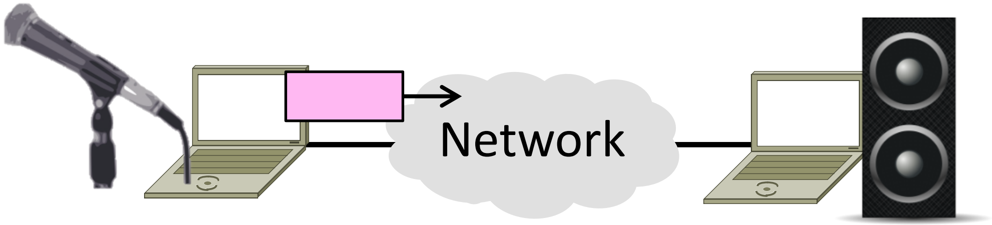
Topic

- Sending interactive real-time media over the network, e.g., VoIP
  - Using the best effort Internet
  - Playout buffer technique



Challenge – Network Delay

- Consider one direction
  - Constant rate of media is generated at source, later consumed at receiver
  - Network must have enough bandwidth, and adds a delay

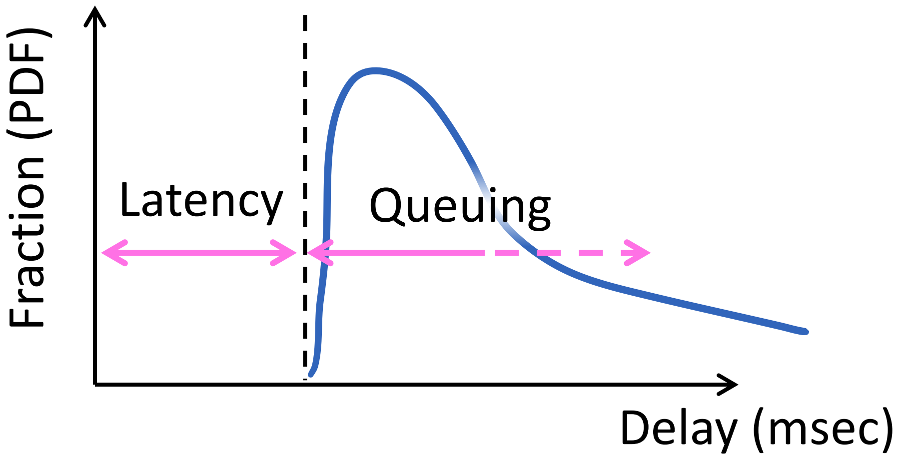


Microphone

Speaker

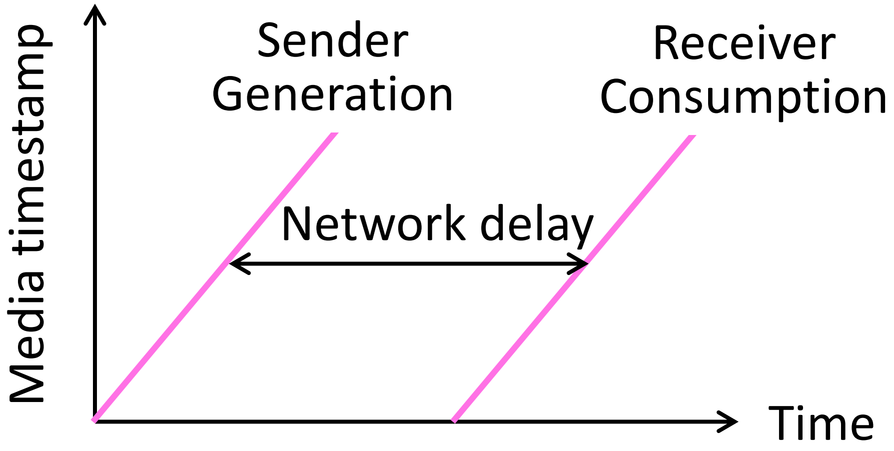
Network Delay (2)

- Network delay is variable
  - Message latency plus queuing delay
  - Variability in delay is called <u>jitter</u>



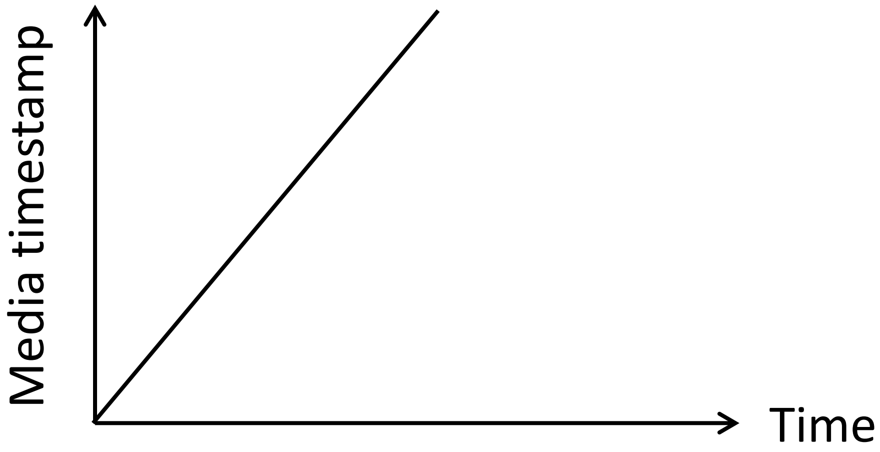
Playout

- Ideally want fixed, and small network delay for interactivity
  - Emulate the telephone network



