Chapter 1: roadmap
- What is the Internet?
- Protocols
- Network Structure
- Network Types
- Network topologies
- Network edge
- Network core, transmission types
- Network access and physical media
- Internet structure and ISPs
- Delay & loss in packet-switched networks
- Protocol layers, service models
- History

How do loss and delay occur?
- Packets queue in router buffers
  - Packet arrival rate to link exceeds output link capacity
  - Packets queue, wait for turn

Packet being transmitted (delay)

Packets queueing (delay)

Free (available) buffers: arriving packets dropped (lost) if no free buffers
Four sources of packet delay

1. Processing:
   - Check bit errors
   - Determine output link

2. Queuing:
   - Time waiting at output link for transmission
   - Depends on congestion level of router

Delay in packet-switched networks

3. Transmission delay:
   - \( R = \text{link bandwidth (bps)} \)
   - \( L = \text{packet length (bits)} \)
   - Time to send bits into link = \( \frac{L}{R} \)

4. Propagation delay:
   - \( d = \text{length of physical link} \)
   - \( s = \text{propagation speed in medium (~2x10^8 \text{ m/sec})} \)
   - Propagation delay = \( \frac{d}{s} \)

Note: \( s \) and \( R \) are very different quantities!

Caravan analogy

- Cars "propagate" at 100 km/hr
- Toll booth takes 12 sec to service a car (transmission time)
- Car-bit; caravan ~ packet
- Q: How long until caravan is lined up before 2nd toll booth?

- Time to "push" entire caravan through toll booth onto highway = \( 12 \times 10 = 120 \text{ sec} \)
- Time for last car to propagate from 1st to 2nd toll booth:
  - \( \frac{100 \text{km}}{100 \text{km/hr}} = 1 \text{ hr} \) (prop. delay)
  - A: 62 minutes
Caravan analogy (more)

- Cars now "propagate" at 1000 km/hr
- Toll booth now takes 1 min to service a car
- Q: Will cars arrive to 2nd booth before all cars serviced at 1st booth?
  - Yes! After 7 min, 1st car at 2nd booth and 3 cars still at 1st booth.
  - 1st bit of packet can arrive at 2nd router before packet is fully transmitted at 1st router.

Nodal delay

\[ d_{\text{nodal}} = d_{\text{proc}} + d_{\text{queue}} + d_{\text{trans}} + d_{\text{prop}} \]

- \( d_{\text{proc}} \) = processing delay
  - typically a few microsecs or less
- \( d_{\text{queue}} \) = queuing delay
  - depends on congestion (varies from packet to packet)
- \( d_{\text{trans}} \) = transmission delay
  - \( = L/R \) significant for low-speed links
- \( d_{\text{prop}} \) = propagation delay
  - a few microsecs to hundreds of msecs

Queueing delay (revisited)

- \( R \) = link bandwidth (bps)
- \( L \) = packet length (bits)
- \( a \) = average packet arrival rate (packets/sec)

\[ \text{traffic intensity} = \frac{La}{R} \]

- \( La/R \approx 0 \): average queueing delay small
- \( La/R \rightarrow 1 \): delays become large
- \( La/R > 1 \): more "work" arriving than can be serviced, average delay infinite.
"Real" Internet delays and routes

- What do "real" Internet delay & loss look like?
- **Traceroute** (traceroute.org) program: provides delay measurement from source to router along end-end Internet path towards destination. For all:
  - sends three packets that will reach router / on path towards destination
  - router will return packets to sender
  - sender times interval between transmission and reply.

![Traceroute Diagram]

---

Packet loss

- queue (aka buffer) preceding a link has finite capacity (buffer capacity)
- when packet arrives to full queue, packet is dropped (aka lost)
- lost packet may be retransmitted by previous node, by source end system, or not retransmitted at all
Chapter 1: roadmap
- What is the Internet?
- Protocols
- Network Structure
- Network Types
- Network topologies
- Network edge
- Network core, transmission types
- Network access and physical media
- Internet structure and ISPs
- Delay & loss in packet-switched networks
- Protocol layers, service models
- History

Protocol "Layers"

Networks are complex!
- many "pieces":
  - hosts
  - routers
  - links of various media
  - applications
  - protocols
  - hardware, software

Question: Is there any hope of organizing structure of network?
Or at least our discussion of networks?

Organization of air travel
- a series of steps
- ticket (purchase)    ticket (complain)
- baggage (check)     baggage (claim)
- gates (load)        gates (unload)
- runway takeoff      runway landing
- airplane routing    airplane routing

WMU-CS, Dr. Gupta
**Organization of air travel: a different view**

<table>
<thead>
<tr>
<th>Layer</th>
<th>Service</th>
</tr>
</thead>
<tbody>
<tr>
<td>ticket</td>
<td>(purchase)</td>
</tr>
<tr>
<td>baggage</td>
<td>(check)</td>
</tr>
<tr>
<td>gates</td>
<td>(load)</td>
</tr>
<tr>
<td>runway</td>
<td>takeoff</td>
</tr>
<tr>
<td></td>
<td>airplane routing</td>
</tr>
</tbody>
</table>

