ETSN10: Network Architecture and Performance — Reference Sheet
Lund University

Karl Hallsby

Last Edited: March 18, 2020

Contents

List of Theorems

1 Networking Review

This course assumes that you already know about basic networking terms and models. However, this section is meant as a refresher for people who have already taken that particular set of courses, or as a quick introduction for those who have not taken those courses yet.

1.1 Networking Stack

Defn 1 (Networking Stack)**.** The *networking stack* is a set of layers that are built on top of another that are used to implement inter-computer communication. There are 5 layers:

1. Physical Layer
2. Data-Link Layer
3. Network Layer
4. Transport Layer
5. Application Layer

Each of these fulfill different roles, allowing for different implementations to be used interchangeably. Additionally, not all points in the network graph require the entire networking stack. For example, a router or switch does not require the Transport Layer or the Application Layer, since the job of this hardware is to just move packets around.

Each layer of the networking stack add a header and possibly a footer. Information in headers and footers includes source and destination addresses, checksums, packet size, and protocol identifiers.

Defn 2 (Encapsulation)**.** The process of adding headers and footers is called *encapsulation*.

Defn 3 (Decapsulation)**.** The process of removing the headers and footers of a Packet at the other end is called *decapsulation*.

1.2 Physical Layer

Defn 4 (Physical Layer)**.** The *physical layer* in the Networking Stack is the lowest layer in the stack. It consists of the physical connection that is made between computers and the modulation and coding to represent data as a signal on that connection. For obvious reasons, all devices **must** implement this layer.

Some examples of this layer are:

- Copper twisted-pair cables
- Fiber optic wires
- Radio transmission

Remark 4.1. The design of different Physical Layers is more a hardware design question than that of a software implementation issue. So, it is not discussed in this course, beyond the use of cellular networks.

1.3 Data-Link Layer

Defn 5 (Data-Link Layer)**.** The *data-link layer* is the next layer in the Networking Stack. All devices must implement this layer. The data-link layer is responsible for getting data between two devices that have a Physical Layer connection between them. It is also known as the *Medium Access Control (MAC) layer*. It controls when each device should use the link, and may also include error correction and retransmission.

Some examples of these are:

- Ethernet
- Point-to-Point Protocol (PPP)
- 802.11 (WiFi)
- Bluetooth

1.4 Network Layer

Defn 6 (Network Layer)**.** The *network layer* is the third layer in the Networking Stack. This is the last layer that every device must have. It is responsible for end-to-end data delivery between two devices (hosts) anywhere on the network. It is responsible for routing and forwarding.

Some examples of this layer are:

- Internet Protocol (IP)
- Internet Control Message Protocol (ICMP)

- Routing Protocols

Defn 7 (Routing)**.** *Routing* is the process of determining and selecting the best end-to-end path through a network.

There are algorithms designed to discover the other participants in the network by analyzing the grpah that is created when each participant is considered a node in this graph.

Defn 8 (Forwarding)**.** *Forwarding* is the process of selecting the next hop for the given packet.

Defn 9 (Circuit Switching)**.** In *circuit switching*, dedicated resources are allocated end-to-end for a particular flow of data. No other flows may use those resources while the connection is maintained. Meaning, that during a connection, there is a single dedicated circuit between the host and the receiver.

Defn 10 (Packet Switching)**.** In *packet switching*, no resources are allocated to a flow, and each flow's data is broken up into discrete packets. Each Packet is routed and forwarded independently.

Defn 11 (Packet)**.** A connection from one device to another is a stream. However, to effectively implement many things in the Networking Stack, the stream must be broken into several discrete *packets*.

1.4.1 Discovering Routing Information

1.4.1.1 Dijkstra's Algorithm

Dijkstra's Algorithm starts at the source and builds up the shortest path step-by-step.

There are 2 main problems with Dijkstra's Algorithm:

1. The weights along the edges of the graph cannot be negative.
    - If this were to occur, then we would have an infinite cycle going through the negative edge.
2. We **must** know the entire network's topology before beginning the algorithm.
    - Typically, this is not possible on vast networks, like the Internet.

Algorithm 1.1: Dijkstra's Algorithm

```
Input  : A graph with weights on each edge representing "distance" between the nodes.
Output: The path through the graph with the least weight from some starting node.
1  Assign tentative distance estimate to each node: 0 for the initial node, $\infty$ for all other nodes.
2  Mark all nodes except the initial node as unvisited.
3  Set the initial node as current node.
4  for The current node's unvisited neighbours do
5      Calculate their distance.
6      Compare this new distance against the current distance.
7      Take whichever distance is smaller and use that as the new value.
8      Mark the current node as visited.
9      if Reached target node OR No more unvisited nodes with distance $< \infty$ then
10         Stop and terminate the algorithm.
11     Set the unvisited node with the smallest distance as the new current node.
```

1.4.1.2 Bellman-Ford Algorithm

The Bellman-Ford algorithm works with graphs that have negative weight values on edges. It also **does not** require us to know the entire network's topology before beginning. However, it is much slower than

Dijkstra's Algorithm.

Algorithm 1.2: Bellman-Ford Algorithm

```
Input  : A graph with weights on each edge representing "distance" between the nodes.
Output: The path through the graph with the least weight from some starting node.
1  Set the distance for the source node to 0 and for all other nodes to $\infty$.
2  Set the predecessor of all nodes to null.
3  for $n \in N$ where $N$ is the number of nodes AND $n \leq N - 1$ do
4      We repeat $N - 1$ times, because $N - 1$ is the maximum length of a non-cyclic path.
5      for Each edge $(u, v)$ do
6          if distance to $u$, plus the edge weight $<$ current distance to $v$ then
7              We have discovered a shorter path to $v$.
8              Set the distance to $v$ to this new value and the predecessor of $v$ to $u$.
9  for each edge $(u, v)$ do
10     if distance to $u$ plus the edge weight $<$ the distance to $v$ then
11         The graph contains a negative-weight cycle.
12         Terminate.
13     We have found the shortest paths to each node from the source.
14     Terminate.
```

1.4.2 Routing Information

Defn 12 (Routing Protocol)**.** A *routing protocol* is used to construct a Routing Table in each node.

Defn 13 (Routing Table)**.** The *routing table* is a table that contains entries on the shortest distances through the current network graph. This is used to help determine where to next forward packets onto.

Each node has a routing table that contains the cost to the destination and the next hop for each destination. For a valid routing table, we need to have:

1. Which destination the packets are going to?
2. What is the next hop after the destination?
3. The "metric" or weight of that particular path through the network graph?
    - We need this term so we can update the routing table as we discover new routes.
    - This way if a new route with a lower cost presents itself, we can replace the higher-cost one with the lower one.

There are 2 ways to collect the routing information necessary to properly route packets through a network. These 2 routing tables are:

1. Static Routing Tables
2. Dynamic Routing Tables

1.4.2.1 Static Routing Tables

Defn 14 (Static Routing Table)**.** *Static routing table*s are manually configured by a user or a program before the system begins routing the information.

This means that there is no overhead to attempt to figure out the network's topology. The information needed has already been given to the device's Routing Table, so that step is completed.

1.4.2.2 Dynamic Routing Tables

Defn 15 (Dynamic Routing Table)**.** A *dynamic routing table* is a Routing Table that is built automatically by the device. To do this, a Routing Protocol is used to build the table while the device is running. This is done using either:

1. Distance Vector Routing Protocol
2. Link State Routing Protocol

Defn 16 (Distance Vector Routing Protocol)**.** The *distance vector routing algorithm* is a Routing Protocol that uses Bellman-Ford Algorithm to construct a Routing Table.

Each node only begins with knowledge of their immediate neighbours and the costs to reach them.

- Nodes then send this information (the routing table) to their neighbours.

- If a neighbour sends us a route that is shorter than one we already have, update our table to reflect this.
- After updating, send the new table to our neighbours.
- If a node goes down, discard any lines in the routing table that have it as the next hop and follow the above to find a new route.

Defn 17 (Link State Routing Protocol)**.** The *link state routing protocol* is a Routing Protocol that uses Dijkstra's Algorithm to construct a Routing Table. Thus, we need to know what the whole network looks like.

- Each node floods the network with the list of nodes it can connect to and the costs to them.
- Every node builds up a picture of the entire network, then can use Dijkstra's Algorithm to determine the shortest path to each destination.
- The Routing Table is then constructed based on the computed shortest paths.

However, the problem with both Distance Vector Routing Protocol and Link State Routing Protocol is that they do not scale well. However, to help with this, both protocols are contained within a single Autonomous System.

Defn 18 (Autonomous System)**.** An *autonomous system* is a smaller network, like a business or home, that performs a Routing Protocol upon itself. It is done this way because both the Distance Vector Routing Protocol and Link State Routing Protocol do not scale to Internet-sized networks well.

To combat this, each of the autonomous systems has a Speaker Node that speaks to the outside world. This speaker node is used in the Path Vector Routing Protocol.

Defn 19 (Speaker Node)**.** A *speaker node* is a specially designated node within a single Autonomous System that communicates **only with other speaker nodes**. Then, the Path Vector Routing Protocol is performed upon all the speaker nodes to construct a wider network graph consisting only of the Autonomous Systems that the speaker nodes belong to.

Defn 20 (Path Vector Routing Protocol)**.** Between Autonomous Systems, we use a variant of Distance Vector Routing Protocol called *path vector routing protocol*. Each Autonomous System has its own Speaker Node. Only the Speaker Nodes can communicate across the Autonomous System boundary, and exchange information about which destinations they can reach and the paths to them.

1.4.3 IP Addresses

Defn 21 (IP Address)**.** *IP Address*es are the unique end-point routing identifiers used on Packets to deliver their data. A natural analogy is that of apartment numbers on apartment buildings.

IPv4 uses 32 bits, split up into 4 8-bit chunks. Each chunk is read as its decimal equivalent, for humans. IPv6 uses 128 bits, where every 16 bits are interpreted as a hexadecimal number.

Defn 22 (Subnet)**.** All hosts that are in the subnet will have IP Addresses that match the number of bits that are present in the subnet. Keeping with the apartment analogy, the subnet is like the address of the apartment building.

For example, in `192.168.2.0/24`, the `24` means the first 24 bits of the subnet are 1's (`255.255.255.0`). So, the first 24 bits, or 3 IP Address blocks (`192.168.2`), will remain the same for every client in that subnet.

Defn 23 (Classless Inter-Domain Routing)**.** *Classless Inter-Domain Routing* or *CIDR* (pronounced "cider") is a hierarchical way to organize IP Addresses.

This allowed:

- Subnets to be of any length.
- Destinations in a router's routing and forwarding tables may be full IP Addresses or Subnets.
  - A destination IP Address will then be matched to the most specific destination in the table when making forwarding decisions.

1.5 Transport Layer

Defn 24 (Transport Layer)**.** The *transport layer* is built where the Network Layer has delivered all the data to the end host.

This means that only the source and destination hosts have this layer. So, routers and switches do not have this, but your phone, laptop, and the server you're connecting to do.

This layer may be responsible for:

- Connection-Oriented Communication
- Multiplexing Data Flows
- Reliable data delivery
- Data flow control
- Data congestion control

Some implementation of this layer do not handle all of these functions, but all **must** handle the multiplexing of different data flows.

There are 2 main transport layers in-use today:

1. Transmission Control Protocol
2. User Datagram Protocol

Defn 25 (Connection-Oriented Communication)**.** A *connection-oriented communication* system means that a connection must be established before any data is transferred. In addition, this connection must be maintained throughout the transmission of data.

Remark 25.1 (Connectionless Communication)*.* In a *connectionless communication* system, hosts can send data at any time without prior connection being made.

1.5.1 Transport Protocols

Defn 26 (User Datagram Protocol)**.** *User Datagraph Protocol (UDP)* is the simplest transport protocol available. It performs multiplexing and error checking, **but nothing else**.

When a host wants to send data to another host, it just sends it, making UDP a Connectionless Communication protocol. If the data gets lost, too bad.

Defn 27 (Transmission Control Protocol)**.** *Transmission Control Protocol (TCP)* is a Connection-Oriented Communication protocol. It performs:

- Multiplexing
- Reliable data delivery
- Error detection
- Ordered data delivery
- Flow control
- Congestion control

Transmission Control Protocol uses a 3-way hand shake to establish a connection.

Defn 28 (Three-Way Handshake)**.** Transmission Control Protocol uses a *Three-Way Handshake* to establish a connection.

The client sends a `SYN` message to the recipient. The receiver sends back a `SYN_ACK` message to acknowledge the receipt of the original `SYN` message. The client then sends another `ACK` to the receiver to acknowledge the receipt of the `SYN_ACK` message.

Because this handshake is asymmetrical, there is a difference between servers and clients. A server must have a Port open, and be listening for client connections.

Defn 29 (Four-Way Handshake)**.** Transmission Control Protocol uses a *four-way handshake* to close a connection. Each host can close its side of the connection independently.

1.5.2 Multiplexing Data Flows

Defn 30 (Port)**.** A *port* is a single instance of a data flow. There are many different flows of data. These may be from different applications or different instances of the same application. In both Transmission Control Protocol and User Datagram Protocol, flows are given unique *port numbers*.

Some of these are standard for particular applications, e.g. port 80 for HTTP (web), port 25 for SMTP (email). The transport protocol uses the port number to deliver data to the correct application.

1.5.3 Data Delivery

1.5.3.1 Data Delivery in Transmission Control Protocol

Defn 31 (Sequence Number)**.** Transmission Control Protocol uses *sequence numbers* to make sure data is complete and in order when it is delivered. It also includes error detection, and segments with errors are retransmitted.

1.5.4 Flow Control

Defn 32 (Flow Control)**.** *Flow control* refers to signalling between the sender and receiver to ensure the sender does not send data faster than the receiver can process.

A receiving host may have limited buffer space for incoming messages and takes time to process each message.

1.5.4.1 Flow Control in Transmission Control Protocol

Defn 33 (Sliding Window)**.** Flow Control in Transmission Control Protocol uses a *sliding window* mechanism.

1.5.5 Congestion Control

Defn 34 (Congestion Control)**.** *Congestion control* refers to mechanisms for detecting and reducing Congestion.

Defn 35 (Congestion)**.** *Congestion* in a network occurs when there is too much data being sent and the network is unable to deliver it all. This can result in data loss or long delays.

1.5.5.1 Congestion Control in Transmission Control Protocol

Defn 36 (Congestion Window)**.** In Transmission Control Protocol, lost data is considered a sign of Congestion and the sender should reduce its rate. The sender has a *congestion window*, which refers to the maximum number of unacknowledged segments that may be in transit at a time.

A Transmission Control Protocol connection begins in **slow start**, where the Congestion Window is doubled every round trip. When a threshold is reached, it changes to *congestion avoidance*, where the congestion window increases by 1 maximum segment size each time. When packet loss is detected, the congestion window is halved.

$$\text{BW}_{Max} = \frac{\text{MSS} \times \sqrt{\frac{3}{2}}}{\text{RTT} \times \sqrt{p}} \tag{1.1}$$

- BW_{Max}: Maximum bandwidth.
- MSS: Maximum Transmission Control Protocol Packet size.
- RTT: Round trip time.
- p: Packet loss probability.

1.6 Application Layer

Defn 37 (Application Layer)**.** The *application layer* is the last and highest layer in the Networking Stack. This layer is created by the actual applications that interface with the user, their email client, web browser, games, etc. This is the layer where data is actually generated and consumed by any application that communicates over the Internet.

Some examples of item in this layer are:

- Hypertext Transfer Protocol (HTTP)
- Simple Mail Transfer Protocol (SMTP)
- Extensible Messaging and Presence Protocol (XMPP)
- Skype

2 Probability Review

This section is meant to quick review and introduce the equations that will be used throughout this course. It is not meant to be comprehensive and/or in-depth. For more information about the topic of probability and statistics, refer to the Math 374 - Probability and Statistics document.

2.1 Axioms of Probability

Defn 38 (Sample Space)**.** The *sample space* is the set of all possible outcomes in a random experiment. It is denoted with the capital Greek omega.

$$\Omega \tag{2.1}$$

Defn 39 (Event)**.** An *event* is a subset of the Sample Space that we are interested in. These are generally denoted with capital letters.

$$A \subseteq \Omega \tag{2.2}$$

Defn 40 (Mutually Exclusive)**.** Any two Events are *mutually exclusive* if the equation below holds.

$$\text{P}(A \cup B) = \text{P}(A) + \text{P}(B) \tag{2.3}$$

Laws that follow from the above definitions (Definitions 38 to 40).

1. The conjugate of the Event occurring, i.e. the Event **not** occurring is:

$$\mathrm{P}(\bar{A}) = 1 - \mathrm{P}(A) \tag{2.4}$$

2. The probability of the union of 2 Events is:

$$\mathrm{P}(A \cap B) = \mathrm{P}(A) + \mathrm{P}(B) - \mathrm{P}(A \cup B) \tag{2.5}$$

   - If A and B are Mutually Exclusive, then $\mathrm{P}(A \cup B) = 0$.

2.2 Conditional Probability

Defn 41 (Conditional Probability)**.** *Conditional probability* is the probability of an Event occurring when it is known that another Event occurred.

$$\mathrm{P}(A \,|\, B) = \frac{\mathrm{P}(A \cap B)}{\mathrm{P}(B)} \tag{2.6}$$

Defn 42 (Independent)**.** Events are *independent* if the probability of the events' intersection is the same as their probabilities multipled together.

$$\mathrm{P}(A \cap B) = \mathrm{P}(A)\,\mathrm{P}(B) \tag{2.7}$$

Remark 42.1 (Conditional Probability and Independent Events). If A and B are Events and are Independent, then

$$\begin{aligned}\mathrm{P}(A \,|\, B) &= \mathrm{P}(A) \\ \mathrm{P}(B \,|\, A) &= \mathrm{P}(B)\end{aligned} \tag{2.8}$$

2.3 Random Variables

Defn 43 (Random Variable)**.** A *random variable* is a mapping from an Event's outcome to a real number.

