A Quantitative View: Delay, Throughput, Loss

Antonio Carzaniga

Faculty of Informatics
University of Lugano

September 26, 2018

Outline

- Quantitative analysis of data transfer concepts for network applications
- Propagation delay and transmission rate
- Multi-hop scenario

- How do we measure the “speed” and “capacity” of a network connection?

Quantifying Data Transfer

- How do we measure the “speed” and “capacity” of a network connection?
- Intuition
  - water moving in a pipeline
  - cars moving along a road

Quantifying Data Transfer

- How do we measure the “speed” and “capacity” of a network connection?

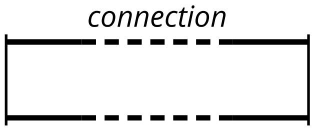
- Intuition
  - water moving in a pipeline
  - cars moving along a road

- ***Delay*** or ***Latency***
  - the time it takes for *one bit* to go through the connection (from one end to the other)

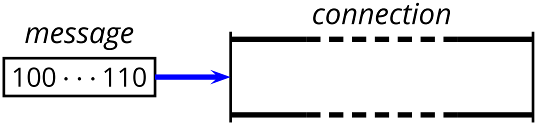
Quantifying Data Transfer

- How do we measure the "speed" and "capacity" of a network connection?
- Intuition
  - water moving in a pipeline
  - cars moving along a road
- ***Delay*** or ***Latency***
  - the time it takes for *one bit* to go through the connection (from one end to the other)
- ***Transmission rate*** or ***Throughput***
  - the amount of information that can get into (or out of) the connection in a time unit

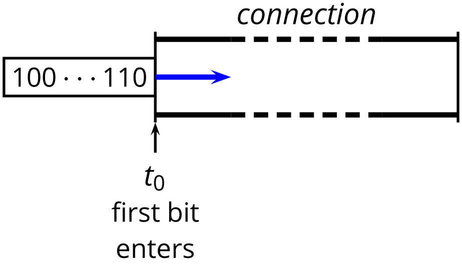
Delay (Latency) and Rate (Throughput)



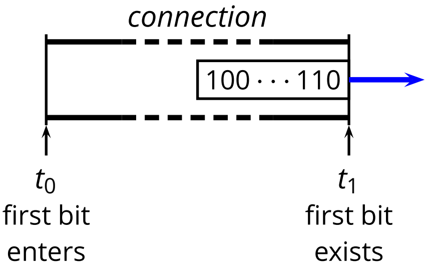
Delay (Latency) and Rate (Throughput)



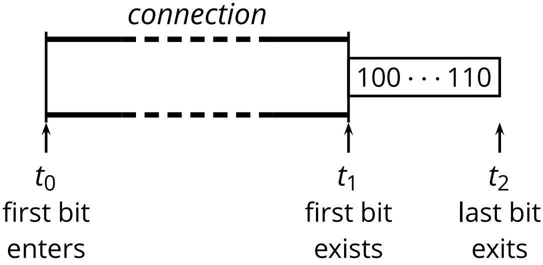
Delay (Latency) and Rate (Throughput)



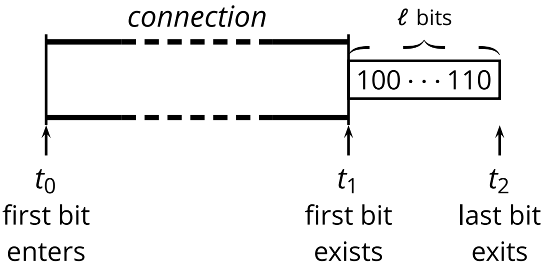
Delay (Latency) and Rate (Throughput)



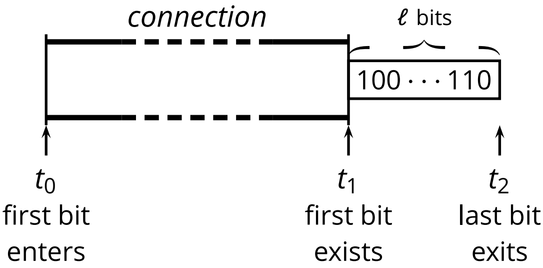
Delay (Latency) and Rate (Throughput)



Delay (Latency) and Rate (Throughput)

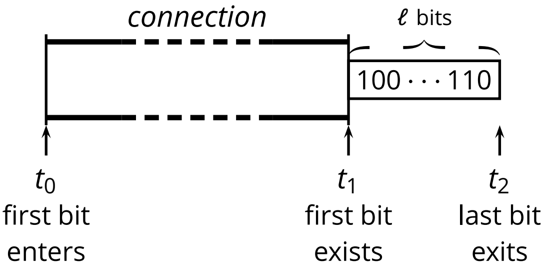


Delay (Latency) and Rate (Throughput)