**Layers:** each layer implements a service
- via its own internal-layer actions
- relying on services provided by layer below

**Layered air travel: services**

- Counter-to-counter delivery of person+bags
- Baggage-check-to-baggage-claim delivery
- People transfer: loading gate to arrival gate
- Runway-to-runway delivery of plane
- Airplane routing from source to destination

**Distributed implementation of layer functionality**

<table>
<thead>
<tr>
<th>Departing airport</th>
<th>Arriving airline</th>
</tr>
</thead>
<tbody>
<tr>
<td>ticket (purchase)</td>
<td>ticket (complain)</td>
</tr>
<tr>
<td>baggage (check)</td>
<td>baggage (claim)</td>
</tr>
<tr>
<td>gates (load)</td>
<td>gates (unload)</td>
</tr>
<tr>
<td>runway takeoff</td>
<td>runway landing</td>
</tr>
<tr>
<td>airplane routing</td>
<td>airplane routing</td>
</tr>
</tbody>
</table>

**Intermediate air traffic sites**

- Airplane routing
- Airplane routing
Why layering?

Dealing with complex systems:
- explicit structure allows identification, relationship of complex system’s pieces
- layered reference model for discussion
- modularization eases maintenance, updating of system
  - change of implementation of layer’s service transparent to rest of system
  - e.g., change in gate procedure doesn’t affect rest of system
- layering considered harmful?

Internet protocol stack

- **application**: supporting network applications
  - FTP, SMTP, STTP
- **transport**: host-host data transfer
  - TCP, UDP
- **network**: routing of datagrams from source to destination
  - IP, routing protocols
- **link**: data transfer between neighboring network elements
  - PPP, Ethernet
- **physical**: bits “on the wire”

Layering: logical communication

Each layer:
- distributed
- “entities” implement layer functions at each node
- entities perform actions, exchange messages with peers
Layering: logical communication

- Example: transport
  - Takes data from application
  - Adds addressing, reliability check info to form "datagram"
  - Sends datagram to peer
  - Waits for peer to acknowledge receipt
  - Analogy: postal office

Layering: physical communication

Protocol layering and data

Each layer takes data from above
- Adds header information to create new data unit
- Passes new data unit to layer below
Chapter 1: roadmap

- What is the Internet?
- Protocols
- Network Structure
- Network Types
- Network topologies
- Network edge
- Network core, transmission types
- Network access and physical media
- Internet structure and ISPs
- Delay & loss in packet-switched networks
- Protocol layers, service models
- History

Internet History

1961-1972: Early packet-switching principles

- 1961: Kleinrock - queueing theory shows effectiveness of packet-switching
- 1964: Baran - packet-switching in military nets
- 1967: ARPAnet conceived by Advanced Research Projects Agency
- 1969: first ARPAnet node operational
- 1972:
  - ARPAnet demonstrated publicly
  - NCP (Network Control Protocol) first host-host protocol
  - First e-mail program
  - ARPAnet has 15 nodes

1972-1980: Internetworking, new and proprietary nets

- 1970: ALOHAnet satellite network in Hawaii
- 1973: Metcalfe's PhD thesis proposes Ethernet
- 1974: Cerf and Kahn - architecture for interconnecting networks
- late 70's: proprietary architectures: DECnet, SNA, XNS
- late 70's: switching fixed length packets (ATM precursor)
- 1979: ARPAnet has 200 nodes

Cerf and Kahn’s internetworking principles:
- minimalism, autonomy - no internal changes required to interconnect networks
- best effort service model
- stateless routers
- decentralized control
  define today’s Internet architecture
Introduction

Internet History

1980-1990: new protocols, a proliferation of networks
- 1983: deployment of TCP/IP
- 1982: SMTP e-mail protocol defined
- 1983: DNS defined for name-to-IP-address translation
- 1985: FTP protocol defined
- 1988: TCP congestion control
- New national networks: CSnet, BITnet, NSFnet, Minitel
- 100,000 hosts connected to confederation of networks

1980-1990: new protocols, a proliferation of networks
- Early 1990’s: ARPAnet decommissioned
- Early 1990’s: Web
  - hypertext (Bush 1945, Nelson 1960’s)
  - HTML, HTTP: Berners-Lee
  - 1994: Mosaic, later Netscape
- Late 1990’s - 2000’s:
  - more killer apps: instant messaging, peer2peer file sharing (e.g., Napster)
  - network security to forefront
  - est. 50 million host, 100 million+ users
  - backbone links running at Gbps

Late 1990’s - 2000’s:
- more killer apps: instant messaging, peer2peer file sharing (e.g., Napster)
- network security to forefront
- est. 50 million host, 100 million+ users
- backbone links running at Gbps

Introduction: Summary

Covered a “ton” of material:
- Internet overview
- what’s a protocol?
- network edge, core, access network
  - packet-switching versus circuit-switching
- Internet/ISP structure
- performance: loss, delay
- layering and service models
- history

You now have:
- context, overview, “feel” of networking
- more depth, detail to follow!
Chapter 11

Network Modeling & Simulation

Source: Guizani Book, OPNET docs

Agenda

- MODELING CONCEPTS
- MODELING METHODOLOGY: HOW TO GET STARTED?
- OPNET MODELING & SIMULATION PLATFORM
- OPNET examples

Learning Objectives

- The importance of modeling and simulation in characterizing systems behavior
- Practical limitations and issues associated with modeling complex systems
- The different steps of a successful modeling methodology
- The OPNET modeling and simulation platform and software modules
- Practical use cases of the OPNET software product
What's a Simulation?