There are 2 types of random variables, based on what the mapping ends up with:

1. Discrete Random Variables are mapped to integers, $\mathbb{Z}$.
2. Continuous Random Variables are mapped to the real numbers, $\mathbb{R}$.

2.3.1 Discrete Random Variables

Defn 44 (Discrete Random Variable)**.** A *Discrete Random Variable* is one whose values are mapped from an Event's outcome to the integer numbers ($\mathbb{Z}$). These Random Variables are drawn from outcomes that are finite (sides on a die) or countably infinite.

The probability of a single value of the discrete random variable is denoted differently here than in the course material. The subscript refers to which discrete random variable we are working with (in this case X) and the variable in parentheses is the value we are calculating for (in this case $x \in X$).

$$p_X(x) \tag{2.9}$$

The sum of all probabilities for values that the discrete random variable can take **must** sum to 1.

$$\sum_{x \in X} p_X(x) = 1 \tag{2.10}$$

The mean or expected value of a discrete random variables is shown below:

$$\begin{aligned}\mu &= \sum_{x \in X} x p_X(x) \\ \mathbb{E}[X] &= \sum_{x \in X} x p_X(x)\end{aligned} \tag{2.11}$$

The variance of a discrete random variable is how "off" a value from the random variable is from the mean/expected value.

$$\begin{aligned}\sigma^2 &= \sum_{x \in X} (x - \mu)^2 p_X(x) \\ \mathrm{VAR}[X] &= \sum_{x \in X} \big(x - \mathbb{E}[X]\big)^2 p_X(x)\end{aligned} \tag{2.12}$$

The standard deviation is the square root of the variance.

$$\begin{aligned}\sigma &= \sqrt{\sigma^2} = \sqrt{\sum_{x \in X} (x-\mu)^2 p_X(x)} \\ \text{STD}[X] &= \sqrt{\text{VAR}[X]} = \sqrt{\sum_{x \in X} \left(x - \mathbb{E}[X]\right)^2 p_X(x)}\end{aligned} \tag{2.13}$$

There are 5 different Discrete Random Variable distributions that we will be heavily utilizing in this course.

2.3.1.1 Uniform Random Variable

Defn 45 (Uniform Random Variable)**.** The *uniform random variable* is a Discrete Random Variable whose probabilities for each outcome is equal.

For a Discrete Random Variable X, which has $|X|$ possible values,

$$p_X(x) = \frac{1}{|X|} \tag{2.14}$$

Example 2.1: Uniform Random Variable. Lecture 1

For example, the roll of a die is typically modelled as a uniform random variable. Find the probability distribution function, the expected value, and the variance.

Let's assume this is a 6-sided die. And let's map each side's number to a value in the range of $X \in [1, 6]$.
Using Equation (2.14), we can find the probability distribution easily.

$$p_X(x) = \begin{cases} \frac{1}{6} & x = 1 \\ \frac{1}{6} & x = 2 \\ \frac{1}{6} & x = 3 \\ \frac{1}{6} & x = 4 \\ \frac{1}{6} & x = 5 \\ \frac{1}{6} & x = 6 \end{cases}$$

Using Equation (2.11), we can find the the expected value/mean.

$$\begin{aligned}\mu = \mathbb{E}[X] &= \sum_{x=1}^{6} x p_X(x) \\ &= \frac{1}{6} + \frac{2}{6} + \frac{3}{6} + \frac{4}{6} + \frac{5}{6} + \frac{6}{6} \\ &= \frac{1+2+3+4+5+6}{6} \\ &= \frac{21}{6} = 3.5\end{aligned}$$

Using Equation (2.12), we can find the variance.

$$\begin{aligned}\sigma^2 = \text{VAR}[X] &= \sum_{x=1}^{6} \left(x - \mathbb{E}[X]\right)^2 p_X(x) \\ &= \sum_{x=1}^{6} (x - 3.5)^2 \left(\frac{1}{6}\right) \\ &= 2.91667\end{aligned}$$

Using Equation (2.13), we can find the standard deviation.

$$\begin{aligned}\sigma = \text{STD}[X] &= \sqrt{\sum_{x=1}^{6} \left(x - \mathbb{E}[X]\right)^2 p_X(x)} \\ &= \sqrt{\sum_{x=1}^{6} (x-3.5)^2 \left(\frac{1}{6}\right)} \\ &= \sqrt{2.91667} \\ &= 1.70783\end{aligned}$$

2.3.1.2 Bernoulli Random Variable

Defn 46 (Bernoulli Random Variable)**.** The *Bernoulli random variable* is one where **only one** test occurs, and there are only 2 outcomes.

The probability of success is denoted

$$p_X(\text{success}) = p \tag{2.15}$$

The probability of failure is denoted

$$p_X(\text{failure}) = 1 - p \tag{2.16}$$

The mean/expected value is:

$$\mu = \mathbb{E}[X] = p \tag{2.17}$$

The variance is:

$$\sigma^2 = \text{VAR}[X] = (1-p)p \tag{2.18}$$

2.3.1.3 Binomial Random Variable

Defn 47 (Binomial Random Variable)**.** The *binomial random variable* is one where n trials are run with no stops for a success, where the Random Variable in each run is a Bernoulli Random Variable.

The probability of k successes with n trials is

$$\binom{n}{k} p^k (1-p)^{n-k} = \frac{n!}{k!(n-k)!}(1-p)^{n-k} \tag{2.19}$$

The mean/expected value after n trials is

$$\mu = \mathbb{E}[X] = np \tag{2.20}$$

The variance after n trials is

$$\sigma^2 = \text{VAR}[X] = np(1-p) \tag{2.21}$$

2.3.1.4 Geometric Random Variable

Defn 48 (Geometric Random Variable)**.** The *geometric random variable* is one where n trials are run, where the nth trial is a success, meaning there are $n-1$ previous failures. The Random Variable in each run is a Bernoulli Random Variable.

This means **each trial** has a probability of success of

$$p_X(\text{success}) = p \tag{2.22}$$

And **each trial** has a probability of failure of

$$p_X(\text{failure}) = 1 - p \tag{2.23}$$

The mean/expected value is

$$\mu = \mathbb{E}[X] = \frac{1}{p} \tag{2.24}$$

The variance is

$$\sigma^2 = \text{VAR}[X] = \frac{1-p}{p^2} \tag{2.25}$$

2.3.1.5 Poisson Random Variable

Defn 49 (Poisson Random Variable)**.** The *Poisson random variable* is used to model the number of independent events that occur over a given period of time.

The Poisson random variable has one parameter,

$$\lambda \tag{2.26}$$

λ is the average number of events per unit of time.

The probability function for the value of $x \in X$ of this random variable is

$$p_X(x) = e^{-\lambda} \left(\frac{\lambda^x}{k!} \right) \tag{2.27}$$

The mean/expected value is

$$\mu = \mathbb{E}[X] = \lambda \tag{2.28}$$

The variance is

$$\sigma^2 = \text{VAR}[X] = \lambda \tag{2.29}$$

2.3.2 Continuous Random Variables

Defn 50 (Continuous Random Variable)**.** A *Continuous Random Variable* is one whose values are mapped from an Event's outcome to the real numbers ($\mathbb{R}$).

Continuous random variables cannot be calculated at single points, but must be integrated over a range. This is called the Cumulative Distribution Function (CDF). The subscript refers to the random variable we are using, and x is the value being calculated for.

$$F_X(x) = \text{P}(X \leq x) \tag{2.30}$$

Continuous random variables have a Probability Density Function (PDF), which is the derivative of the CDF.

$$f_X(x) = \frac{d}{dx} F_X(x) \tag{2.31}$$

This PDF must integrate to 1

$$\int\limits_{x \in X} f_X(x) = 1 \tag{2.32}$$

The expected value/mean is

$$\begin{aligned} \mu &= \int\limits_{x \in X} x f_X(x) \, dx \\ \mathbb{E}[X] &= \int\limits_{x \in X} x f_X(x) \, dx \end{aligned} \tag{2.33}$$

The variance is

$$\begin{aligned} \sigma^2 &= \int\limits_{x \in X} (x - \mu)^2 f_X(x) \, dx \\ \text{VAR}[X] &= \int\limits_{x \in X} \big(x - \mathbb{E}[X]\big)^2 f_X(x) \, dx \end{aligned} \tag{2.34}$$

2.3.2.1 Negative Exponential Random Variable

Defn 51 (Negative Exponential Random Variable)**.** The *negative exponential random variable*, sometimes shortened to exponential random variable, is a Continuous Random Variable that has 1 parameter μ, the rate of decay.

The negative exponential random variable has a PDF of

$$f_X(x) = \mu e^{-\mu x} \tag{2.35}$$

The negative exponential random variable has a CDF of

$$F_X(x) = 1 - e^{-\mu x} \tag{2.36}$$

Its mean/expected value is

$$\begin{aligned} \mu &= \frac{1}{\mu} \\ \mathbb{E}[X] &= \frac{1}{\mu} \end{aligned} \tag{2.37}$$

Its variance is

$$\begin{aligned} \sigma^2 &= \left(\frac{1}{\mu}\right)^2 \\ \text{VAR}[X] &= \left(\frac{1}{\mu}\right)^2 \end{aligned} \tag{2.38}$$

Its standard deviation is

$$\begin{aligned} \sigma &= \sqrt{\sigma^2} = \frac{1}{\mu} \\ \text{STD}[X] &= \sqrt{\text{VAR}[X]} = \frac{1}{\mu} \end{aligned} \tag{2.39}$$

2.3.2.2 Gaussian Random Variable

Defn 52 (Gaussian Random Variable)**.** The *Gaussian random variable*, sometimes called the normal random variable, has 2 parameters, μ and σ.

The Probability Density Function (PDF) is

$$f_X(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x-\mu^2}{2\sigma^2}} \tag{2.40}$$

The mean/expected value is

$$\begin{aligned} \mu &= \mu \\ \mathbb{E}[X] &= \mu \end{aligned} \tag{2.41}$$

The variance is

$$\begin{aligned} \sigma^2 &= \sigma^2 \\ \text{VAR}[X] &= \sigma^2 \end{aligned} \tag{2.42}$$

The standard deviation is

$$\begin{aligned} \sigma &= \sqrt{\sigma^2} = \sigma \\ \text{STD}[X] &= \sqrt{\text{VAR}[X]} = \sigma \end{aligned} \tag{2.43}$$

2.4 Multiple Random Variables

Defn 53 (Joint Probability Function)**.** If X, Y are Discrete Random Variables, the probability of the joint event occurring is

$$p_{X,Y}(x, y) \tag{2.44}$$

where the subscripted X, Y refers to the Discrete Random Variables in use and x, y refers to the values being calculated for.

$$\text{P}(x, y) = \text{P}(X = x, Y = y) \tag{2.45}$$

Defn 54 (Joint Cumulative Distribution Function)**.** If X, Y are Continuous Random Variables, then their *joint cumulative distribution function* is

$$F_{X,Y}(x, y) = \text{P}(X \leq x, Y \leq y) \tag{2.46}$$

Remark 54.1 (Independent Multiple Random Variables). X, Y are independent if and only if

$$F_{X,Y}(x, y) = F_X(x)F_Y(y) \tag{2.47}$$

Defn 55 (Joint Probability Density Function)**.** If X, Y are Continuous Random Variables, then their *joint probability density function* is

$$f_{X,Y}(x, y) = \frac{\partial^2}{\partial x \partial y} F_{X,Y}(x, y) \tag{2.48}$$

2.5 Properties of Expected Value

(i) Expected Value obeys the laws of linearity.

$$\mathbb{E}[aX + b] = a\,\mathbb{E}[X] + b \tag{2.49}$$

(ii) For any Random Variables X, Y

$$\mathbb{E}[X + Y] = \mathbb{E}[X] + \mathbb{E}[Y] \tag{2.50}$$

(iii) For any Independent Multiple Random Variables X, Y,

$$\mathbb{E}[XY] = \mathbb{E}[X]\,\mathbb{E}[Y] \tag{2.51}$$

2.6 Properties of Variance

(i) Variance does not obey the laws of linearity.

$$\mathrm{VAR}[aX + b] = a^2\,\mathrm{VAR}[X] \tag{2.52}$$

(ii) For any Independent Multiple Random Variables X, Y,

$$\mathrm{VAR}[X + Y] = \mathrm{VAR}[X] + \mathrm{VAR}[Y] \tag{2.53}$$

2.7 Covariance

Defn 56 (Covariance)**.** The *covariance* of two Random Variables is defined as

$$\begin{aligned}\mathrm{Cov}[X, Y] &= \mathbb{E}\Big[\big(X - \mathbb{E}[X]\big)\big(Y - \mathbb{E}[Y]\big)\Big] \\ &= \mathbb{E}[XY] - \mathbb{E}[X]\,\mathbb{E}[Y]\end{aligned} \tag{2.54}$$

Remark 56.1 (Relationship Between Covariance and Variance). Variance is actually a case of Covariance of a Random Variable with itself.

$$\mathrm{VAR}[X] = \mathrm{Cov}[X, X] \tag{2.55}$$

The Covariance of two Random Variables can have 3 possible values:

1. Positive: If one Random Variable increases, the other does too.
2. Negative: If one Random Variable increases, the other decreases.
3. Zero: If one Random Variable increases, the other does nothing.

2.8 Correlation Coefficient

Defn 57 (Correlation)**.** The *correlation of X and Y* is defined as the $1, 1$ moment.

$$\mathbb{E}\left[X^1 Y^1\right] \tag{2.56}$$

Defn 58 (Correlation Coefficient)**.** The *correlation coefficient of X and Y* is a measure of the **LINEAR relationship** between X and Y. It does not say anything about nonlinear dependence. It is defined as

$$\begin{aligned}\rho_{X,Y} &= \frac{\mathrm{Cov}[X, Y]}{\sigma_X \sigma_Y} \\ &= \frac{\mathrm{Cov}[X, Y]}{\mathrm{STD}[X]\,\mathrm{STD}[Y]}\end{aligned} \tag{2.57}$$

Remark 58.1. $\rho_{X,Y}$ only ranges from $-1 \leq \rho_{X,Y} \leq 1$.

Remark 58.2. The closer $\rho_{X,Y}$ are to $+1$, the closer X and Y are to having a positive linear relationship. The closer $\rho_{X,Y}$ are to -1, the closer X and Y are to having a negative linear relationship. The closer $\rho_{X,Y}$ are to 0, the closer X and Y are to being *uncorrelated.*

If $\rho_{X,Y} = 0$, then

- If X, Y are Independent Multiple Random Variables, then they are uncorrelated.
- If X, Y are uncorrelated ($\rho_{X,Y} = 0$), then they may still **not** be independent.

2.9 Stochastic Processes

Defn 59 (Stochastic Process)**.** A *stochastic process* or random process $\mathbf{x}(t)$ has 2 meanings:

1. For every time instant, $x(t)$ is a Random Variable.
2. For every point (sample) in an outcome space Ω, $x(t)$ is a real-valued function of time.

There are 4 different possible types:

1. Discrete-Time and discrete-value
2. Discrete-Time and continuous-value
3. Continuous-Time and discrete-value
    - Packet arrival to destination over time.
4. Continuous-Time and continuous-value
    - End-to-end delay in a network.

Defn 60 (Stationary Process)**.** A Stochastic Process $x(t)$ is called (weakly) stationary if:

- $\mathbb{E}[x(t)]$ is a constant, it is independent of t.
- The Correlation or Covariance of the process at 2 points in time $x(t_1)$ and $x(t_2)$, is a function of the difference $t_2 - t_1$ **only**.

2.10 The Poisson Process

Defn 61 (The Poisson Process)**.** This is a continuous-time, discrete-value Stochastic Process. It is also a Stationary Process. This process in **memoryless**, the previous time instant's values do not affect this time instant's value.

This is very commonly used to describe the arrivals into a queueing system or network. There is a parameter λ that is the average rate (packets per second, in this case).

There are 3 different definitions for this process:

1. Behaviour for a very small interval of time.
    - Approximately a Bernoulli Random Variable.
    - The time interval is small enough such that only 1 event occurs with probability $\lambda\Delta t + o(\Delta t)$, which becomes $\lambda\Delta t$ for small Δt.
    - 0 events occur with probability $1 - \lambda\Delta t + o(\Delta t)$, which becomes $1 - \lambda\Delta t$ for small Δt.
    - Probabilities between non-overlapping intervals are independent
2. Behaviour over a longer period of time.
    - Receive multiple events, k.
    - Probability of k events is $p_X(k) = e^{-\lambda}\frac{\lambda^k}{k!}$
3. Behaviour between events.
    - Approximately a Negative Exponential Random Variable.
    - Time t between events, $p(t) = \lambda e^{-\lambda t}$

Poisson Process Definition 1/2 Equivalence. Find the probability of k arrivals in time t. Divide the period t into N small periods of Δt each such that as $N \to \infty$, $\Delta t \to 0$.

Approximate the sum of N independent Bernoulli Random Variables with probability $\frac{\lambda t}{N}$ each.

The probability that k out of N will be 1 and the rest 0 is:

$$\frac{N!}{k!(N-k)!}\left(\lambda\frac{t}{N}\right)^k\left(1-\lambda\frac{t}{N}\right)^{N-k}$$

When N is large, 3 things happen.