Propagation ***Delay*** $d_{prop} = t_1 - t_0$ sec

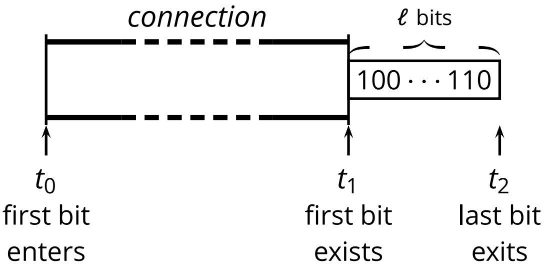
Delay (Latency) and Rate (Throughput)



Propagation ***Delay*** $d_{prop} = t_1 - t_0$ sec

Transmission ***Rate*** $R = \dfrac{\ell}{t_2 - t_1}$ bits/sec

Delay (Latency) and Rate (Throughput)



Propagation ***Delay***	$d_{prop} = t_1 - t_0$	sec
Transmission ***Rate***	$R = \dfrac{\ell}{t_2 - t_1}$	bits/sec
Total transfer time	$d_{end\text{-}end} = d + \dfrac{\ell}{R}$	sec

Examples

- How long does it take to tranfer a file between, say, Lugano and Zürich?

Examples

- How long does it take to tranfer a file between, say, Lugano and Zürich?

- How big is this file? And *how fast* is our connection?

Examples

- How long does it take to tranfer a file between, say, Lugano and Zürich?
- How big is this file? And *how fast* is our connection?

  E.g., a (short) e-mail message

$$\begin{aligned} \ell &= 4\text{Kb} \\ d_{prop} &= 500\text{ms} \\ R &= 1\text{Mb/s} \\ d_{end\text{-}end} &= ? \end{aligned}$$

- How long does it take to tranfer a file between, say, Lugano and Zürich?

- How big is this file? And *how fast* is our connection?

  E.g., a (short) e-mail message

$$
\begin{aligned}
\ell &= 4\text{Kb} \\
d_{prop} &= 500\text{ms} \\
R &= 1\text{Mb/s} \\
d_{end\text{-}end} &= 500\text{ms} + 4\text{ms} = 504\text{ms}
\end{aligned}
$$

Examples

- How about a big file? (E.g., a music collection of 50 songs)

- How about a big file? (E.g., a music collection of 50 songs)

$$
\begin{aligned}
\ell &= 400Mb \\
d_{prop} &= 500ms \\
R &= 1Mb/s \\
d_{end\text{-}end} &= ?
\end{aligned}
$$

- How about a big file? (E.g., a music collection of 50 songs)

$$
\begin{aligned}
\ell &= 400Mb \\
d_{prop} &= 500ms \\
R &= 1Mb/s \\
d_{end\text{-}end} &= 500ms + 400s = 400.5s = 6'40''
\end{aligned}
$$

- How about a big file? (E.g., a music collection of 50 songs)

$$
\begin{aligned}
\ell &= 400Mb \\
d_{prop} &= 500ms \\
R &= 1Mb/s \\
d_{end\text{-}end} &= 500ms + 400s = 400.5s = 6'40''
\end{aligned}
$$

- How about a bigger file? (E.g., a 32 Gb SSD)

Examples

- How about a big file? (E.g., a music collection of 50 songs)

$$
\begin{aligned}
\ell &= 400Mb \\
d_{prop} &= 500ms \\
R &= 1Mb/s \\
d_{end\text{-}end} &= 500ms + 400s = 400.5s = 6'40''
\end{aligned}
$$

- How about a bigger file? (E.g., a 32 Gb SSD)

$$
\begin{aligned}
\ell &= 32Gb \\
d_{prop} &= 500ms \\
R &= 1Mb/s \\
d_{end\text{-}end} &= ?
\end{aligned}
$$

- How about a big file? (E.g., a music collection of 50 songs)

$$
\begin{aligned}
\ell &= 400Mb \\
d_{prop} &= 500ms \\
R &= 1Mb/s \\
d_{end\text{-}end} &= 500ms + 400s = 400.5s = 6'40''
\end{aligned}
$$

- How about a bigger file? (E.g., a 32 Gb SSD)

$$
\begin{aligned}
\ell &= 32Gb \\
d_{prop} &= 500ms \\
R &= 1Mb/s \\
d_{end\text{-}end} &= \epsilon + 32000s = 8h\ 53'20''
\end{aligned}
$$

- How about going to Zürich on a Vespa?