- "Simulation" refers to the process of replicating the behavior of a real-life system using a systematic methodology that is both accurate and repeatable.

- Simulation is widely recognized by technical and management circles that live testing cannot reasonably perform all possible configurations required by a detailed analysis study due to several reasons:
  - Cost
  - Flexibility
  - New Functionality

Simulation & Real Life Relevance

- Modeling network systems accurately and efficiently is a difficult task.

- Generally, there is a trade to be made between how accurate the simulation should versus how quickly is the answer required.

- Some of the issues the developers have to deal with in creating these simulation models include:
  - Memory Limitations
  - Processor Limitations
  - Simulation Platform Limitations

Modeling Methodology

- To tackle any modeling project, the user should follow a systematic methodology to arrive at the answers required by the analysis study.

- This systematic methodology has a number of distinct characteristics, including:
  - Tested
  - Repeatable
  - Methodical
  - Achievable

- There is no single way to define a modeling methodology, but the following describe it in the most general form so that it can be usable in as many possible cases as possible.
Step 1: Define the Problem

- This is the most important step in any modeling project.
- Nevertheless, the analysts in the fury of schedule limitations and budget constraints commonly skip it.
- It is wise to spend as much time as needed to arrive at a very clear problem statement that is as complete as possible from both qualitative and quantitative standpoints.

Step 1: Define the Problem

- An example of a good question is: How does HTTP/1.1 perform over a geostationary satellite network when compared to HTTP/1.0 with connection reuse?
- An example of a bad question addressing similar concerns is: Is HTTP/1.1 better or worse than HTTP/1.0?

Step 2: Define/Build a Network Topology

- Generally, there are three different topology types that can be built: complete, partial, and single path.
- A **complete** topology includes all the links and devices in the network.
- A **partial** topology models a section of the network in detail (e.g., a data-center subnet).
- A **single-path** topology contains only the infrastructure portion supporting the traffic flow in question among the network devices.
- A complete topology is necessary (in most cases) when a particular problem is spread across the network and it is important to identify the impact of the problem on all devices involved.
Step 3: Define/Configure Traffic

- There are two techniques for representing traffic in OPNET: explicit and background traffic.
- The architecture of background traffic in OPNET is designed so that background traffic influences links, devices, and explicit packet flows along its path.
- Background traffic can be represented as a flow between a source and destination or as static utilization on a link or device.
- For example, if we are focusing on the impact of server traffic on the network and not studying the server itself in detail, this methodology can use the background traffic model without too much loss of accuracy or model flexibility.

Step 4: Choose Statistics

- The user has to select the statistics that are relevant to the problem statement defined in the first step of this modeling process.
- For example, when studying failure/recovery scenarios, link and circuit utilization levels are prime candidates for statistical monitoring and analysis.
- OPNET uses the concept of "buckets" (or bins) to collect statistics during the simulation.
- Using buckets, the simulation kernel groups a number of data points into a bucket and presents a single value to represent those data points.
- Occasionally, the collected statistics may not be granular enough forcing the user to change the collection mode.
**Bucket Size Effects**

100 buckets vs. 3600 buckets

---

**Step 5: Perform What-if’s**

- The initial set of simulation runs is necessary to establish a baseline of the model and its statistics. Once the initial runs are completed, further studies can be carried out.

- For example, when conducting a routing analysis study, the user may perform other what-if scenarios on the routing protocols such as: selecting a different routing protocol, changing the attributes of an existing routing protocol, using routing to achieve load balancing etc.

---

**Step 5: Perform What-if’s (cont.)**

- There are a number of analysis techniques that can be used to conduct the what-if scenarios. These include:
  - Discrete event simulation (DES)
  - Analytical prediction
  - Hybrid simulation
  - Parallel simulation
  - Co-simulation
Step 6: Present Results & Provide Documentation

- The model developer (or any member of the modeling team) should take the responsibility of documenting the results and providing the appropriate medium to share the results with other team members.

- The level of details of such a presentation depends on the target audience.

- Usually, management is interested in high level results, and the way they can impact the company’s financial bottom-line.

- On the other hand, the technical group is usually interested in the intimate details of the model to ensure its accuracy and verify its strength.

Outline of a Technical Report

- Introduction
- OPNET Modeling & Simulation Platform
- OPNET has several products that serve the needs of individual audiences: enterprises, network R&D organizations, and network service providers.
  - IT Guru: used by enterprises
  - SP Guru: used by service providers
  - Modeler: used for R&D and academic purposes
  - VNE Server: data collection and analysis
  - WDM Guru: focuses on optical networking technologies such as WDM and SONET.