1. $$\frac{N!}{(N-k)!} \approx N^k \tag{2.58}$$
2. $$\left(1-\lambda\frac{t}{N}\right)^N \approx e^{-\lambda t} \tag{2.59}$$
3. $$\left(1-\lambda\frac{t}{N}\right)^{-k} \approx 1 \tag{2.60}$$

So, when $N \to \infty$, we obtain

$$\mathrm{P}(k \text{ arrivals in time } t) = e^{-\lambda t}\frac{(\lambda t)^k}{k!}$$

■

Poisson Process Definition 2/3 Equivalence. Since

$$\mathrm{P}(k \text{ arrivals in time } t) = e^{-\lambda t}\frac{(\lambda t)^k}{k!}$$

The probability that time between arrivals is more than t is the same as the probability of 0 arrivals during time $t = e^{-\lambda t}$. So, this is exponentially distributed as required. ■

Poisson Process Definition 3/1 Equivalence. Suppose that the last arrival was time t ago.

A priori probability that the next arrival comes after more than t from the previous arrival is $e^{-\lambda t}$.
A priori probability this next arrival is after more than $t + \Delta t = e^{-\lambda(t+\Delta t)}$.
A priori probability that this next arrival happens within the next $\Delta t = e^{-\lambda t} - e^{-\lambda(t+\Delta t)}$.

Conditioning this on knowing that there were no other arrivals in the last t time, and using a Taylor expansion:

$$\frac{e^{-\lambda t} - e^{-\lambda(t+\Delta t)}}{e^{-\lambda t}} = 1 - e^{-\lambda \Delta t} \qquad \approx \lambda \Delta t - \frac{(\lambda \Delta t)^2}{2} + \frac{(\lambda \Delta t)^3}{6} - \dots$$

Since this result is independent of t, this satisfies the requirement of the independence from previously arrived messages. ■

2.10.1 Sums of the Poisson Processes

The superposition of several different Poisson processes.

- There are m independent Poisson Processes, with rates λ_i for $i = 1, 2, \dots, m$
- Sum is also a Poisson Process: $\lambda = \sum_{i=1}^{m} \lambda_i$

All of this comes together for the equation below. It models the number of things k that have arrived over a certain period of time t.

$$\mathrm{P}(\mathbf{x}(t) = k) = e^{-\lambda t}\frac{(\lambda t)^k}{k!} \tag{2.61}$$

3 Performance Evaluation

We do this to:

- Evaluate existing systems
- Design new network systems
- Predict system behaviors under different conditions

3.1 Performance Measures

How do we measure the performance of a large complex network?

- Data transfer speed
- Reliability:
  - Guaranteed throughput
  - Guarantee of any other performance measurement
  - Integrity of data
  - Predictability of errors
  - Uptime/Downtime/Availability
- Security
- User satisfaction
- Sustainability
- Maintainability
- Throughput/Goodput
- Delay/Latency
- Energy Efficiency
- Jitter (Delay variance)
- Packet Loss

3.2 Performance Evaluation

How can we evaluate the performance of a large complex network?

- Analysis: Mathematical modeling, calculations.
- Simulation: Software implementation of system model.
- Real-World Experimentation: Testing the actual system.

Analysis	Simulation	Experimentation
— Requires detailed understanding of system properties	+ Only requires modeling the environment with a straightforward implementation	++ No modeling or understanding of how the system required
— Usually requires approximations and simplifying assumptions.	+ Possible to implement complex details of system without approximation	++ Captures complete behavior of system and environment without approximation.
++ Allows for deep insight for a broad range of scenarios.	+ Allows insight to broad range of scenarios.	— Requires deployment of every scenarios tested and may be difficult to reproduce.
+ Rare events and boundary cases are included.	+ Study of rare events is tricky, but possible.	— Rare events may be impossible to study.

Table 3.1: Performance Evaluation Pros and Cons

3.3 Statistical Data Analysis

Only analysis produces exact results. Simulation and experimentation produce samples from some underlying random distribution. This means we need to perform statistical analysis of these results.

3.3.1 Sample Mean

We assume a random variable Z with an unknown probability distribution, but we can assume a distribution to start with. We estimate the key distribution metrics:

- Mean (1st moment)
- Variance (2nd moment)
- Variance of the variance (3rd moment)

We obtain n **independent** samples, $z_1, z_2, \dots, z_n$. To estimate the mean/expected value, we use the equation below.

$$\bar{z} = \frac{1}{n}\sum_{i=1}^{n} z_i \tag{3.1}$$

Where $\bar{z}$ is also a Random Variable. So, we can perform an expected value calculation on $\bar{z}$.

$$\mathbb{E}[\bar{z}] = \mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n} z_i\right] = \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}[z_i] \tag{3.2}$$

So, as $n \to \infty$, $\mathbb{E}[\bar{z}] \to \mu$.

3.3.2 Sample Variance

Now that we have found the sample mean (Equation (3.1)), we can attempt to find the variance.

We start by estimating the variance:

$$\begin{aligned}
\text{VAR}\big[\mathbb{E}[\bar{z}]\big] &= \mathbb{E}\left[(\bar{z}-\mu)^2\right] \\
(\bar{z}-\mu)^2 &= \frac{1}{n^2}\left(\sum_{i=1}^{n}(z_i-\mu)\right)^2 \\
&= \frac{1}{n^2}\sum_{i=1}^{n}(z_i-\mu)^2 + \sum_{i=1}^{n}\sum_{j\neq i}(z_i-\mu)(z_j-\mu) \\
\mathbb{E}\left[(\bar{z}-\mu)^2\right] &= \frac{1}{n^2}\sum_{i=1}^{n}\mathbb{E}\left[(z_i-\mu)^2\right] + \sum_{i=1}^{n}\sum_{j\neq i}\mathbb{E}[(z_i-\mu)]\,\mathbb{E}[(z_j-\mu)] \\
&= \frac{1}{n^2}\left(n\sigma^2+0\right) \\
&= \frac{\sigma^2}{n}
\end{aligned}$$

The estimated variance is:

$$\text{VAR}[\bar{z}] = \frac{\sigma^2}{n}$$

meaning that the variance of the sample mean $\bar{z}$ around the population mean μ decreases as the number of trials n increases.

The sample variance is the variance of the sample data around its mean.

$$\widehat{V} = \frac{1}{n}\sum_{i=1}^{n}(z_i-\bar{z})^2 \tag{3.3}$$

The expected value of the sample variance, $\mathbb{E}[\widehat{V}]$ is

$$\mathbb{E}[\widehat{V}] = \frac{n-1}{n}\sigma^2$$

This means that the expected value of the variance is actually slightly biased by the $n-1$ numerator. To correct for this, we use Bessel's Correction.

Defn 62 (Bessel's Correction)**.** The division of $n-1$ instead of n in Equation (3.4) is called *Bessel's Correction.*

Defn 63 (Unbiased Sample Variance)**.** By using Definition 62, instead of the normal n value, we find ourselves with the *unbiased sample variance.*

$$s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(z_i-\bar{z})^2 \tag{3.4}$$

3.3.3 Confidence Intervals

Defn 64 (Confidence Interval)**.** *Confidence interval*s are a tool to describe how much we can trust the set of results that we gather. Essentially, how confident we are in the results that we gathered. More mathematically, they describe how sure we are (95%-confident, 99%-confident, etc.) we are that the data point we gathered is the same as the underlying distribution's value.

The confidence interval is derived from the Gaussian Random Variable. Thus, it is only useful if the sample size is large, because of the Central Limit Theorem. It is derived from the below equation:

$$\text{P}\left(|\bar{z}-\mu| \leq \alpha\frac{\sigma}{\sqrt{n}}\right) \tag{3.5}$$

Some common values of this are:

- $= 0.9$ for $\alpha = 1.645$
- $= 0.95$ for $\alpha = 1.96$
- $= 0.99$ for $\alpha = 2.576$

The actual interval is calculated with this equation:

$$\left[\bar{z}-\alpha\frac{\sigma}{\sqrt{n}}, \bar{z}+\alpha\frac{\sigma}{\sqrt{n}}\right] \tag{3.6}$$

However, because the standard deviation σ is not known, we estimate using the square root of the Unbiased Sample Variance $\sqrt{s^2}$ instead. So, the actual confidence interval is:

$$\left[\bar{z} - \alpha\frac{\sqrt{s^2}}{\sqrt{n}}, \bar{z} + \alpha\frac{\sqrt{s^2}}{\sqrt{n}}\right] \tag{3.7}$$

Remark 64.1 (Stricter Definition of Confidence Intervals)*.* More strictly speaking, the confidence level represents the frequency (i.e. the proportion) of possible confidence intervals that contain the true value of the unknown population parameter. In other words, if confidence intervals are constructed using a given confidence level from an infinite number of independent sample statistics, the proportion of those intervals that contain the true value of the parameter will be equal to the confidence level. For example, if the confidence level (CL) is 90% then in hypothetical indefinite data collection, in 90% of the samples the interval estimate will contain the true population parameter.

4 Queuing Theory

In queuing theory, we view a network as a collection as first-in-first-out (FIFO) data structures. Queuing theory provides a probabilistic analysis of these queues.

Defn 65 (Queuing System)**.** In a *queuing system* model, there is a FIFO queue with a task arrival rate of λ. The server processes these with a service time of μ.

4.1 Little's Law

Defn 66 (Little's Law)**.** *Little's Law* or *Little's Formula* models the mean number of tasks in a queue-based system. This applies to **any system in equilibrium**, as long as the system does not create or destroy tasks itself.

$$r = \lambda T_r \tag{4.1}$$

- r — The mean number of tasks in a queuing system.
- λ — The average arrival rate of tasks to the system.
- T_r — The mean time for which the task sits in the queue waiting.

Example 4.1: Application of Little's Law. Lecture 3

If 40 customers visit a pub per hour, and these customers spend an average of 15 minutes in the pub, what is the average number of customers in the pub at any given time?

Using Equation (4.1),

$$\lambda = 40$$
$$T_r = \frac{15}{60} = \frac{1}{4} = 0.25$$

Multiplying these together,

$$r = 40 * \frac{1}{4} = 10$$

4.2 Kendall's Notation

Defn 67 (Kendall's Notation)**.** *Kendall's Notation* is a shorthand to specify the characteristics of a Queuing System. There are 6 symbols, where the first 2 are distributions, the next 3 are numbers, and the last is the discipline the server uses to process tasks.

$$x_1/x_2/x_3/x_4/x_5/x_6 \tag{4.2}$$

1. x_1: The arrival distribution (See Section 4.2.1).
2. x_2: The server's service distribution (See Section 4.2.1).
3. x_3: The number of servers.
4. x_4: The total capacity of the system (Assumed to be ∞ if not specified).
   - The assumption of ∞ is not a bad assumption usually.

- Packets are usually only a couple of kibibytes, whereas main memory is gibibytes.

5. x_5: The population size (the total possible tasks, assumed to be ∞ unless specified).
6. x_6: Service Discipline (FIFO, LIFO, etc.) (FIFO is assumed, unless specified).

Remark 67.1 (Shortened Kendall's Notation). Typically, only the first 3 terms in Kendall's Notation are used. The last 3 have assumed conditions if they are not specified.

4.2.1 Distributions in Kendall's Notation

- M — Exponential, Memoryless, Markovian, Poissonian.
- D — Deterministic, Fixed arrival rate.
- E_k — Erlang with a parameter k.
- H_k — Hyperexponential with a parameter k.
- G — General (The distribution can be anything).

Some examples are:

Example 4.2: Kendall's Notation 1. Lecture 3

What does $M/M/1$ mean in Kendall's Notation?

- Really $M/M/1/\infty/\infty/FIFO$, since last 3 not specified.
- Exponential arrival distribution
- Exponential service distribution
- 1 server
- Infinite capacity
- Infinite population
- FCFS (FIFO)

Example 4.3: Kendall's Notation 2. Lecture 3

What does $M/M/n$ mean in Kendall's Notation?

- Really $M/M/n/\infty/\infty/FIFO$, since last 3 notation specified.
- Exponential arrival distribution
- Exponential service
- n servers
- Infinite capacity
- Infinite population
- FCFS (FIFO)

Example 4.4: Kendall's Notation 3. Lecture 3

What does $G/G/3/20/1500/SPF$ mean in Kendall's Notation?

- General arrival distribution
- General service distribution
- 3 servers
- 17 queue slots $(20 - 3)$
- 1500 total jobs
- Shortest Packet First

4.3 A General $M/M/1$ Queue

If we have a general $M/M/1$ queue, then:

- We may assume messages arrive at the channel according to The Poisson Process at a rate λ messages/sec.
- We may assume that the message lengths have a Negative Exponential Random Variable with a mean of $\frac{1}{v}$ bits/message.
- The channel transmits messages from its buffer at a constant rate c bits/sec.

- The buffer associated with the channel can be considered to be effectively of infinite length.

Defn 68 (Message Transmission Rate)**.** The *message tramisssion rate*, c, is the rate at which tasks are moved from the queue/buffer to the server.

Defn 69 (Message Arrival Rate)**.** The *message arrival rate*, denoted μ is defined to be the number of messages that are received per unit time.

$$\mu = cv \tag{4.3}$$

- c: Message Transmission Rate
- v: Message length

Defn 70 (Occupancy)**.** The *occupancy* of a system, denoted ρ, is the factor to which the rate of message arrivals is greater than the message transmissions after processing.

$$\rho = \frac{\lambda}{\mu} \tag{4.4}$$

- λ: The message arrival rate.
- μ: The message tramisssion rate after processing.

Remark 70.1 (Steady-State Solution). The only steady-state solution for infinite-buffer systems occurs when $\rho < 1$. If $\rho \geq 1$, then packets are arriving at the same or greater rate as they are pushed out after processing. Meaning, eventually the queue will fill up.

Remark 70.2 (Utilization). Sometimes Occupancy is referred to as *utilization*. In the end, they mean the same thing.

If we want to find the average number of messages in the system, we need to find $\mathbb{E}[R]$.

$$\mathbb{E}[R] = \frac{\rho}{1-\rho} \tag{4.5}$$

The average number of packets in **just** the queue is given by $\mathbb{E}[T_W]$.

$$\mathbb{E}[T_W] = \frac{1}{\mu}\frac{\rho}{1-\rho} \tag{4.6}$$

The mean delay of a packet due to the system is given by $\mathbb{E}[T_R]$.

$$\mathbb{E}[T_R] = \frac{1}{\mu}\frac{1}{1-\rho} \tag{4.7}$$

Defn 71 (Probability of Delay)**.** The *probability of delay for an $M/M/1$ queuing system* for a given amount of time is given by

$$p_{T_R}(t) = \mu(1-\rho)e^{-\mu(1-\rho)t} \tag{4.8}$$

The probability the delay is within a given range is given by the equation below.

$$\mathrm{P}(T_R \leq t) = 1 - e^{-\mu(1-\rho)t} \tag{4.9}$$

4.4 State-Dependent Queues

Defn 72 (State-Dependent Queue)**.** A *state-dependent queue* is a queue that behaves in certain ways depending on the state that it is in. It can cycle endlessly **between** states, but can never cycle on itself.

There are 2 variables needed for each state.

1. λ_i: The arrival rate of tasks when the system is in state i.
2. μ_i: The servers' service rate when the system is in state i.

In each state, there needs to be a balance between the packets going to the next stage and leaving the current one and vice versa.

$$\lambda_{i-1}p_{i-1} = \mu_i p_i, \text{ for } i = 1, 2, \ldots \tag{4.10}$$

The general solution to Equation (4.10) is:

$$p_i = \frac{\lambda_0\lambda_1\cdots\lambda_{i-1}}{\mu_1\mu_2\cdots\mu_i}p_0, \text{ for } i = 1, 2, \ldots \tag{4.11}$$

4.4.1 A General $M/M/2$ Queue

A general $M/M/2$ Queuing System can be represented as a State-Dependent Queue.

$$\lambda_i = \lambda, \text{ for } i = 1, 2, \dots$$
$$\mu_i = \begin{cases} \mu & i = 1 \\ 2\mu & i = 2, 3, \dots \end{cases}$$

where i is the number of packets in the system, which represents a different state of the system.

Now, let's compare a $M/M/2$ Queuing System against a $M/M/1$ Queuing System.

$$\begin{aligned}
\mathbb{E}[T_R]_{1 \text{ server, } 2\mu \text{ rate}} &= \frac{1}{2\mu}\frac{1}{1-\rho} \\
\mathbb{E}[T_R]_{2 \text{ servers}} &= \frac{1}{\mu}\frac{1}{(1+\rho)(1-\rho)} \\
\frac{\mathbb{E}[T_R]_{1 \text{ server, } 2\mu \text{ rate}}}{\mathbb{E}[T_R]_{2 \text{ servers}}} &= \frac{1+\rho}{2} \leq 2
\end{aligned}$$

The conclusion here is concrete.

A single faster server is always more efficient.

If we extend our findings about State-Dependent Queues to $M/M/\infty$ systems, the general solution becomes a Poisson Random Variable's distribution.

$$\begin{aligned}
p_i &= \frac{\lambda_i}{\mu^i i!} p_0 \\
&= e^{-\frac{\lambda}{\mu}} \frac{\left(\frac{\lambda}{\mu}\right)^i}{i!} \\
&= e^{-A} \frac{A^i}{i!}
\end{aligned} \tag{4.12}$$

4.5 The Finite Buffer $M/M/1/N$ Queue

The $M/M/1/n$ is a queue with a finite-sized buffer. The real question with this is what is the probability the buffer will be full and the next packet will have to be blocked.