Examples

- How about going to Zürich on a Vespa?
  - you carry 500, 32-Gb memory cards in your backpack
  - four seconds to take the cards out of your backpack

- How about going to Zürich on a Vespa?
  - you carry 500, 32-Gb memory cards in your backpack
  - four seconds to take the cards out of your backpack

$$\begin{aligned} \ell &= 32Gb \\ d_{prop} &= ? \\ R &= \\ d_{end\text{-}end} &= \end{aligned}$$

- How about going to Zürich on a Vespa?
  - you carry 500, 32-Gb memory cards in your backpack
  - four seconds to take the cards out of your backpack

$$
\begin{aligned}
\ell &= 32Gb \\
d_{prop} &= 6h \\
R &= ? \\
d_{end\text{-}end} &=
\end{aligned}
$$

- How about going to Zürich on a Vespa?
  - you carry 500, 32-Gb memory cards in your backpack
  - four seconds to take the cards out of your backpack

$$\begin{aligned} \ell &= 32Gb \\ d_{prop} &= 6h \\ R &= 4Tb/s \\ d_{end\text{-}end} &= ? \end{aligned}$$

- How about going to Zürich on a Vespa?
  - you carry 500, 32-Gb memory cards in your backpack
  - four seconds to take the cards out of your backpack

$$
\begin{aligned}
\ell &= 32Gb \\
d_{prop} &= 6h \\
R &= 4Tb/s \\
d_{\textit{end-end}} &= 6h
\end{aligned}
$$

- How about going to Zürich on a Vespa?
  - you carry 500, 32-Gb memory cards in your backpack
  - four seconds to take the cards out of your backpack

$$
\begin{aligned}
\ell &= 32Gb \\
d_{prop} &= 6h \\
R &= 4Tb/s \\
d_{\text{end-end}} &= 6h
\end{aligned}
$$

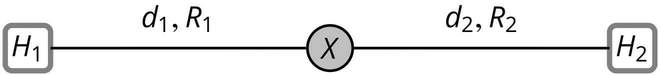
If you need to transfer a couple of SSD cards from Lugano to Zürich, and time is crucial. . . then you might be better off riding your Vespa to Zürich rather than using the Internet.

For more than 5 cards, you might also prefer the Post office!

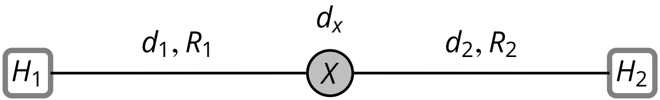
Store-And-Forward Delay

Store-And-Forward Delay

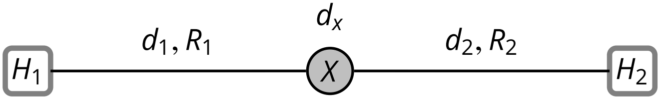
Store-And-Forward Delay



Store-And-Forward Delay

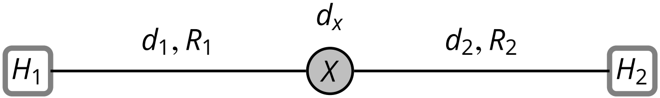


Store-And-Forward Delay



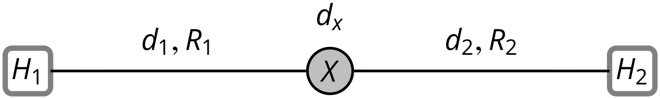
$$d_{end\text{-}end} = d_1 + \frac{\ell}{R_1}$$

Store-And-Forward Delay



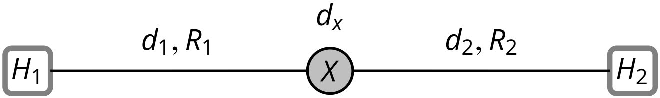
$$d_{end\text{-}end} = d_1 + \frac{\ell}{R_1} + d_x$$

Store-And-Forward Delay



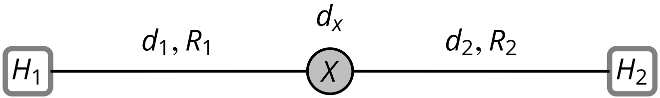
$$d_{\textit{end-end}} = d_1 + \frac{\ell}{R_1} + d_x + \frac{\ell}{R_2}$$