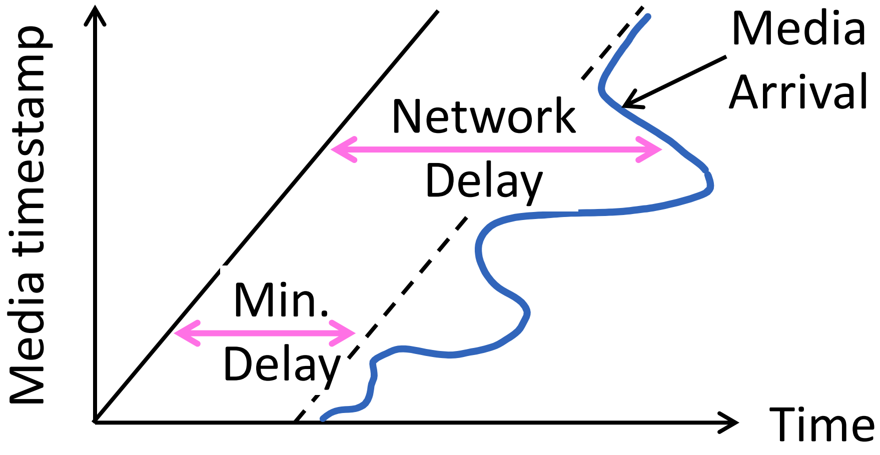
Playout (2)

- Media arrives at receiver after variable network delay



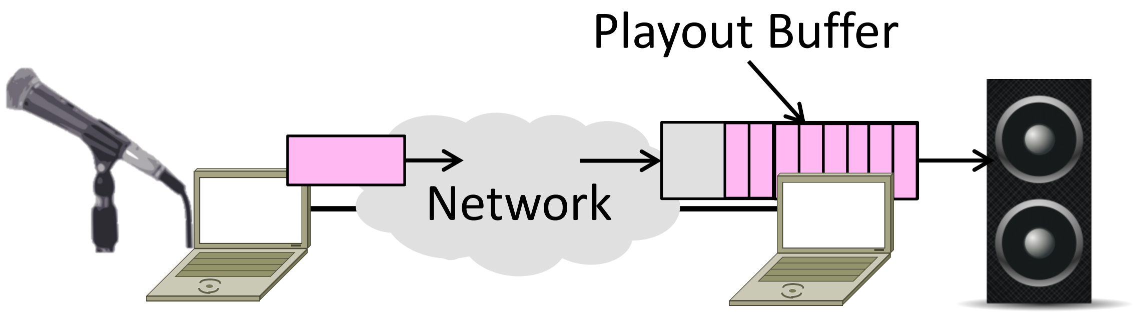
Playout (3)

- Media arrives at receiver after variable network delay



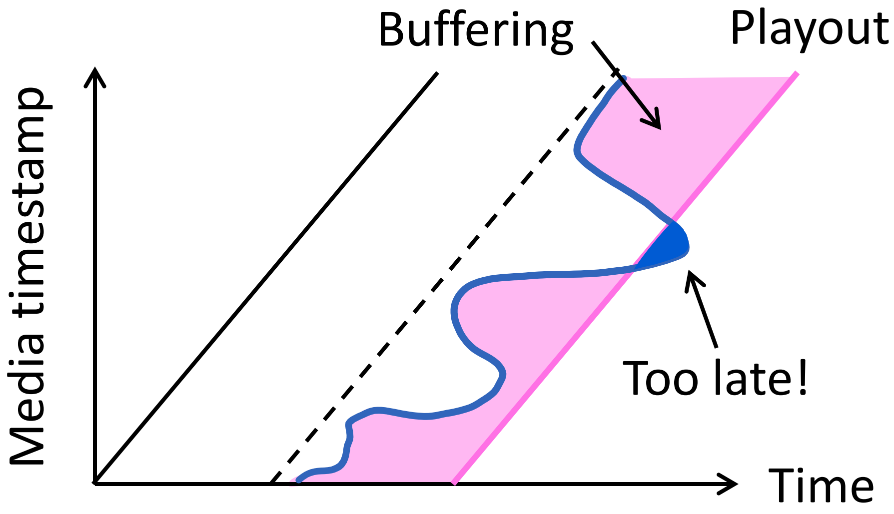
Playout Buffer

- Put media in playout buffer at receiver until consumption time
  - Smooth out variable network delay