Defn 73 (Blocking Probability)**.** The *blocking probability* is the probability that the next packet that arrives to the system will be blocked because the queue is full. Instead of making the queue longer to keep track of the packets, the system just denies the packet access to the queue.

$$P_B = \text{P[Message is blocked]} = p_n = \frac{1-\rho}{1-\rho^{n+1}}\rho^n \tag{4.13}$$

Example 4.5: Length of Queue for No Packets Blocked. Lecture 4

Assume $\rho = 0.5$, and that we want a system that blocks packets with probability $P_B \leq 10^{-3}$. Find n, the minimum length of the queue to satisfy these requirements?

Using Equation (4.13),

$$\begin{aligned}
\frac{1-\rho}{1-\rho^{n+1}}\rho^n &= \frac{1-0.5}{1-0.5^{n+1}} 0.5^n \\
&= \frac{0.5^{n+1}}{1-0.5^{n+1}} \leq 10^{-3} \\
0.5^{n+1} &\leq \frac{10^{-3}}{1+10^{-3}} \\
n+1 &\geq 9.96 \\
n &\geq 8.96
\end{aligned}$$

So, n should be 9.

Remark. Note that the solution found in Example 4.5 is actually the general solution. Meaning, that solution for n is valid for all ρ. This is because the finite buffer copes with overloading by blocking any additional messages from entering the system.

4.5.1 Applying Little's Law

To even use Little's Law, we must account for the blocking of messages by only counting the messages that make it into the system.

$$\gamma = \lambda(1 - P_B) \tag{4.14}$$

This means our equations change a little bit.

$$\rho = \frac{\lambda}{\mu}, \text{ can be } \geq 1 \text{ r} < 1 \tag{4.15}$$

$$\begin{aligned}\mathbb{E}[R] &= \gamma\,\mathbb{E}[T_R] \\ &= \lambda(1 - P_B)\,\mathbb{E}[T_R]\end{aligned} \tag{4.16a}$$

$$\begin{aligned}\mathbb{E}[R] &= \sum_{i=0}^{n} i p_i \\ &= \frac{1-\rho}{1-\rho^{n+1}} \sum_{i=0}^{n} i \rho_i\end{aligned} \tag{4.16b}$$

4.6 Modeling Circuit Switching

This is easiest to visualize with the old-school telephone communications with human operators physically connecting calls to make a single large continuous circuit.

- The rate of call initiation can be modeled as a Poisson Random Variable, with an arrival rate of λ.
- The length of the calls can be modeled as a Negative Exponential Random Variable, denoted as h, $\mu = \frac{1}{h}$.
- The number of circuits is fixed at n.
- If all circuits are in use, there is no queue to wait in until one is available.

This makes Circuit Switching a $M/M/n/n$ Queuing System in Kendall's Notation.

Defn 74 (Offered Traffic)**.** *Offered traffic* is the amount of traffic that the end-users are offering the network.

$$A = \lambda h \tag{4.17}$$

Defn 75 (Carried Traffic)**.** *Carried traffic* is the amount of traffic that the network is currently handling.

$$A_C = A(1 - P_B) \tag{4.18}$$

Defn 76 (Lost Traffic)**.** *Lost traffic* is traffic that is lost because all the resources available have been allocated and no more are available. This cannot be regained later because there is no queue available to handle the things that arrive to the system after all resources have been allocated.

$$L = A - A_C \tag{4.19}$$

Remark 76.1 (Validity). Equation (4.19) is valid for all non-negative exponential distributions. However, it is invalidated if repeats occur at all, or if they are allowed to occur, occur to soon.

Let the Random Variable R represent the number of customers currently in the system, meaning R is between 0 and n. The service rate for this is now

$$i\mu, \text{ when } R = i$$

Defn 77 (Blocking Probability)**.** The *blocking probability* is the probability that the next thing that comes to the system will be blocked because all resources have already been allocated and there is no queue.

$$\begin{aligned}P_B &= p_n \\ &= \frac{\frac{A^n}{n!}}{\sum\limits_{i=0}^{n} \frac{A^j}{j!}}\end{aligned} \tag{4.20}$$

Defn 78 (Time Congestion)**.** *Time congestion* represents the proportion of time that all the circuits are busy. This is only viewed from the system-side, so the end-user will never know about time congestion.

To find the time congestion, it is simply the expected value of the Offered Traffic, when in state n.

$$\begin{aligned}\mathbb{E}_n[A] &= P_B \\ &= p_n \\ &= \frac{\frac{A^N}{N!}}{\sum_{j=0}^{n} \frac{A^j}{j!}}\end{aligned} \tag{4.21}$$

Defn 79 (Call Congestion)**.** *Call congestion* is the proportion of calls that find the system busy. This is viewed from the end-user side.

Remark. For arrivals that follow a Poisson Random Variable's distribution, Time Congestion = Call Congestion. This follows according to the PASTA Theorem.

Theorem 4.1 (PASTA Theorem)**.**

4.7 Queuing Networks

Defn 80 (Queuing Network)**.** A *queuing network* is a network of nodes in which each node is a queue. The output of one queue is connected to the input of another node's queue.

There are 2 types of queuing networks:

1. Closed Queuing Network
2. Open Queuing Network

Remark 80.1 (Limiting Analysis). We will only consider $M/M/n$ queues here. Analysis of more general queueing networks gets complicated fast.

Defn 81 (Closed Queuing Network)**.** A *closed queuing network* is one in which packets (customers, tasks, etc.) can never enter or leave the Queuing Network.

Defn 82 (Open Queuing Network)**.** An *open queuing network* is a Queuing Network that is not closed. This means that an open queuing network is one in which packets (customers, tasks, etc.) may enter a node's queue from outside the Queuing Network, and after processing, may leave the Queuing Network entirely.

Theorem 4.2 (Burke's Theorem)**.** *The interdeparture times from a Markovian (M/M/n) queue are exponentially distributed with the same parameter as the interarrival times. Meaning*

$$\lambda_{\text{Out}} = \lambda_{\text{In}} \tag{4.22}$$

Remark 82.1. This allows us to analyze each queue/node independently. We do not need to consider all the nodes at the same time.

4.7.1 Jackson Networks

Defn 83 (Jackson Network)**.** A *Jackson network* is an Open Queuing Network of $M/M/n$ queues. Each node can receive traffic from other nodes and from outside the network. Traffic from each node can go to other nodes, or leave the network entirely.

- N: The total number of nodes in a Jackson Network.
- λ_i: Total incoming average traffic rate to node i.
- γ_i: Average rate of traffic entering node i from outside the network
- r_{ij}: Probability a packet leaving node i will then go to node j
  - Note r need not be 0 – a packet may immediately return to the node it just left
- So the probability that a packet leaves the network after leaving node i is given by

$$1 - \sum_{j=1}^{N} r_{ij} \tag{4.23}$$

- You can find the total incoming traffic to node i by summing all incoming streams together.

$$\lambda_i = \gamma_i + \sum_{j=1}^{N} \lambda_j r_{ji}, \text{ for } i = 1, 2, \ldots, N \tag{4.24}$$

- In general, the whole network will not behave like a The Poisson Process.
- Jackson, the namesake of the Jackson Network, showed that each individual node behaves as though it were.
- The state variable for the entire system of N nodes consists of the vector

$$\langle k_1, k_2, \ldots, k_N \rangle \tag{4.25}$$

  where k_i is the number of packets (including those currently in service) at the ith node.
- Let the equilibrium probability associated with a given state be denoted

$$\mathrm{P}\big[\langle k_1, k_2, \ldots, k_N \rangle\big] \tag{4.26}$$

  and the probability of finding k_i customers in the ith node be

$$\mathrm{P}_i[k_i]$$

- The joint probability distribution for all nodes is the product of the distributions for the individual nodes

$$\mathrm{P}\big[\langle k_1, k_2, \ldots, k_N \rangle\big] = \mathrm{P}_1[k_1]\, \mathrm{P}_2[k_2] \cdots \mathrm{P}_N[k_N]$$

  and each $\mathrm{P}_i[k_i]$ is given by the solution to the classical $M/M/n$ system.
- So we can treat each node as an $M/M/n$ queue, even though the input is not necessarily Markovian.

4.8 Queuing Disciplines

Defn 84 (Queuing Discipline)**.** A *queuing discipline* is a way to order the packets to send them in a particular order. Depending on how we order, namely how we schedule the packets, we achieve different things. However, the overarching requirement is to minimize delay, improve throughput, increase busy time, and decrease idle time.

There are 2 types of queuing disciplines:

1. Work Conserving
2. Non-Work Conserving

Defn 85 (Work Conserving)**.** A *work conserving* Queuing Discipline always handles packets right away. Meaning, if there is a packet waiting in the queue and the server is sitting idle, it will always serve the packet.

This is in contrast to a Non-Work Conserving Queuing Discipline.

Defn 86 (Non-Work Conserving)**.** A *non-work conserving* Queuing Discipline may not handle packets right away. Meaning, if there is a packet waiting in the queue and the server is sitting idle, the server might not server the packet right away. Packets are switched only at **the right time towards the right destination**.

While this type of Queuing Discipline may not make intuitive sense, it has some advantages.

- Reduces jitter
- It *shapes* the traffic, making it predictable by holding traffic back until a certain time.

This is contrast to a Work Conserving Queuing Discipline.

Traditionally, First-In First-Out was used. But, there are some other ones possible that improve the Queuing System in various ways.

4.8.1 First-In First-Out

Defn 87 (First-In First-Out)**.** In a *first-in first-out* Queuing Discipline, the first packet that enters the queue is the first one served. This is a Work Conserving Queuing Discipline. This follows Kleinrock's Conservation Law.

$$C = \sum_{n=1}^{N} \rho_n q_n \tag{4.27}$$

- C: A constant value of throughput for the system.
- ρ: The Occupancy for each flow.

- q: The mean scheduler delay.

However, there are some problems:

- Small packets are held up by large packets ahead of them in the queue. This leads to:
  - Larger average delays for smaller packets.
  - Greater throughput for larger packets.
  - Flows of greedy packets get better service.
- Greedy TCP connections crowd out more altruistic connections.
  - If one connection does not back off, others may back off more.
  - This is a feature of the TCP protocol, because this manages network congestion.

4.8.2 Fair Queuing (FQ)

Defn 88 (Fair Queuing)**.** The *fair queuing* Queuing Discipline is a Work Conserving one. There are multiple queues for each port, one for each source or flow of data to that port. The various queues are serviced in a round-robin fashion, where one packet from a queue is handled in a single cycle, before moving onto the next queue. This achieves load balancing among the different flows, meaning there is no advantage for a particular flow to be greedy.

However, the drawback with this is that short packets are penalized as only one packet from one queue is sent per round.

4.8.3 Priority Queuing (PQ)

Defn 89 (Priority Queuing)**.** *Priority queuing* is a Work Conserving Queuing Discipline. There are K queues that lead to the server. The kth queue is constrained by $1 \leq k \leq K$. The $k+1$th queue has a higher priority than the kth queue. The higher priority queue is served first.

This has a simple implementation with low processing overhead. There needs to be an extra step where the packets are sorted into the correct priority queue, but that is quite efficient now.

However, there is no fairness among the packets. The lower priority queues can be starved of service time if the higher queues are always full.

4.8.4 Processor Sharing (PS)

Defn 90 (Processor Sharing)**.** A *processor sharing* Queuing Discipline is Work Conserving. However, it is not practical, because it is modeled on sending individual bits rather than whole packets. It is more of a meta-Queuing Discipline, because it forms the basis for Bit-Round Fair Queuing.

In this discipline, there are multiple queues, just like in Fair Queuing. One bit from each queue is sent per round of the round-robin. This prevents longer packets from getting an advantage.

The system needs to figure out the virtual start and finish times for a given packet, i, of the α queue.

$$F_i^\alpha = S_i^\alpha + P_i^\alpha \tag{4.28}$$

where

- F_i^α: The αth queue's ith packet's finish time.
- S_i^α: The αth queue's ith packet's start time, which is defined in Equation (4.29).
- P_i^α: The αth queue's ith packet's time to send.

$$S_i^\alpha = \max\left(F_{i-1}^\alpha, A_i^\alpha\right) \tag{4.29}$$

4.8.5 Bit-Round Fair Queuing (BRFQ)

Defn 91 (Bit-Round Fair Queuing)**.** The *bit-round fair queuing* Queuing Discipline is based off the Processor Sharing discipline and inherits its Work Conserving property. However, instead of sending individual bits, we send whole packets, but choose which to send based on the same calculations as Processor Sharing Each data flow gets approximately $\frac{1}{N}$ of the overall bandwidth in the system, given there are N flows.

The virtual start and finish times are still computed. However, unlike the Fair Queuing that Processor Sharing is inspired by, the next packet that is sent is the one with the **earliest finish time**.

The same equations are used as in Processor Sharing (Equations (4.28) to (4.29)).

4.8.6 Generalized Processor Sharing (GPS)

Defn 92 (Generalized Processor Sharing)**.** *Generalized processor sharing* is a Work Conserving meta-Queuing Discipline. Like in Processor Sharing, we send some number of bits per round, specified by the weight w_α. If a queue has a weighting of 5, then $w_\alpha = 5$ means 5 bits are sent from the αth queue each round. This solves the problem in Bit-Round Fair Queuing of not being able to handle queues that output different amounts of information during each round. This is used to form the basis of Weighted Fair Queuing.

There are slight generalizations that need to be made to Equations (4.28) to (4.29). They are shown below.

$$F_i^\alpha = S_i^\alpha + \frac{P_i^\alpha}{w_\alpha} \tag{4.30}$$

where

- F_i^α: The αth queue's ith packet's finish time.
- S_i^α: The αth queue's ith packet's start time, defined in Equation (4.31).
- P_i^α: The αth queue's ith packet's time to send.
- w_α: The weight (number of bits sent per round) of the αth queue.

$$S_i^\alpha = \max\left(F_{i-1}^\alpha, A_i^\alpha\right) \tag{4.31}$$

Remark 92.1 (Service Differentiation). The ability to attach different weights to different queues means we now have a way to respond to different service levels. This leads to the concept of *service differentiation*.

4.8.7 Weighted Fair Queuing (WFQ)

Defn 93 (Weighted Fair Queuing)**.** *Weighted fair queuing* is a Work Conserving implementation of Generalized Processor Sharing. It also uses the same packet management strategy as Bit-Round Fair Queuing, of sending the packet with the **earliest virtual finish time**.

By attaching a weight for the number of possible bits to send, we can guarantee a bound on delay. The maximum buffer/queue size needed is proportional to this delay.

Example 4.6: WFQ vs. BRFQ. Lecture 5

Given the packet reception table below, which arrive at the same time on the same link, find the packet transmission order?

Packet	Size	Flow
1	100	1
2	100	1
3	60	2
4	120	2
5	60	2

You may assume that the link can send one unit of each packet per time instance.

Example 4.7: Packet Transmission Order. Assignment 2

A router has 3 data flows and uses Weighted Fair Queuing. Flow 2 has 3 times as much bandwidth as Flow 1, and Flow 3 had 2 times as much bandwidth as Flow 1. Assuming all these packets arrive at the router at roughly the same time, in what order will the packets be sent?

Packet	**Size (bits)**	**Flow**
1	120	1
2	150	1
3	60	2
4	90	2
5	120	2
6	120	3
7	40	3
8	200	3

First, we need to determine the weightings, $w_\alpha, \alpha \in [1, 3]$

$$w_1 = 1$$
$$w_2 = 3$$
$$w_3 = 2$$

Now, we construct our "transmission table". We can assume that the first packet in each flow has a start time, $S_i = 0$.

Packet	**Size**	**Flow**	$\mathbf{S_i}$	$\frac{P_i}{w_i}$	F_i
1	120	1	0		
2	150	1			
3	60	2	0		
4	90	2			
5	120	2			
6	120	3	0		
7	40	3			
8	200	3			

4.8.8 Class-Based Queuing (CBQ)

Defn 94 (Class-Based Queuing)**.** *Class-based queuing* is a meta-Queuing Discipline, that is neither Work Conserving or Non-Work Conserving. It is a method of assigning fractions of overall bandwidth to a set of nodes within a class. Nodes that are within the same class may borrow bandwidth if the allocated node is not using it.

4.9 Actual IP Network Behavior

Actual IP traffic doesn't actually follow a Poisson Random Variable's distribution very well. It has:

- Self-similarity
- Lots of dependency (Current traffic depends on the previous traffic).
- Infinite variance.

All of this means that all the statistical properties (mean, variance, standard deviation, etc.) of a self-similar process are the same at all timescales. Meaning that considering a larger timescale does not smooth out variations in the data.

Defn 95 (Autocorrelation)**.** *Autocorrelation* is a function that measures the similarity of a function with itself, typically over a period of time.

$$R_{x,x}(t_1, t_2) = \mathbb{E}[X(t_1), X(t_2)] \tag{4.32}$$

Defn 96 (Long-Range Dependence)**.** A process that has *long-range dependence* has an autocorrelation that decays slowly as the difference between t_1 and t_2 tends towards ∞. Mathematically, these processes are characterized by an atocorrelation function that decays hyperbolically as k increases. This is represented with Equation (4.33), the *Non-summability of correlation.*

$$\sum_k r(k) = \infty \tag{4.33}$$

Remark 96.1 (Uncorrelated). If the result from Equation (4.32) is 0 when $t_2 = t_1 + \Delta t$, i.e. when the time difference is very small, then the function $X(t)$ is no correlation.

Remark 96.2 (Short-Range Correlation). if the result from Equation (4.32) decays to 0 very quickly, then the function $X(t)$ has *short-range correlation*.

4.9.1 The Hurst Parameter

Defn 97 (Hurst Parameter)**.** The *Hurst parameter*, denoted H, is a measure of the "burstiness" of a system. It is also considered a measure of self-similarity.