Store-And-Forward Delay

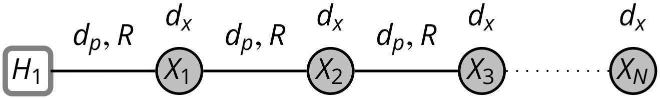


$$d_{\text{end-end}} = d_1 + \frac{\ell}{R_1} + d_x + \frac{\ell}{R_2} + d_2$$

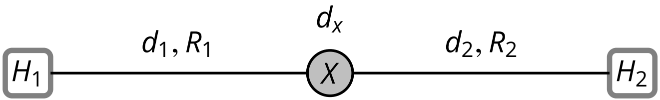
Store-And-Forward Delay



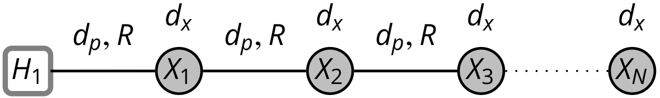
$$d_{end\text{-}end} = d_1 + \frac{\ell}{R_1} + d_x + \frac{\ell}{R_2} + d_2$$



Store-And-Forward Delay



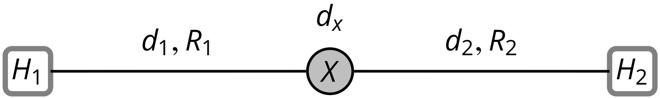
$$d_{end\text{-}end} = d_1 + \frac{\ell}{R_1} + d_x + \frac{\ell}{R_2} + d_2$$



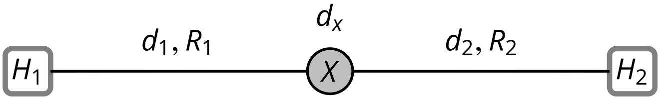
$$d_{end\text{-}end} = N\left(d_p + \frac{\ell}{R} + d_x\right)$$

End-toEnd Throughput

End-toEnd Throughput

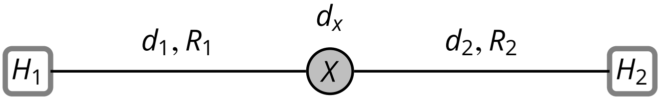


End-toEnd Throughput



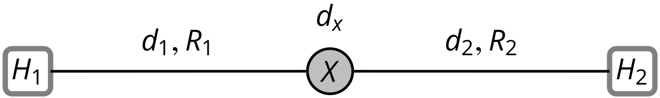
$$R_{end\text{-}end} =$$

End-toEnd Throughput

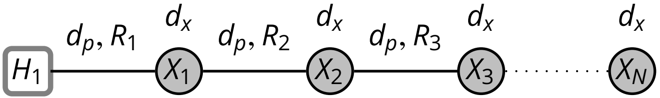


$$R_{end\text{-}end} = \min\{R_1, R_2\}$$

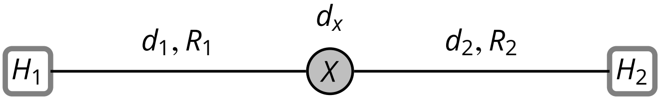
End-toEnd Throughput



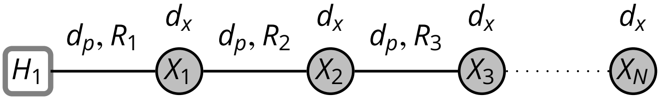
$$R_{end\text{-}end} = \min\{R_1, R_2\}$$



End-toEnd Throughput



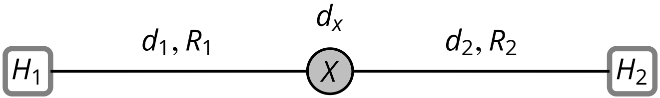
$$R_{end\text{-}end} = \min\{R_1, R_2\}$$



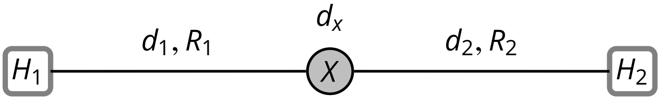
$$R_{end\text{-}end} = \min\{R_1, R_2, \ldots, R_N\}$$