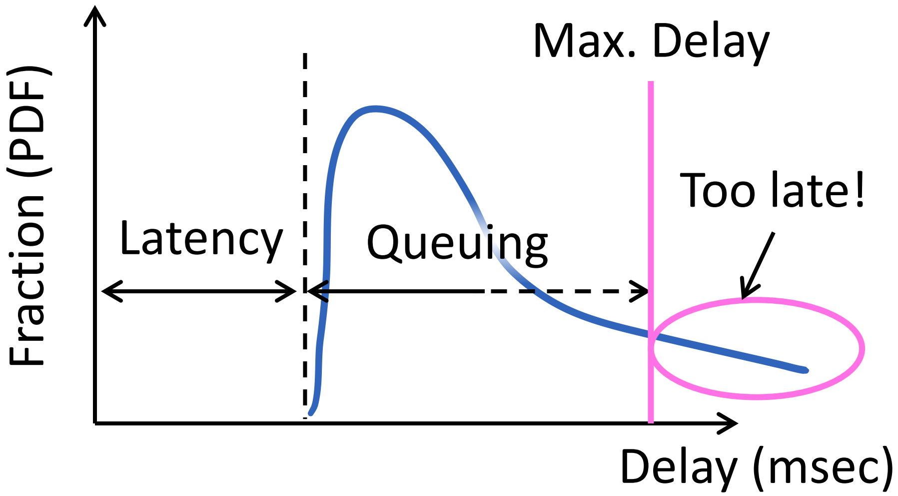
Playout Buffer (2)

- Media arrival curve determines time in playout buffer and deadline



Playout Buffer (3)

- Pick largest acceptable network delay to set the playout point



Playout Buffer (4)

- Tradeoff:
  - Larger acceptable network delay
    → larger buffer/delay, less loss
  - Smaller acceptable network delay
    → smaller buffer/delay, more loss

- Typically can't recover loss for interactive, real-time scenario
  - Instead, do without (glitch)

Topic

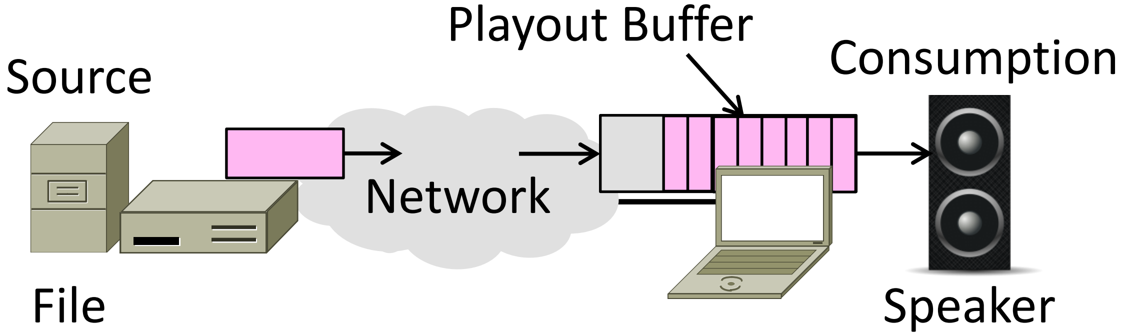
- Playback of media over the network
  - Using the best effort Internet
  - YouTube, Netflix, etc.
  - Huge usage!

Streamed vs. Interactive Media

- Streamed is less demanding case:
  - Only a single direction to consider
  - Low delay not essential; affects startup but not interactivity
  - Still need to handle bandwidth, jitter

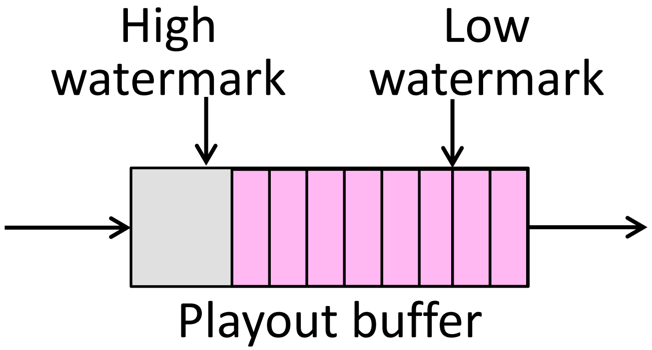
Handling Jitter

- As before, buffer media at receiver until ready for playout time
  - Smooth out variable network delay



Handling Jitter (2)

- Use HIGH and LOW watermarks to control source over/underfill
  - Start pulling media at low level
  - Stop pulling media at high level



Handling Bandwidth

- Send file in one of multiple encodings
  - Higher quality encodings require more bandwidth
  - Select best encoding given available bandwidth

Higher quality
More bandwidth

(JPEG example)

Lower quality
Less bandwidth

15:1

23:1

46:1

144:1

Streaming over TCP or UDP?

- UDP minimizes message delay for interactive, real-time sessions
- TCP is typically used for streaming
  - Low delay is not essential; startup
  - Loss recovery simplifies presentation
  - HTTP/TCP passes through firewalls