H always lies within the range $0.5 < H < 1.0$.

The expected value of the Hurst parameter is

$$\mathbb{E}[x(t)] = \frac{\mathbb{E}[x(at)]}{a^H} \tag{4.34}$$

The variance of the Hurst parameter is

$$\text{VAR}[x(t)] = \frac{\text{VAR}[x(at)]}{a^{2H}} \tag{4.35}$$

The autocorrelation of the Hurst parameter is

$$R_{x,x}(t,s) = \frac{R_{x,x}(at,as)}{a^{2H}} \tag{4.36}$$

4.9.2 Self-Similarity and Autocorrelation

For this section, let $X(t)$ be a Stochastic Process with

- Constant mean μ
- Constant variance σ^2
- An autocorrelation function depending only on k.

$$r(k) = \frac{\mathbb{E}[(X(t)-\mu)(X(t+k)-\mu)]}{\mathbb{E}\left[\left(X(t)-\mu\right)^2\right]}$$

Now let $X^{(m)}$ be a new **aggregate** time series formed by averages of X over non-overlapping blocks in time of size m. This has some intersting implications.

- X is exactly self-similar if
  - The aggregated processes have the same autocorrelation structure as X.
  - $r^{(m)}(k) = r(k), k \geq 0$ for all $m = 1, 2, \ldots$
- X is asymptotically self-similar if the above holds when $r^{(m)}(k) \sim r(k), m \to \infty$
- The most striking feature of self-similarity is correlation structures of the aggregated process do not degenerate as $m \to \infty$
- This is in contrast to traditional models
  - Correlation structures of their aggregated processes degenerate, i.e. $r^{(m)}(k) \to 0$ as $m \to \infty, k = 1, 2, 3, \ldots$

4.9.3 Squared Coefficient of Variation

This is a critical factor in determining the type of Queuing System to use. It is a measure of variability of the system.

$$c^2 = \frac{\sigma^2}{\mu^2} \tag{4.37}$$

The squared coefficient of variation for some common Queuing Systems are shown below.

- $M/M/1 = 1$
- $M/D/1 = 0$
  - Namely, for any system with a deterministic distribution, the squared coefficient of variation will be 0.

5 Wireless Medium Access Control

We need to separate when devices are transmitting. Due to the nature of radio waves, we cannot transmit simultaneously on the same frequency from multiple devices.

5.1 Reservation Schemes

To begin our discussion of Reservation Schemes, we must talk about ALOHA.

Defn 98 (ALOHA)**.** *ALOHA* was one of the earliest protocols for wireless data transmission. It used a purely random access scheme, so when you had a packet to send, it was sent. To ensure devices knew if their transmission was successful, an acknowledgment was sent when the next recipient in the chain received the message.

Remark 98.1. It is only mentioned here because it is used as a baseline measurement for other protocols.

Defn 99 (Reservation Scheme)**.** A *reservation scheme* is the idea of reserving resources for a node in a wireless network to prevent collisions. The resources that can be reserved include: time, frequency, and others.

We would like to reserve resources because the system will not lose capacity to collisions. But, if not **all** the resources are being fully utilized, then the system loses out on throughput.

Because of the limitations of the radio spectrum and light being a broadcast medium, we have 4 main solutions for supporting multiple devices that was to transmit.

1. Space Division Multiple Access
2. Frequency Division Multiple Access
3. Time Division Multiple Access
4. Code Division Multiple Access

Approach	**SDMA**	**TDMA**	**FDMA**	**CDMA**
Idea	Segment space into cells/sectors	Segment sending time into mutually disjoint time-slots, demand-driven or pattern-driven.	Segment the frequency band into disjoint sub-bands.	Spread the spectrum according to orthogonal codes.
Terminals	One one terminal can be active in a single cell/sector.	All terminals active for short time periods on the same frequency.	Every terminal has its own frequency, uninterrupted.	All terminals can be active at the same time at the same place, on the same frequency.
Signal Separation	Cell structure, directed antennas.	Synchronization in time domain.	Filtering in frequency domain.	Code plus special receivers.
Advantages	Very simple, increases capacity per unit area.	Established, **fully digital**, flexible.	Simple, established, robust.	Flexible, less frequency planning needed, soft hand-over.
Disadvantages	Inflexible, antennas typically fixed.	Guard time needed (multi-path propagation), synchronization difficult.	Inflexible, frequencies are a scarce resource and must be decided from beginning.	Complex receivers, needs more complicated power management from senders.
Comments	Only useful in combination with TDMA, FDMA, or CDMA.	Standard in fixed networks. Used together with FDMA and SDMA in mobile networks.	Typically combined with TDMA (frequency hopping patterns) and SDMA (frequency reuse)	Has some problems still. Higher complexity, lower expectations. Needs integration with TDMA/FDMA.

Table 5.1: Division Multiple Access Comparison

5.1.1 SDMA

Defn 100 (Space Division Multiple Access)**.** In a *space division multiple access* (*SDMA*) setup, the space is segmented into sectors, then addressed with directed antennas. These tend to follow a distinct cell structure to handle varying number of people.

- Multiple users are separated by having separate physical spaces allocated to them.

+: This is very simple to implement

—: Relatively fixed setup, meaning it is not easy to move or change.

Example: This scheme is used in most cellular networks.

5.1.2 FDMA

Defn 101 (Frequency Division Multiple Access)**.** In a *frequency division multiple access* (*FDMA*) system, a certain frequency is assigned to a certain transmission channel between the sender and receiver.

Because of the way FDMA works, it effectively acts like a Circuit Switching system, so we can apply the same equations as Modeling Circuit Switching.

$$\eta = \frac{1}{nv} \sum_{i=1}^{N} \rho_i \tag{5.1}$$

where

- v: The achievable rate, in bits/Hz/s. This includes:
  - Guard bands
  - Signaling Overheads
- N: The number of nodes
- ρ: The utilized rate
- η: The rate of work

- Multiple users are given their own unique frequency.

+: Simple and robust system.

—: The number of possible frequencies are limited, so there may not be enough frequencies for all users.

Example:
- Permanent: Radio broadcasting
- Slow Hopping: GSM (2G)
- Fast Hopping: FHSS (Frequency Hopping Spread Spectrum)

5.1.2.1 OFDM

Defn 102 (Orthogonal Frequency Division Multiplexing)**.** *Orthogonal frequency division multiplexing* or *OFDM* is a way of breaking up the frequency spectrum into orthogonal frequencies into orthogonal subcarriers. An orthogonal subcarrier is a frequency that has a waveform that has a peak where all other subcarriers' waveform is zero. To achieve this, the equation below is used.

$$\Delta f = \frac{k}{T_U} \tag{5.2}$$

where

- $k \in \mathbb{N}$
- T_U is the useful symbol duration

Because these subcarriers are orthogonal, there do not need to be guard bands between them, allowing for more efficient usage of the transmission spectrum. Also, each symbol is spread over a wider frequency range of many slowly-modulated narrow-band signals instead of one rapidly-modulating wide-band signal. If this is paired with error detection and correction, the transmission is even more robust.

We can use Orthogonal Frequency Division Multiplexing to do 2 things:

1. Subcarrier Allocation
2. Transmission Power Control

5.1.2.2 Subcarrier Allocation If a single node does not need the entire frequency spectrum that we have available, we can can share the channel's subcarrier frequencies between multiple nodes. This flexibility means:

- We can assign different numbers of subcarriers to different nodes depending on the node's throughput requirements.
- We can adjust which subcarriers are allocated to account for adverse channel conditions.

5.1.2.3 Transmission Power Control In general, the greater the power transmission, the greater the data rate.

Remark. This does not hold true for Code Division Multiple Access systems. That small difference will be discussed there.

The flexibility that transmission power control gives us is:

- If a node does not require the highest possible data rate for the current SINR, it can reduce transmission power.
- SINR is lowered but the node changes to a different modulation and coding scheme to compensate.
- This reduces the data rate, so the transmission power should be chosen to match the required rate.
- Adjusting the nodes' transmission rate through transmission power creates a form of Space Division Multiple Access. Nodes will cause interference to other nodes in a larger area if the transmission power increases.

Data Rate (mbps)	Modulation	Coding Rate	Coded bits per subcarrier	Coded bits per OFDM symbol	Coded bits per OFDM symbol
6	BPSK	1/2	1	48	25
9	BPSK	3/4	1	48	36
12	QPSK	1/2	2	96	48
18	QPSK	3/4	2	96	72
24	16-QAM	1/2	4	192	96
36	16-QAM	3/4	4	192	144
48	64-QAM	2/3	6	288	192
54	64-QAM	3/4	6	288	216

Table 5.2: Transmission Power Control

5.1.2.4 Joint Subcarrier Allocation and Transmission Power Control We can perform both Subcarrier Allocation and Transmission Power Control at the same time.

This has the

- Advantages of
  - Lots of flexibility with ow we allocate resources.
- Disadvantages of
  - Being a difficult optimization problem, especially for real-time applications.

5.1.3 TDMA

Defn 103 (Time Division Multiple Access)**.** A *time division multiple access* (*TDMA*) system assigns a fixed sending frequency to a transmission channel **for a specific amount of time**.

Because of the way FDMA works, it effectively acts like a Circuit Switching system, so we can apply the same equations as Modeling Circuit Switching.

$$\eta = \frac{1}{nv} \sum_{i=1}^{N} \rho_i \tag{5.3}$$

where

- v: The achievable rate, in bits/Hz/s. This includes:
  - Guard bands
  - Signaling Overheads
- N: The number of nodes
- ρ: The utilized rate
- η: The rate of work

- Multiple users are only allowed to transmit in certain reserved timeslots.

+: This is a completely digital system, so it is easy to change and configure.

—: All nodes must keep their time synchronized, which is difficult. Multi-path propagation of the transmission wave also complicates things.

Example: Bluetooth.

5.1.4 CDMA

Defn 104 (Code Division Multiple Access)**.** A *code division multiple access* (*CDMA*) system works by having all terminals send on the same frequency, likely at the same time. These terminals can use the whole bandwidth of the transmission channel. To differentiate these terminals,

- Each sender has a unique random number, the sender XORs the signal with this pseudorandom number
- The receiver can "tune" into this signal if it knows the pseudorandom number, tuning is done via a correlation function

- Each user in the system gets an (pseudo-) orthogonal code, which is used to encode their data. The receiver can then decode the signal.

+: This is a flexible system, with less management on transmission frequencies required.

—: Complex receivers are required to implement this.

Example: UMTS

To handle multiple data streams inside the same user, there needs to be multiple codes that are used. These are the:

1. Spreading Code
2. Scrambling Code

Defn 105 (Spreading Code)**.** The *spreading code* is used at the end of each data stream, so we can figure out which application/data stream the data belongs to. It is only used inside the user's device, and it done before each of the data streams is added together, XORed with the Scrambling Code, and pushed out.

Defn 106 (Scrambling Code)**.** The *scrambling code* is used once all the applications/data streams have been XORed with their Spreading Codes and aggregated. This code is then XORed with this output and then sent out to be handled by the receiver.

There is a concept of the spreading factor.

$$\frac{R_c}{R_i} = \frac{\frac{1}{R_i}}{\frac{1}{R_c}} \tag{5.4}$$

5.1.4.1 Orthogonal Variable Spreading Factor Codes The codes used for Code Division Multiple Access are generated in a recursive manner. Each generation of codes halves the overall throughput of the system.

$$x_{i-1} \rightarrow \begin{cases} (x_i, x_i) \\ (x_i, -x_i) \end{cases} \tag{5.5}$$

	Down Link	Up Link
Scrambling Code	Identify cells. Code assigned to each cell. Only 512 codes. Assignment done by system designer.	Identify users. Code assigned to each user. 2^{24} codes. Assignment done by algorithm.
Spreading Code	Identify channels used by a cell. Code assigned to each user.	Identify channels to be used by users. Code assigned to each channel.

Table 5.3: How to use OVSF Codes

The codes are not perfectly orthogonal, meaning that some amount of the transmission's value can be viewed as noise (really it's lost information), so each node causes some interference. Because of the way this system works and interacts with devices near the receiver, there is a general performance degradation that closely follows the Gaussian Random Variable's distribution. Meaning, there is only a soft performance degradation as the system gains additional users.

5.2 Polling

Defn 107 (Polling)**.** *Polling* is the act of having one terminal that can be heard by all others ask for a small amount of information. This "central" terminal, e.g. a base station can poll all other terminals according to a certain scheme. This is a typical server-client workflow, where the server is the one polling all the clients regularly.

We are usually interested in the effective performance of the polling system. To calculate this, we can use Equation (5.6).

$$E = \frac{T_t}{T_t + T_{\text{Idle}} + T_{\text{Poll}}} \tag{5.6}$$

where

- T_t: Transmission time of the polling packet.
- T_{Idle}: Time waiting to detect no transmission.
- T_{Poll}: Time to send the polling message.

5.2.1 Randomly Addressed Polling

To perform randomly addressed polling:

1. Base station signals readiness to all mobile terminals
2. Terminals ready to send transmit a random number without collision with the help of Code Division Multiple Access or Frequency Division Multiple Access
    - The random number can be seen as dynamic address
3. Base station chooses one address for polling from list of all random numbers
    - Collision if two terminals choose the same address
4. Base station acknowledges correct packets and continues polling the next terminal
5. Cycle starts again after polling all terminals of the list

5.2.2 Point Coordination Function

Defn 108 (Point Coordination Function)**.** *Point Coordination Function* or *PCF* is a centralized Polling mechanism for WiFi (802.11). The central poller is the current Access Point (AP).

In this system, there are 2 types of tranmission periods. The first is during the Point Coordinated Function, when the resources on the AP are reserved for users. The second is a pure Random Access period. Here, the rules of a random access sytem are obeyed.

- Advantages
    - Resource management. Resources are only given to transmitters who request them.
- Disadvantages
    - Traffic Separation. There is no way to separate all the traffic flows.

5.2.2.1 Problems with the Point Coordination Function

The performance of Point Coordination Function can be quite poor, especially when used for voice communications. When a PCF system is running, and a poll occurs, the users that have nothing to send return null packets. While these packets are empty (contain no information), they are sitll packets that need to be transmitted, thus taking up tranmission time.

Additionally, there is no Quality of Service (QoS) signalling, which can further degrade performance.

There is also no tranmission time limit. If a user is polled, and it has something to transmit, the transmission will continue until the user is done. This incurs a cost, because other nodes may not have their chance to transmit during the Point Coordination Function window, and must wait for the Random Access window, or must wait until the next round of polling.

5.3 Random Access

Defn 109 (Random Access)**.** *Random Access* is the process by which a user can request resources from a system for a duration of time. After this time period has passed, the resources are freed from that user and may be passed to another. These are also called *Contention-Based protocols.*

This allows for sharing the same resources among many users, but there may be contention for the resources. In addition, because there is not way to arbitrate the delivery of data, there may be packet collisions when multiple transmitters send data, and the receiver cannot decode the data from just one of them.

5.3.1 Fundamentals of ALOHA

- Full duplex communication.
    - Up-link (Upload)
    - Down-link (Download)
- Originally UHF (Ultra-High Frequency) communication
- Incoming traffic (to base station):
    - 9,600 bps (403.500 MHz)
    - Random access (Pure ALOHA)
- Outgoing traffic (from base station):
    - 9,600 bps (413.474 MHz)
    - Broadcast

5.3.1.1 Pure ALOHA

- Whenever a station has something to send, it sends.
- When a packet is received by the central, an acknowledgment (`ACK`) is sent back in broadcast.
- If the sending station does not receive an `ACK` within a set time, a collision is assumed.
- When a collision occurs, retransmit within a random time slot, 200–1500 ms.

Remark. We wait a random amount of time before transmitting to prevent retransmission collisions. If 2 nodes were to experience a collision, and both waited a fixed amount of time, then they would collide a again during their retransmission.

5.3.1.2 Slotted ALOHA

- Packets may just be transmitted within time slots.
- If a station has started to transmit in a time slot, other station who wish to transmit within this time slot can not interfere.
- This principle leads to a much better utilization of the channel.

The utilization gets so much better because there is **no chance** of a collision. Since each station can only transmit if no other station is transmitting in this time slot, there cannot be collisions part-way through a packet's transmission.

5.3.1.3 ALOHA Performance Packet generation process is Poisson distributed with n packets generated during time t:

$$\mathrm{P}(n,t) = \frac{(\lambda_p t)^n e^{(-\lambda_p t)}}{n!}$$

The probability zero packets generated during time t is

$$\mathrm{P}(n=0,t) = \frac{(\lambda_p t)^0 e^{-\lambda_p t}}{0!} = e^{-\lambda_p t}$$

The Effective throughput is equal to the Rate multiplied by the probability of success.

$$T_{\text{Put}} = \lambda_p T_p e^{(-\lambda_p T_p)}$$

The maximum for Pure ALOHA is found at:

$$\frac{\partial}{\partial x} T_{\text{Put}} = 0 \Rightarrow \frac{1}{2e}$$

In Slotted ALOHA, $t = T_{\mathrm{P}}$, so its maximum is found at:

$$\frac{\partial}{\partial x} T_{\text{Put}} = 0 \Rightarrow \frac{1}{e}$$

5.3.2 Carrier Sense Multiple Access

Defn 110 (Carrier Sense Multiple Access)**.** In *Carrier Sense Multiple Access* (*CSMA*), a channel can only transmit its message if the underlying network connection does not have an ongoing transmission. This is sometimes called the polite method or Listen Before Talk network communication because of the Carrier Sense that must occur.