Processing and Queuing Delays

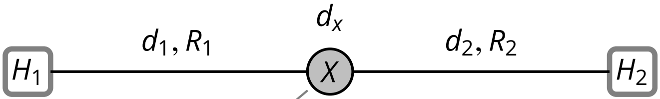
Processing and Queuing Delays



Processing and Queuing Delays



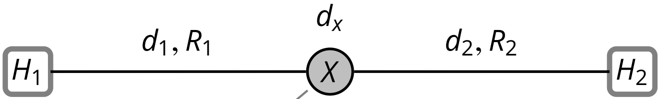
Processing and Queuing Delays



$$d_x = d_{cpu} + d_{queue}$$

$$d_{queue} = |q|/R_x$$

Processing and Queuing Delays



$$d_x = d_{cpu} + d_{queue}$$

$$d_{queue} = |q| / R_x$$

queue length

output rate

$$d_x = d_{cpu} + d_{queue}$$

where

$$d_{queue} = \underbrace{|q|}_{\text{queue length}} / \underbrace{R_x}_{\text{output rate}}$$

$$d_x = d_{cpu} + d_{queue}$$

where

$$d_{queue} = |q|/R_x$$

queue length

output rate

$\ldots R_x$ is also the rate at which packets get out of the queue

Queuing Delay

- ***Ideal case:*** constant input data rate

$$\lambda_{in} < R_x$$

In this case the $d_{queue} = 0$, because $|q| = 0$

- ***Ideal case:*** constant input data rate

$$\lambda_{in} < R_x$$

In this case the $d_{queue} = 0$, because $|q| = 0$

- ***Extreme case:*** constant input data rate

$$\lambda_{in} > R_x$$

In this case $|q| = (\lambda_{in} - R_x)t$ and therefore

$$d_{queue} = \frac{\lambda_{in} - R_x}{R_x} t$$

Queuing Delay

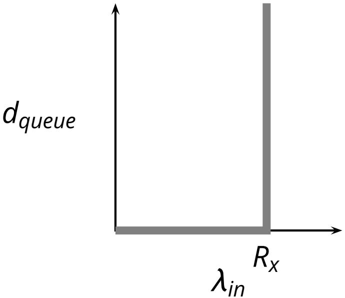
Queuing Delay

- Steady-state queuing delay

$$d_{queue} = \begin{cases} 0 & \lambda_{in} < R_x \\ \frac{\lambda_{in} - R_x}{R_x} t & \lambda_{in} > R_x \end{cases}$$

- Steady-state queuing delay

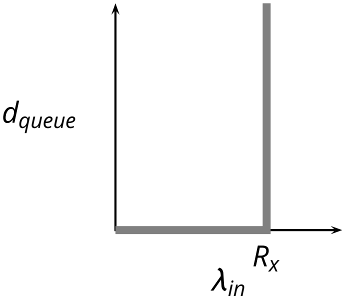
$$d_{queue} = \begin{cases} 0 & \lambda_{in} < R_x \\ \frac{\lambda_{in} - R_x}{R_x} t & \lambda_{in} > R_x \end{cases}$$



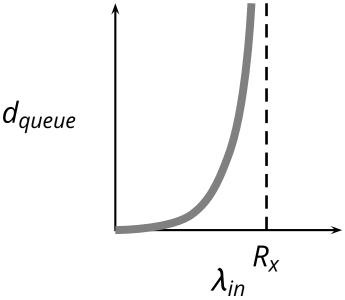
ideal input flow
λ_{in} constant

- Steady-state queuing delay

$$d_{queue} = \begin{cases} 0 & \lambda_{in} < R_x \\ \frac{\lambda_{in} - R_x}{R_x} t & \lambda_{in} > R_x \end{cases}$$



ideal input flow
λ_{in} constant



realistic input flow
λ_{in} variable

Queuing Delay

- *Conclusion:* as the input rate λ_{in} approaches the maximum throughput R_x, packets will experience very long delays

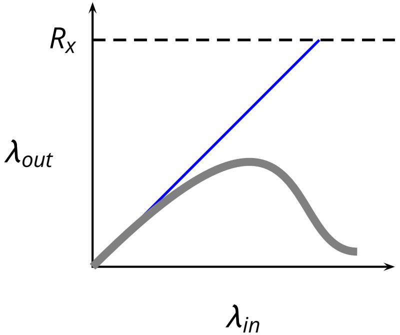
- *Conclusion:* as the input rate λ_{in} approaches the maximum throughput R_x, packets will experience very long delays
- More realistic assumptions and models
  - finite queue length (buffers) in routers

  $\Rightarrow$ ***packets are dropped***

Queuing Delay

- *Conclusion:* as the input rate λ_{in} approaches the maximum throughput R_x, packets will experience very long delays
- More realistic assumptions and models
  - finite queue length (buffers) in routers

  $\Rightarrow$ ***packets are dropped***



- *Conclusion:* as the input rate λ_{in} approaches the maximum throughput R_x, packets will experience very long delays
- More realistic assumptions and models
  - finite queue length (buffers) in routers

  $\Rightarrow$ ***packets are dropped***