- If the channel is free, it starts to send.
- If the device that wants to send senses the channel is busy, it waits to send.
  - The station might be persistent, meaning it might continue to try sensing the network.
- Acknowledgement

Defn 111 (Persistence)**.** *Persistence* is a property of Carrier Sense Multiple Access systems. Systems that use persistence are known as persistent systems.

Remark. It might be helpful to think of persistence as the device's persistence to listen to the channel.

There are 3 types of persistent systems.

1. Non-Persistent
2. p-Persistent
3. Persistent

Defn 112 (Non-Persistent)**.** A *Non-Persistent* system behaves as follows:

1. If the channel is busy, the device will wait a random time before performing another carrier sense on the channel.

2. Once the channel is idle, the packet is sent immediately.

The performance of a non-persistent system is:

$$S_{\text{Non}} = \frac{G}{1+G} \tag{5.7}$$

where

- G: Offered Load
- S: Throughput

Defn 113 (p-Persistent)**.** A *p-Persistent* system behaves as follows:

1. If the channel is busy, the device will continue to perform carrier sense on the channel.
2. Once the channel is idle, there is a probability p that the device will send.
3. There is a $1-p$ probability that the deivce will instead wait one time unit before performing carrier sense again.

The performance of a p-persistent system is:

$$S_p = \frac{Ge^{-G}(1+pG)x}{G+e^{-G}} \tag{5.8}$$

where

- G: Offered Load
- S: Throughput
- x: Is defined by Equation (5.9)

$$x = \sum_{k=0}^{\infty} \frac{(qG)^k}{(1-q^{k+1})!} \tag{5.9}$$

Defn 114 (Persistent)**.** A *Persistent* system behaves as follows:

1. If the channel is busy, the device will continue to use carrier sense on the channel.
2. Once the channel is idle, this device will start sending its packet immediately.

5.3.3 Carrier Sense Multiple Access/Collision Detection

Defn 115 (Carrier Sense Multiple Access/Collision Detection)**.** *Carrier Sense Multiple Access/Collision Detection* is a modification of Carrier Sense Multiple Access. It functions in much the same way, but:

1. Perform Carrier Sensing
   - If the channel is free, send.
2. The station listens when sending
   - When a station detects a collision, it stops sending
3. Retransmissions

If we use a simplistic model:

- All stations are the same
- No exponential backoff of tranmissions
- All packets are the same size
- All stations have the same transmission probability
- k stations
- p Probability of transmission
- A Probability 1 station acquires channel in slot

$$A(k,p) = kp(1-p)^{k-1} \tag{5.10}$$

We can find the maximum performance of the system, which occurs when $p = \frac{1}{k}$.

$$A_{\text{Max}} = \left(1 - \frac{1}{k}\right)^{k-1} \tag{5.11}$$

If we allow the number of stations to grow towards infinity, then

$$\lim_{k\to\infty} A(k,p) = \frac{1}{e} \tag{5.12}$$

If we want to know the number of repeated collisions that occur, then a successful transmission, we can use the derivation below. We define w to be the number of slots i in a row with collisions or an idle channel, with a subsequent successful tranmission.

$$\begin{aligned}\mathbb{E}[w] &= \sum_{i=0}^{\infty} i\,\mathrm{P}[w] \\ &= \sum_{i=0}^{\infty} i(1-A)^i A \\ &= \frac{1-A}{A} \text{ Geometric Series}\end{aligned}$$

Now if we want to find the utilization of the system, U, we apply the equation below.

$$U = \frac{T_{\text{Trans}}}{T_{\text{Trans}} + 2T_{\text{Prop}} + T_{\text{Prop}}\left(\frac{1-A}{A}\right)} \tag{5.13}$$

This is interpreted as transmission time divided by the sum of the transmission time, the total propagation delay, and the contention time. If we define $a = \frac{T_{\text{Prop}}}{T_{\text{Trans}}}$, the the normalized utilization is as shown in Equation (5.14).

$$U = \frac{1}{1 + 2a + a\left(\frac{1-A}{A}\right)} \tag{5.14}$$

Remark (Utilization vs. Propagation Delay). The utilization of a Carrier Sense Multiple Access/Collision Detection system falls as the propagation delay increases, because there is a greater delay for any single node to be able to see that the network is busy. If the propagation delay increases past a certain point, then the system is waiting for so long that it can never find that someone else is transmitting.

5.3.4 WiFi

Defn 116 (WiFi). *WiFi* is defined by the IEEE 802.11 family of protocols. Many different protocols (with different letters) exist, all of which have different purposes. The current general-purpose version is 802.11ac.

The 802.11 specification allows different MAC protocols to be used.

- Distributed Coordination Function (DCF)
  - Carrier Sense Multiple Access-based random access.
- Point Coordination Function (PCF)
  - Random access periods followed by non-contention periods with polling.
  - Rarely used outside of industry.
- Enhanced Distributed Channel Access (EDCA)
  - Distributed Coordination Function with some QoS features added.

Defn 117 (Distributed Coordination Function). The *Distributed Coordination Function* (*DCF*) is similar to a p-Persistent Carrier Sense Multiple Access system, but with a different backoff method.

- Each node has a contention window (CW).
- When the node detects channel busy, it chooses a (uniform) random number of slots to wait in the range $[1, \text{CW}]$.
- CW starts at a minimum value CW_{Min}.
- When a collision is detected, CW doubles (binary exponential back-off) up to a maximum value of CW_{Max}.
- Whenever a successful transmission occurs, the receivers sends an acknowledgement.
- Lack of acknowledgement indicates a collision.

To help the rest of the DCF system run, there are 2 temporal buffers that we used, called Inter-Frame Spacing.

1. Data Inter-Frame Spacing
2. Short Inter-Frame Spacing

Defn 118 (Inter-Frame Spacing)**.** *Inter-Frame Spacing* is just a buffer in time. It helps separate different types of tranmissions in a time period in a Distributed Coordination Function system.

There are 2 different kinds of inter-frame spacing:

1. Data Inter-Frame Spacing
2. Short Inter-Frame Spacing

Defn 119 (Data Inter-Frame Spacing)**.** *Data Inter-Frame Spacing* (*DIFS*) is the amount of time a node waits, after detecting the channel is free, before sending a new frame.

$$\text{DIFS} = \text{SIFS} + (2 * \text{slot time}) \tag{5.15}$$

Remark 119.1 (Amount of DIFS Time). The Data Inter-Frame Spacing must be greater than the propagation delay.

Defn 120 (Short Inter-Frame Spacing)**.** *Short Inter-Frame Spacing* (*SIFS*) is the amount of time a receiver must wait after detecting the channel is free before it can send an acknowledgement.

SIFS must be less than Data Inter-Frame Spacing. This means acknowledgements get priority over data.

Remark 120.1 (Amount of SIFS Time). The Short Inter-Frame Spacing must be greater than the propagation delay.

5.3.5 Carrier Sense Multiple Access/Collision Avoidance

Defn 121 (Carrier Sense Multiple Access/Collision Avoidance)**.** In *Carrier Sense Multiple Access/Collision Avoidance* (*CSMA/CA*), the sender would send as soon as the network medium is free. The sender would then listen into the network medium if a collision occurred.

However, there are some problems with CSMA/CA that occur because of the nature of wireless radio networks.

- Signal strength decreases proportional to the square of the distance (from electromagnetics $\frac{1}{r^2}$).
- The sender would apply Carrier Sense and Carrier Sense Multiple Access/Collision Detection, but the collisions happen at the receiver.
- It might be the case that a sender cannot "hear" the collision, i.e., Carrier Sense Multiple Access/Collision Detection does not work.
- Furthermore, Carrier Sense might not work if, for example, a terminal is "hidden" (Hidden Node).

Defn 122 (Hidden Node)**.** A *hidden node* is one that is out of the range of one transmitter, but is transmitting. Then, any node in between them will receive both signals and a collision will occur.

The steps to recreate this are as follows:

1. Suppose ther are 3 transmitters in a line, in this order A, B, C (from left-to-right).
2. C is already transmitting to B.
3. A wants to transmit to B.
   (a) A listens
   (b) But can't hear C. A is too far away.
4. A transmits at the same time as C.
5. B can't decode the data, leading to a collision.
6. Thus, C is hidden from A.

Defn 123 (Exposed Node)**.** An *exposed node* is one that is in range of a currently transmitting transmitter, but the desired node that would receive the tranmission is out-of-range of this transmission.

The steps to recreate this are as follows:

1. Suppose there are 4 transmitters in line, in this order D, A, B, C (from left-to-right).
2. A wants to transmit to D.
3. A listens and detects that B is already transmitting to C.
4. A waits, even though C is too far away to overhear. This means there is no risk of a collision.
5. A is exposed to C.

5.3.6 Multiple Access with Collision Avoidance

Defn 124 (Multiple Access with Collision Avoidance)**.** *Multiple Access with Collision Avoidance* (*MACA*) is a method of collision avoidance that uses short signals/packets to arbitrate transmissions and avoid collisions. There are 2 packets,

1. Request To Send (RTS), which is sent by the sender. It requests the right to send a packet and include some additional information, such as how long the transmission will be, the address, etc.

2. Clear to Send (CTS), which is sent by the receiver. It responds to the sender that they are allowed to send, and also includes some additional information, such as the time the transmission will take.

Multiple Access with Collision Avoidance manages to avoid both the Hidden Node and Exposed Node problems at the same time.

- For both examples below, consider 3 nodes in a line, named A, B, C from left-to-right.
- Hidden Node
    1. A and C want to send to B.
    2. A sends RTS first.
    3. C waits after receiving CTS from B.
- Exposed Node
    1. B wants to send to A, C to another terminal.
    2. Now C does not have to wait for it cannot receive CTS from A.

5.3.6.1 Problems with Multiple Access with Collision Avoidance

It seems like Multiple Access with Collision Avoidance could solve all our problems, but there are still some pitfalls.

- RTS/CTS does not guarantee success
    - RTS/CTS packets may be lost due to contention
- The traffic composition may still yield poor results
- External interference may also cause problems

5.3.7 802.11e

This is a second attempt at delivery of Quality of Service in the 802.11 family after Point Coordination Function To do so, we created the Hybrid Coordination Function and Enhanced Distributed Channel Access. These should just be viewed as new MAC schemes for the 802.11 family.

Defn 125 (Hybrid Coordination Function)**.** The *Hybrid Coordination Function* (*HCF*) introduces 2 new concepts that are used in conjunction.

1. Transmission Opportunity
2. Traffic Specification

Defn 126 (Transmission Opportunity)**.** The *Transmission Opportunity* (*TXOP*) is a bounded time interval during which a station may transmit multiple frames.

This solves the Point Coordination Function problem of packets with unknown transmission times.

Defn 127 (Traffic Specification)**.** The *Traffic Specification* (*TSPEC*) contains information about the Quality of Service expectation of a traffic stream (frame size, service interval, data rate, burst size, delay bound, etc.).

This solves the solves the Point Coordination Function problem with the inability to send QoS needs.

Defn 128 (Enhanced Distributed Channel Access)**.** The *Enhanced Distributed Channel Access* (*EDCA*) scheme is very different than many others. It can be thought of as a contention-based "enhanced Distributed Coordination Function".

There are *access categories* (*ACs*), which can be thought of as tranmission queues. Each station has 4 of these access categories. Each AC contends for Transmission Opportunity independently of the other ACs.

Thus, we can provide service differentiation, by varying:

- $\text{CW}_{\text{Min}}[AC]$
- $\text{CW}_{\text{Max}}[AC]$
- $\text{AIFSN}[AC]$
- $\text{TXOP}_{\text{Limit}}[AC]$

However, this means there is no deterministic behavior, only statistical.

Defn 129 (HCF Controlled Channel Access)**.** *HCF Controlled Channel Access* (*HCCA*) is akin to Point Coordination Function contention free but extended to provide traffic separation whereby the AP hands out a TXOP with different sizes to different flows using Traffic specifications. The scheme still suffers from the overheads associated with the PCF.

5.4 Demand Assigned Multiple Access

The use of time reservation can increase the efficiency of a system up to 80%. To do this,

- A sender **reserves** a future timeslot.
- Sending within this timeslot is possible without any collision.
- However, the act of having to reserve something increases the delay.
- This is typically used in satellite links.

There are 2 reservation algorithms for requesting a timeslot:

1. Explicit Reservation
2. Reservation Time

5.4.1 Explicit Reservation

This scheme is similar to the way ALOHA works. Each node requests a time slot on-the-fly. There are 2 main implementations:

1. ALOHA mode
   - This means that time slots are assigned as they are needed.
   - However, if multiple nodes was the same timeslot, there may be a competition.
   - In addition, multiple timeslots can be reserved at once.
   - The controlling system will grant access to anyone that asks for it, without question.
   - However, there may be collision/overlapping timeslot between 2 nodes.
2. Reserved mode
   - Data transmission can only happen inside of successfully reserved slots.
   - The controlling system will schedule and grant access to these based on some algorithm.
   - This means that there can be no collisions between timeslots.

5.4.2 Reservation Time

In this scheme, each major time interval is a frame. Inside this frame, there are N mini-slots and x data-slots, where N is the number of nodes in the network. Each station gets its own mini-slot, and **must reserve** up to k data slots, so $x \geq Nk$.

If there are unused data-slots, then they can be assigned to other stations that require the additional bandwidth.

5.5 Energy-Efficient MACs

In some systems, we need to make sure the power usage is minimized. The biggest energy costs in a network are:

- Sending Data
- Receiving Data
- Listening to the channel
- Keeping the node's networking system on when it doesn't need to transmit or receive data.

Remark. The costs of both the transmission and reception of data are hard to minimize on a MAC level. These would usually be minimized by the design of hardware on the physical level and determining if it is worth it to send information at all in the application layer.

Remark. Collisions have a massive effect on the energy usage of a networking system. A collision means the receiver did not receive the packet correctly, so it must be retransmitted. Meaning, 2 transmissions and 2 receptions were needed to correctly read the packet, effectively double the correct amount of power.

Remark. Carrier Sense Multiple Access is **not** a power-efficient MAC protocol, because it constantly listens to the channel. However, it can be improved by reducing the Persistence of the system.

5.5.1 The LoRa Protocol

Defn 130 (LoRa)**.** *LoRa* is a protocol suite designed for long range, low power communications.

It uses a Star-of-Stars network topology. The end-point devices (usually Internet of Things devices) generate the data. In between the end-point devices, there is the "fog" which is some amount of network gateways and managers. At the end "star" is the Cloud, where the appropriate company collects their data.

In LoRa, there are 3 different MACs:

1. Class A: The baseline MAC
2. Class B: The beacon MAC
3. Class C: The continuous MAC

5.5.1.1 Reception in the LoRa Protocol

In a Class A LoRa system, the end device is **the only one** that can intiate tranmission. After a transmission, the system goes back to sleep, becoming unreachable by the network. There are 2 reception windows during which the end-point device can receive data. If the unit receives its data during the first reception period, then it skips the second one.

5.5.1.2 Why is LoRa Energy-Efficient?

- Energy efficient physical layer.
- End device can sleep except for exactly when it needs to transmit or receive.
- Listening is kept to a minimum. It is only done during the 2 receive windows.
- Simple protocol with low overhead.
  - Contention between devices resolved using Pure ALOHA. When you have a packet, just send it.
  - MAC headers are small, at a minimum of 13 bytes.

6 Network Architecture

Defn 131 (Network Architecture)**.** *Network Architecture* is:

- How the overall system is designed: what components there are and how they interact.
- Protocols and technologies for all the layers of the network stack.
- Physical infrastructure: base stations, cables, antennas, routers, etc.
- How everything works together to meet the overall performance goals for the whole system.

6.1 Wireless vs. Wired Networks

Wireless and wired (fixed) networks operate under the same underlying network principles. However, wireless systems have a very different set of challenges to also contend with, because of the nature of wireless radio communication, and it being a broadcast medium.

The biggest problems for wireless networks that wired networks do not have are:

- Interference, Section 6.2
- Mobility, Section 6.3
- Wireless channels:
  - Path loss
  - Shadowing
  - Fading
  - These will not be handled in this course.

6.2 Interference in Wireless Networks

The biggest difference between wired and wireless networks when it comes to interference is because of the transportation medium. In a wired system, discrete connections form isolated links between nodes. The properties of any single link do not directly affect the properties of any other node.

However, in wireless networks, each of the nodes is broadcasting into space. Any of these nodes' broadcasts can affect any of the other nodes. You can think of it as if each node has a large sphere extending radially outwards, where multiple nodes can cover the same zone. Additionally, this may also mean that some nodes are dependent on other nodes.

6.2.1 Performance in Interfering Networks

In the work done by Gupta and Kumar, it was found there is a closed-form solution for the performance of interfering wireless networks.

Place n nodes on a unit disk. Each node as the capacity W bits per second. Then, the upper bound on the capacity for the network as $n \to \infty$ is

$$\frac{W}{\sqrt{n \log(n)}} \tag{6.1}$$

If the nodes are placed optimally, i.e. each node is placed such that its transmission range never crosses another node's range. Then the capacity of the network is given in Equation (6.2)

$$\frac{W}{\sqrt{n}} \tag{6.2}$$

6.3 Mobility

Defn 132 (Mobility)**.** *Mobility* is a unique property of wireless networks. Since the nodes are not fixed in place by anything, they are free to roam/move. This is a big problem, because nodes can enter, leave, and move freely throughout/through the network.

There are 3 classifications of mobility.

1. Pedestrian: $\approx 1.5\text{m/s}$
2. Vehicle: $5 - 33\text{m/second}$
3. High-Speed: $> 70\text{m/s}$

Mobility has several effects on a wireless network system.

1. Physical Layer
   - Doppler shift
   - Constantly changing channel $\rightarrow$ higher bit-error rate
2. Data-Link Layer/MAC
   - Which nodes share "links" (and thus can experience collisions) constantly changes.
   - Hidden Node and Exposed Node problems!
3. Network Layer
   - Constantly changing topology $\rightarrow$ need to constantly change end-to-end path through the network.
4. Transport/Application Layers
   - Not directly affected.
   - Delays and errors at lower layers can cause problems though.

6.4 Licensed vs. Unlicensed Spectrum

There are 2 types of frequencies in use by people and companies today.

1. Licensed Spectrum
2. Unlicensed Spectrum

6.4.1 Licensed Spectrum

The licensed spectrum requires that users obtain exclusive rights to use that frequency spectrum.

- Solution for QoS-sensitive applications
- Exclusive right to spectrum use
- Network engineering possible
  - Predictability
  - Manageability
- Complex and costly systems are generally built around these.
- Typically big players take these.

6.4.2 Unlicensed Spectrum

The unlicensed spectrum does not require that users obtain exclusive rights, but they may require that only certain types of data can be used in that frequency.

- Inherent best effort systems
- "Some" QoS support possible
- No right to use spectrum
  - Competition
  - Collaboration
- Simple and cheap systems
- For small/medium players

6.5 Data Rate vs. Range

Higher data rates means lower tranmission range. Higher data rates also means greater amounts of power required to transmit the message.

6.6 Centralized Infrastructure Networks

Defn 133 (Centralized Infrastructure Network)**.** *Centralized Infrastructure Networks* have some fixed nodes which have the connection to the wider Internet. These gateways are the ones that provide access to the Internet. This system works in a hierarchical structure, following a Star-of-Stars topology.

Defn 134 (Star-of-Stars)**.** A network with a *star-of-stars* topology is a hierarchical setup, where the leaf nodes in the system are the ones generating data to send. The next rows up may generate some data, but mainly serve as aggregators for the lower nodes in the system. Eventually, all connections terminate at one head star node.

6.6.1 Example Centralized Architectures

1. 802.11 a/b/g/n/ac
   - Most common WiFi network type, used in homes, shops, offices, etc.
   - 2.4 or 5 GHz bands, range $\approx$ 500m
   - MAC: 802.11 DCF
2. 802.11 ah
   - WiFi for Internet of Things
   - 800 MHz band, range $\approx$ 1km
   - MAC: 802.11 DCF with some modifications:
     - Relay nodes
     - Target wake times
     - Contention groups
   - Aimed at reducing energy usage and providing coverage to more nodes in a single network
3. LoRa
   - Low-Power Wide Area Network (LPWAN) for Internet of Things
   - 800 MHz band, range 10s of km
   - MAC: ALOHA for the uplink, with receive windows for the downlink

Remark. For the above architectures, standard Internet routing (Distance Vector Routing Protocol or Link State Routing Protocol) is used.

- Only one wireless hop, so no specialised routing is needed.

Remark. Some wireless-aware protocols can be used at higher layers, e.g. Mobile IP, Wireless TCP (WTCP).

6.7 Infrastructure Mesh Networks

Defn 135 (Wireless Mesh Network)**.** A *wireless mesh network* (*WMN*) is a network where there are multiple wireless hops between nodeswho are all peers, meaning there is little to no hierarchy. There may be some hierarchy, if only some nodes are the ones with the direct Internet connection.

These architectures use many nodes who communicate with each other to move data from anywhere in the network to a select number of nodes that contain the actual Internet connection. This may require several hops until we reach a node with a connection. Throughout this process, we do not have to worry about nodes moving, because they are fixed. However, we still need to maintain the routing information, so the packets are sent back to the correct node.

6.7.1 Example Infrastructure Mesh Architectures

1. 802.11s
   - Backbone of static nodes, with mobile nodes connecting to the nearest static node.
   - Some or all static nodes typically act as Internet gateways.
   - Example use case: metropolitan network
   - Builds on top of 802.11 a, b, g, n, or ac: same MAC and physical layers
   - A routing protocol tailored to wireless mesh networks is added on top
     - Default protocol is Hybrid Wireless Mesh Protocol (HWMP)

6.8 Ad-Hoc Networks

Defn 136 (Ad-Hoc Network)**.** An *ad-hoc network* is a network that does not rely on **ANY** fixed infrastructure. Many ad-hoc networks tend to have similar properties.

(i) Often unplanned: They need to be

1. Self-configuring
2. Self-organizing
3. Self-healing

(ii) The network topology can be highly dynamic because of high Mobility or unreliable links.
(iii) The network usually has a flat architecture, where most nodes are peers, with few if any gateways/base stations.
(iv) The most important performance goal is **low-energy usage**.

This also means that these networks tend to be highly delay-tolerant, with next to no requirements on latency, or data transmission rate.

6.8.1 Uses of Ad-Hoc Networks

- Disaster recovery, rescue services
- Developing areas where there is a lack of infrastructure
- Personal area networks (e.g. Bluetooth)
- Wireless sensor networks: monitoring and data gathering, especially in inaccessible or remote areas
- Vehicular Ad-hoc Networks (VANETs): where vehicles communicate wirelessly

6.8.2 Example Ad-Hoc Architectures

1. 802.11p
   - WiFi for vehicular networks
   - 5 Ghz band, range ˜ 500 m
   - MAC: 802.11 DCF
   - Ad-hoc mode: no access points, no association
   - Beacons sent every 100 ms with vehicle location, speed, etc.
2. Z-Wave
   - Mainly used for smart homes
   - 800 MHz band, range: ˜ 100 m
   - MAC: CSMA/CA

Remark. If multiple hops are required for a transmission, there need to be specialized routing protocols. Because nodes may join, or leave the network at any given time. Meaning, the network's topology is never truly stable, i.e. you can never rely on any node being present at any given point in time.

6.8.3 Ad-Hoc Energy Usage

For most Ad-Hoc Networks, low energy usage is the **most** important performance requirement. Meaning, these networks have few other requirements. For example, the system may be allowed to have very large delays, so long as the data eventually reaches the end-point.

Some of the things that consume the most energy in a wireless transmission system are:

1. Using the radio: both sending and listening
   - CSMA costs extra energy because we must listen to the channel
   - Collisions and other packet losses cost energy because we must re-transmit
   - Protocol overheads:
     - Synchronisation (e.g. for TDMA)
     - Large packet headers, etc. cost energy because we need to transmit and receive information.
2. Keeping the device on: if we can sleep, we can save a lot of energy
3. Some types of processing, e.g. encryption is often costly — but not as much as radio transmission

Remark. It is also important to note that the energy usage is uneven between all the nodes. The nodes places along more frequently used routes will need to transmit more that ones on less-frequently used routes. So, the routes may have to be reconsidered over time.

6.9 Mesh Network Routing

Remark. Most of these can be applied to both Wireless Mesh Networks and Ad-Hoc Networks. However, some of the details relating to the problems due to Mobility will **NOT** apply to Wireless Mesh Networks.

- Mobility causes paths to be unstable.
- Using "traditional" shortest path algorithms is inefficient.
  - Dijkstra
  - Bellman-Ford
  - etc.
- Two main approaches:
  1. Proactive: Continuously update about topology and changes
     - Advantage: Low latency
     - Disadvantage: Costly updates
  2. Reactive: Only determine routing path when sending data
     - Advantage: No constant overhead
     - Disadvantage: Delay due to route discovery
- Some hybrid protocols exist as well, which combine proactive and reactive approaches in different ways.

Because shortest-path routing algorithms are not usable in Ad-Hoc Networks, what can we use? We can route based on:

- Interference
- Social Similarity
- Location
- Residual Energy
- Energy Usage
  - Multiple hops using little power vs. single high power transmission.

6.9.1 Ad-Hoc On-Demand Distance Vector Routing

Defn 137 (Ad-Hoc On-Demand Distance Vector Routing)**.** In *Ad-hoc On-demand Distance Vector routing* (*ADOV*), the sender broadcasts a "Route Request" packet, which is flooded throughout the whole network, until it reaches the intended receiver node. Each node maintains a route cache, which uses the node sequence the route request packet has gone through so far. Once the end node has been reached, the correct route state in each node is used to construct the link route.

This only reacts when routes fail. When they do fail, the nodes go through a 2-step process.

1. Attempt local recovery.
   - Nodes monitor neighbors
   - Issue error message on detected problem.
2. If local recovery fails, the original sender repeats the route discovery process by sending another "Route Request" packet.

The cost of this routing protocol is directly related to the rate of Mobility in the network. It realy only works for smaller networks with limited Mobility.

It is currently popular in deployed systems.

6.9.2 Geographic Routing

Defn 138 (Geographic Routing)**.** *Geographics Routing* is based on the nodes' **physical positions**, rather than network addresses or routing tables. Once the physically shortest route has been found, messages are then routed towards a destination location.

However, Geographic Routing has some issues as well.

- Route discovery is costly in terms of time and energy.
- Routing tables quickly become out of date in highly mobile networks.
- Addressing a message to a position may be more useful than to a network address.
  - Sending an accident or traffic jam notification to upstream vehicles in a VANET
  - Sending a request to collect sensor data in a particular region in a WSN
  - Tracking something through a WSN (e.g. an animal)

To make Geographic Routing less costly, we use Localization.

Defn 139 (Localization)**.** In *localization*, we find out where each of the nodes are in physical space. We can find the absolute locality with GPS, IMU, and many others. If we can use the relative locality, then we can use trilateration. This is done by having each node broadcast a signal and having each node listen for the same kind of signal. This signal is used to locate other beacons by using TDOA.

6.9.3 Greedy Forwarding

Defn 140 (Greedy Forwarding)**.** In *greedy forwarding*, we forward this packet to the node **physically closest** to the destination node. This guaranteed there will be no loops in the traversal of the network graph.

There are various forms and definitions of what "closest" means:

- Distance to target
- Distance along the source-destination line
- Smallest angle to destination (compass routing)

Remark 140.1 (Greedy Forwarding Failure). Greedy Forwarding fails when we we hit a local minimum. This means that the current node has **no** neighbors that are closer to the destination that this node.

However, this can be solved with Face Routing.

6.9.4 Face Routing

Defn 141 (Face Routing)**.** In *face routing*, the route is passed along the faces of a polygon in a set direction. The connection links that make the polygon's faces are based off which nodes are along the line between the source and destination nodes.

The message we are considering is sent around the faces of the polygons using Greedy Forwarding until it either reaches the destination, or you would be required to cross the source-destination line. If you have to cross the source-destination line, then you change to the next face.

Remark 141.1 (Planar Graphs). Face Routing **requires** that the graph is planar, i.e. there are no crossing links. Planarizing the graph can be done by removing crossing edges, using algorithms like Delauney Triangulation, however, this leads to an increase in hop counts. This makes the sending of the packet less efficient in terms of delay and energy usage.

Remark 141.2 (Localization Errors). Face Routing is sensitive to Localization errors, since it relies on geometric information to choose the next node to forward to.

We can use BOTH Greedy Forwarding and Face Routing!!

- Use Greedy Forwarding until we reach a dead-end.
- Then change to Face Routing to get around the obstacle.
- Once we are closer to the destination than the dead-end node, we can resume Greedy Forwarding.
- While using Greedy Forwarding, we can employ all links don't need a planar graph.

6.9.5 Contention-Based Greedy Forwarding

With Greedy Forwarding, we want to select the forwarding node closest to the destination.

One way is to keep track of neighbours' locations, but this increases overhead and breaks down when we have high mobility.

Instead, the source node can send "Request To Send" (RTS), and potential forwarding nodes respond with "Clear To Send" (CTS) at a time determined by their distance to the destination.

However:

- Using this method can increase delay, because you must wait for the timers to expire.
- We still need to know neighbours' locations to do Face Routing when recovering from a dead end.
- Can request locations from neighbours when required.

6.10 Realistic Network Models

The unit disk that we have been using to approximate the radio transmission of a node is not a good model of the transmission geometry of the device.

- The transmission radius is not uniform

- Three regions:
  1. Good Signal
  2. Degrading Signal (which falls off at a logarithmic rate).
  3. No Signal
- In the intermediate region (between good and none), we have a link, but it is unreliable.
- Using Greedy Forwarding, we will select a forwarding node that is far from previous node
  - Likely to land in the region with poor signal
  - This can lead to increased retransmissions
- We need to modify our greedy metric to take into account the cost of retransmissions, and balance this against increased hop count as results if we choose a closer node.
- Nodes can record the number of successfully received packets S and the total number of transmitted packets T on a given link.
- Packet reception rate (PRR):

$$\text{PRR} = \frac{S}{T} \tag{6.3}$$

  - S: Number of successfully received packets.
  - T: The total number of transmitted packets.
  - PRR: The Packet Reception Rate.
- The PRR can serve as a reception probability for future transmissions.
- We can then combine this with the distance gained to select a forwarding node,
  - For example, use PRR $\times$ distance as forwarding node selection metric, instead of only distance.

6.11 Internet of Things

Defn 142 (Internet of Things)**.** The *Internet of Things* (*IoT*) is a type of network where there are small, low-power sensors/devices placed throughout an environment to monitor the condition of the system. Some of these devices may also cause a change in the environment as they have been programmed to do.

The Internet of Things has a hierarchical structure, because of the computing power of the devices. Since the actual computational power of any single device in the network is nearly non-existent, the end devices send their data up to the "fog". This is the place where some data is aggregated, and shared among other devices in the fog for the same user. From the fog, it goes up to data centers that form the Cloud.

- **Cloud:** Large data centres with resources for computation and storage. Typically hosts computation-heavy functions, databases, user interfaces.
- **Fog Nodes:** Can be cellular base stations, local data centres, or local servers. Capable of computation and storage, but not as much capacity as in the Cloud. Used to host latency-sensitive parts of applications.
- **End Devices:** Sensors and actuators. Can be connected to Fog Nodes and backbone network with a range of radio technologies (e.g. cellular, LoRa, WiFi, Bluetooth) or wired connections. These can be routed in networks with a single hop (Star-of-Stars) or mesh topology.

- Physical objects embedded with sensors, actuators processing power, and network connectivity.
- "Smart objects" collect and exchange data with each other.
  - Machine-to-machine communication (M2M)
- Uses a range of different, interacting technologies
  - WSNs, VANETs, WiFi, LoRa, Bluetooth, ZigBee, RFID, ...
- Applications in:
  - Healthcare

  - Energy Management
  - Transportation
  - Building Automation
  - Industrial Processes
  - ... everything!

6.11.1 IoT Performance

The Internet of Things is different from previous networks because:

- Small, resource-constrained, battery-powered devices
  - Protocols need to be simple for the end device.
  - Energy efficiency is typically the most important performance metric.
- Traffic models for IoT: Event-Driven or Periodic.
  - For periodic traffic, scheduled access is usually the most appropriate, e.g. TDMA
  - For event-driven traffic, random access may be more appropriate, e.g. ALOHA
- Many devices in the same network
  - If the number of devices is too high, MAC performance starts to degrade.
  - With many devices, protocol overhead can become significant.
  - Managing, configuring, and monitoring many devices is difficult.

6.12 Cellular Networks

Cellular networks are setup in the Licensed Spectrums. They are the major backbone of data infrastructure that is not completely tied to a physical location.

6.12.1 GSM (2G)

To reuse the frequencies in GSM, we can subdivide an area into cells. These cells are idealized to hexagons, and each hexagon gets its own frequency. Then, any hexagons in adjacent zones cannot have the same frequencies near its border.

The repeating distance of these cells is given in Equation (6.4).

$$D = R\sqrt{3K} \tag{6.4}$$

- R: Cell radius (How large are the hexagons?)
- K: Cluster size (How many hexagons are allowed?)
- D: The repeating distance (How far between hexagons that repeat the same frequency?)

6.12.2 UMTS (3G)

Used Code Division Multiple Access

6.12.3 LTE (4G)

Defn 143 (LTE). *LTE* or *Long Term Evolution* is a standard that was finalized in 2008, and first publicaly available in 2009. Its main goals were to increase speed and capacity, while using a simpler IP-based network architecture. This was dirven by the large increase in data usage compared to phone calls made.

The LTE radio interface was incompatible with previous 2G and 3G networks.

The biggest feature was the concept of **Bearers**. This allows for different Quality of Services for different classes of network traffic.

6.12.3.1 Bearers

Defn 144 (Bearer)**.** *Bearer*s are a way to ensure Quality of Service.

- Minimum Guaranteed Bitrate (GBR)
  - Dedicated resources permanently allocated at bearer establishment
  - Higher bitrates may be allowed when resources are available
- Non-GBR, e.g. FTP, web browsing
  - Best effort service
  - No resources allocated
  - QoS class identifier (QCI): priority, packet delay budget, acceptable packet loss rate

6.12.4 5G

6.12.4.1 Software Defined Networking

6.12.4.2 Network Function Virtualization

6.12.4.3 Network Slicing

6.12.4.4 Cloud RAN

6.12.4.5 Frame Structure

6.12.4.6 Enabling Technologies for 5G

Millimeter Wave

Small Cells

Massive MIMO

7 Modeling

In our discussion of models, we have commonly referred to the OSI model for TCP/IP networking. This is just one model, we have also used the queuing model to measure system performance. But, just what is a model?

Defn 145 (Model)**.** A *model* is a system that is constructed to have some certain similar properties to the system in reality. A model is a way to remove certain factors from the system, or limit them, so the overall system can be tested in a variety of ways.

Models are used when the real-world experiments would be:

- Too big
- Take too much time
- No longer exists
- Dangerous
- Unethical
- Not easily reproducible
- Difficult to analyze

- Limiting

In this course, we will be looking at 2 major models of CSMA-based network traffic.

1. Kleinrock and Tobagi's Model of Packet-Switched Radio Networks
2. Bianchi Model of 802.11

7.1 Model Creation

Models must be created by following the steps below.

1. Determine the purpose of the model
2. Work out which aspects of reality need to be included, and which can be ignored or simplified
3. Choose the type of model or methodology
4. Build the model
5. Analyze the Model, Compare to reality, make predictions
6. Refine the model

The purpose of the model **MUST** be correctly determined. Otherwise, the whole model will not correctly model what you want it to.

7.2 Analyzing Models

Once a model has been constructed and has been running, whatever that may mean in its context, it must be analyzed and compared to reality.

1. What is being modeled?
2. What is the purpose for developing a model?
3. What aspects of reality are needed? What simplifications and assumptions can be made?
4. What techniques are available to build the model? How can we build it?
5. How is the model evaluated?
6. Can it produce results that can be validated (compared to reality or an already-trusted model)?
7. Can we prove the validity of the model?
8. Can it make predictions that we can test?

7.3 Simulations

Defn 146 (Simulation)**.** A *simulation* is a Model that has some dynamic components, which change over time. These are typically implemented in software, but some systems are implemented with hardware too.

Remark 146.1 (Stochastic Simulations). Simulations do not necessarily behave the same way each time, instead they follow stochastic processes.

7.4 Kleinrock and Tobagi's Model of Packet-Switched Radio Networks

The Kleinrock and Tobagi model was published in 1975.

7.4.1 What is Being Modeled?

- Packet-switched radio systems using various protocols:
  - ALOHA
  - Slotted ALOHA
  - Non-Persistent CSMA
  - 1-Persistent CSMA
  - p-Persistent CSMA
- Users in line-of-sight and range of each other, communicating over a broadcast radio channel

7.4.2 What is the Purpose of the Model?

- To investigate the throughput and delay characteristics of the various protocols
- Find the channel capacity (maximum throughput)

7.4.3 Assumptions and Simplifications

- Physics: Assume a spherical
  - A terminal cannot transmit and receive simultaneously.
    - This is actually true for wireless LANs today
    - Radios that can transmit and receive simultaneously tend to be more complicated and expensive.
    - Own transmission drowns out received transmission
  - The delay incurred by switching from one mode to the other is negligible.
    - What do we mean by negligible? How small must the delay be in reality for this assumption to be reasonable?
  - The channel is assumed to be noiseless.
    - This is sometimes true (or almost) and sometimes very much untrue.
    - This assumption limits the applicability of the model to situations where noise is negligible.
  - The propagation delay is identical for all sender-receiver pairs.
    - By taking the largest used propagation delay, we can get pessimistic bounds from the model.
  - The time required to detect the carrier is negligible.
- Protocol: This is out of scope of this work, aka Someone Else's Problem
  - Any overlap in two (or more) packets causes destructive interference in both (all) packets and they must be retransmitted.
    - The validity of this assumption depends on the error detection and correction schemes used.
    - We are therefore examining these protocols in the absence of any such schemes to deal with partial destruction of a packet.
  - The system is synchronised and all transmissions begin at the start of a slot.
    - Synchronisation is thus not included in the model, but it needs to be part of any real system that the model can be reasonably applied to.
  - There is a positive acknowledgement scheme, i.e. a message is assumed lost unless an acknowledgement is sent.
    - Again, the acknowledgement scheme is not examined in the model, but it needs to be present for the model to be applicable to a real system.
  - Acknowledgement packets are always correctly received.
  - The processing time required to check a packet is correct and generate an acknowledgement is negligible.
  - The average retransmission delay is large compared to the packet transmission time.
    - This reduces the number of cases to deal with later. Retransmissions can essentially be treated just like any other packet.
- System inputs and outputs: Play nice
  - All packets are of constant length.
    - This is not true in reality.
    - What do we lose and what do we gain by using this assumption?
  - The traffic source consists of an infinite number of users who collectively form an independent Poisson source.
    - This is perhaps the biggest simplification in the whole model, and demonstrably false in reality
    - But we gain a huge amount of analytic tractability

        * How much does the traffic model affect the results?
    - Steady state conditions are assumed to exist.
        * Kleinrock and Tobagi note in the paper that the protocols they examine have in fact been shown to be unstable.
        * However, the results are nonetheless useful in finite cases where a steady state may indeed be reached.

7.4.4 Model Construction Methodology

TODO!!

7.4.5 Model Throughputs

In all of equations of throughputs, they are true for when $a = 0$.

ALOHA

$$S = Ge^{-G} \tag{7.1}$$

Slotted ALOHA

$$S = Ge^{-2G} \tag{7.2}$$

Non-Persistent CSMA

$$S_N = \frac{G}{1+G} \tag{7.3}$$

1-Persistent CSMA

$$S_1 = \frac{Ge^{-G}(1+G)}{G+e^{-G}} \tag{7.4}$$

p-Persistent CSMA

$$S_p = \frac{Ge^{-G}(1+G)Gx}{G+e^{-G}} \tag{7.5}$$

7.4.6 Model Evaluation

- Validity: Is the model correct?
    - Mathematical proof
    - Simulation results
- Usefulness: What can we do with the model? Did it achieve its aims?
    - The model does indeed produce results for throughput and delay, which can be used in designing packet radio networks.

7.5 Bianchi Model of 802.11

This model was published in 2000, after 802.11 was already in use, but not well understood analytically.

7.5.1 What is Being Modeled

- 802.11 Distributed Co-ordination Function
    - Both basic access and with RTS/CTS, as well as a combinainon of the two where packets over a certain length use RTS/CTS.
- Saturation conditions: This means that every node always has a packet ready to send.
    - Note that maximum throughput does not occur at saturation!
    - Too many packets being sent means we lose a lot of packets to collisions. Throughput occurs when there is a balance between offered load and collision probability.

7.5.2 What is the Purpose of the Model?

- Determine the saturation throughput
  - Maximum load the system can carry under stable conditions
- Determine the optimal transmission probability each station should adopt for a given network scenario (number of stations)
- Produce a model that is both accurate and simple, so that it is easy to use.

7.5.3 Assumptions and Simplifications

- Ideal channel conditions — no hidden terminals
- Finite number of terminals
- Constant and independent collision probability of a packet transmitted by a station, regardless of previous retransmissions
- All packets have a fixed size
- Neglect CTS and ACK timeouts after collision
- Probability of a collision between more than two packets in the same slot is negligible
- Every station always has a packet waiting to be transmitted
  - This is the key assumption for this model.
  - The assumption of saturation conditions greatly reduces the scope and thus leads to tractable analysis

7.5.4 Model Construction Methodology

TODO!!

7.5.5 Model Evaluation

- Validity
  - Model results compared with results from simulation of 802.11 DCF
- Usefulness
  - Able to produce performance analysis results with comparatively simple analysis

7.6 Comparing the Two Models

- The two models have different purposes, which affects
  - Methodology
  - Assumptions
  - Evaluation
- The Kleinrock and Tobagi's Model of Packet-Switched Radio Networks is more complete and general than Bianchi's: can model a variety of different protocols under a range of conditions
- But it is also much more complex and many more assumptions and simplificatinos were required, and they were more drastic than those in the Bianchi Model of 802.11.
- Both models used simulations in their evaluation. This means they used models to evaluate other models!

8 Traffic Management

8.1 Congestion Control

Defn 147 (Congestion Control)**.** *Congestion control* refers to mechanisms for detecting and reducing Congestion.

Defn 148 (Congestion)**.** *Congestion* in a network occurs when there is too much data being sent and the network is unable to deliver it all. This can result in data loss or long delays.

8.2 Flow Control

Defn 149 (Flow Control)**.** *Flow control* refers to signalling between the sender and receiver to ensure the sender does not send data faster than the receiver can process.

A receiving host may have limited buffer space for incoming messages and takes time to process each message.

Defn 150 (Sliding Window)**.** Flow Control in Transmission Control Protocol uses a *sliding window* mechanism.

Defn 151 (Traffic Shaping)**.** *Traffic Shaping* is used to make Stochastic Process more deterministic. This can typically be done with **??**s, among other options.

Traffic Shaping is typically done by determining the totak capacity for all flows aggregated together. Then, each of the flows is divided into classes that gives each flow certain properties (voice, video, data, etc.).

```
Input  : Data Flows and their aggregated output capacity
1 for each flow do
2     if used_capacity + (new flow) < total_capacity then
3         admit
4     else
5         no admission
```

8.2.1 Token Bucket Scheme

Defn 152 (Token Bucket Scheme)**.** In a *token bucket scheme* the transmission sender gets a set of tokens, each of which corresponds to sending a packet, byte, bit, etc. To send anything, you must consume one of your tokens, and these are generated over a regular interval.

To find the maximum number of cells that can depart at any given time, we use Equation (8.1)

$$R = \rho T + \beta \tag{8.1}$$

where

R is the Maximum number of cells that can depart from the system with a given number of tokens.
ρ is the token generation/arrival rate.
T is the amount of time that passes while cells are departing.
β is the capacity of the token bucket.

To find the amount of time a transmitter can send a burst of information with a given size of token bucket, generation/arrival rate, and transmission speed, we use Equation (8.2).

$$T = \frac{\beta}{R_{\text{Max}} - \rho} \tag{8.2}$$

where

T is the maximum amount of time a transmitter can send cells.
β is the capacity of the token bucket.
R_{Max} is the maximum cell transmission rate.
ρ is the generation/arrival of new tokens.

A Complex Numbers

Complex numbers are numbers that have both a real part and an imaginary part.

$$z = a \pm bi \tag{A.1}$$

where

$$i = \sqrt{-1} \tag{A.2}$$

Remark (i vs. j for Imaginary Numbers). Complex numbers are generally denoted with either i or j. Since this is an appendix section, I will denote complex numbers with i, to make it more general. However, electrical engineering regularly makes use of j as the imaginary value. This is because alternating current i is already taken, so j is used as the imaginary value instad.

$$Ae^{-ix} = A\left[\cos(x) + i\sin(x)\right] \tag{A.3}$$

A.1 Complex Conjugates

If we have a complex number as shown below,

$$z = a \pm bi$$

then, the conjugate is denoted and calculated as shown below.

$$\overline{z} = a \mp bi \tag{A.4}$$

Defn A.1.1 (Complex Conjugate)**.** The conjugate of a complex number is called its *complex conjugate*. The complex conjugate of a complex number is the number with an equal real part and an imaginary part equal in magnitude but opposite in sign.

The complex conjugate can also be denoted with an asterisk ($*$). This is generally done for complex functions, rather than single variables.

$$z^* = \overline{z} \tag{A.5}$$

A.1.1 Complex Conjugates of Exponentials

$$\overline{e^z} = e^{\overline{z}} \tag{A.6}$$

$$\overline{\log(z)} = \log(\overline{z}) \tag{A.7}$$

A.1.2 Complex Conjugates of Sinusoids

Since sinusoids can be represented by complex exponentials, as shown in Appendix B.2, we could calculate their complex conjugate.

$$\begin{aligned}\overline{\cos(x)} &= \cos(x) \\ &= \frac{1}{2}\left(e^{ix} + e^{-ix}\right)\end{aligned} \tag{A.8}$$

$$\begin{aligned}\overline{\sin(x)} &= \sin(x) \\ &= \frac{1}{2i}\left(e^{ix} - e^{-ix}\right)\end{aligned} \tag{A.9}$$

B Trigonometry

B.1 Trigonometric Formulas

$$\sin(\alpha) + \sin(\beta) = 2\sin\left(\frac{\alpha+\beta}{2}\right)\cos\left(\frac{\alpha-\beta}{2}\right) \tag{B.1}$$

$$\cos(\theta)\sin(\theta) = \frac{1}{2}\sin(2\theta) \tag{B.2}$$

B.2 Euler Equivalents of Trigonometric Functions

$$e^{\pm j\alpha} = \cos(\alpha) \pm j\sin(\alpha) \tag{B.3}$$

$$\cos(x) = \frac{e^{jx}+e^{-jx}}{2} \tag{B.4}$$

$$\sin(x) = \frac{e^{jx}-e^{-jx}}{2j} \tag{B.5}$$

$$\sinh(x) = \frac{e^{x}-e^{-x}}{2} \tag{B.6}$$

$$\cosh(x) = \frac{e^{x}+e^{-x}}{2} \tag{B.7}$$

B.3 Angle Sum and Difference Identities

$$\sin(\alpha \pm \beta) = \sin(\alpha)\cos(\beta) \pm \cos(\alpha)\sin(\beta) \tag{B.8}$$

$$\cos(\alpha \pm \beta) = \cos(\alpha)\cos(\beta) \mp \sin(\alpha)\sin(\beta) \tag{B.9}$$

B.4 Double-Angle Formulae

$$\sin(2\alpha) = 2\sin(\alpha)\cos(\alpha) \tag{B.10}$$

$$\cos(2\alpha) = \cos^2(\alpha) - \sin^2(\alpha) \tag{B.11}$$

B.5 Half-Angle Formulae

$$\sin\left(\frac{\alpha}{2}\right) = \sqrt{\frac{1-\cos(\alpha)}{2}} \tag{B.12}$$

$$\cos\left(\frac{\alpha}{2}\right) = \sqrt{\frac{1+\cos(\alpha)}{2}} \tag{B.13}$$

B.6 Exponent Reduction Formulae

$$\sin^2(\alpha) = \frac{1-\cos(2\alpha)}{2} \tag{B.14}$$

$$\cos^2(\alpha) = \frac{1+\cos(2\alpha)}{2} \tag{B.15}$$

B.7 Product-to-Sum Identities

$$2\cos(\alpha)\cos(\beta) = \cos(\alpha-\beta) + \cos(\alpha+\beta) \tag{B.16}$$

$$2\sin(\alpha)\sin(\beta) = \cos(\alpha-\beta) - \cos(\alpha+\beta) \tag{B.17}$$

$$2\sin(\alpha)\cos(\beta) = \sin(\alpha+\beta) + \sin(\alpha-\beta) \tag{B.18}$$

$$2\cos(\alpha)\sin(\beta) = \sin(\alpha+\beta) - \sin(\alpha-\beta) \tag{B.19}$$

B.8 Sum-to-Product Identities

$$\sin(\alpha) \pm \sin(\beta) = 2\sin\left(\frac{\alpha \pm \beta}{2}\right)\cos\left(\frac{\alpha \mp \beta}{2}\right) \tag{B.20}$$

$$\cos(\alpha) + \cos(\beta) = 2\cos\left(\frac{\alpha + \beta}{2}\right)\cos\left(\frac{\alpha - \beta}{2}\right) \tag{B.21}$$

$$\cos(\alpha) - \cos(\beta) = -2\sin\left(\frac{\alpha + \beta}{2}\right)\sin\left(\frac{\alpha - \beta}{2}\right) \tag{B.22}$$

B.9 Pythagorean Theorem for Trig

$$\cos^2(\alpha) + \sin^2(\alpha) = 1^2 \tag{B.23}$$

B.10 Rectangular to Polar

$$a + jb = \sqrt{a^2 + b^2}e^{j\theta} = re^{j\theta} \tag{B.24}$$

$$\theta = \begin{cases} \arctan\left(\frac{b}{a}\right) & a > 0 \\ \pi - \arctan\left(\frac{b}{a}\right) & a < 0 \end{cases} \tag{B.25}$$

B.11 Polar to Rectangular

$$re^{j\theta} = r\cos(\theta) + jr\sin(\theta) \tag{B.26}$$

C Product Simplifications

$$\prod_{i=1}^{N} x^{-i} = x^{\frac{N(1+N)}{x}} \tag{C.1}$$

D Calculus

D.1 Fundamental Theorems of Calculus

Defn D.1.1 (First Fundamental Theorem of Calculus)**.** The *first fundamental theorem of calculus* states that, if f is continuous on the closed interval $[a, b]$ and F is the indefinite integral of f on $[a, b]$, then

$$\int_a^b f(x)\,dx = F(b) - F(a) \tag{D.1}$$

Defn D.1.2 (Second Fundamental Theorem of Calculus)**.** The *second fundamental theorem of calculus* holds for f a continuous function on an open interval I and a any point in I, and states that if F is defined by

$$F(x) = \int_a^x f(t)\,dt,$$

then

$$\begin{aligned} \frac{d}{dx}\int_a^x f(t)\,dt &= f(x) \\ F'(x) &= f(x) \end{aligned} \tag{D.2}$$

Defn D.1.3 (argmax)**.** The arguments to the *argmax* function are to be maximized by using their derivatives. You must take the derivative of the function, find critical points, then determine if that critical point is a global maxima. This is denoted as

$$\underset{x}{\operatorname{argmax}}$$

D.2 Rules of Calculus

D.2.1 Chain Rule

Defn D.2.1 (Chain Rule)**.** The *chain rule* is a way to differentiate a function that has 2 functions multiplied together.

If

$$f(x) = g(x) \cdot h(x)$$

then,

$$\begin{aligned} f'(x) &= g'(x) \cdot h(x) + g(x) \cdot h'(x) \\ \frac{df(x)}{dx} &= \frac{dg(x)}{dx} \cdot g(x) + g(x) \cdot \frac{dh(x)}{dx} \end{aligned} \tag{D.3}$$

E Laplace Transform

Defn E.0.1 (Laplace Transform)**.** The *Laplace transformation* operation is denoted as $\mathcal{L}\{x(t)\}$ and is defined as

$$X(s) = \int_{-\infty}^{\infty} x(t)e^{-st}dt \tag{E.1}$$

References

[GK00] P. Gupta and P.R. Kumar. “The Capacity of Wireless Networks.” In: *IEEE Transactions on Information Theory* (2000).