Bits

1 Byte = 8 Bits.

Decimal to Binary Conversion

Example: `45.375`

1. Integer Part (`45`)

- **Method:** Repeatedly divide by 2 & note the remainder.
- **Calculation:**
  - `45 / 2 = 22` Remainder `1`
  - `22 / 2 = 11` Remainder `0`
  - `11 / 2 = 5` Remainder `1`
  - `5 / 2 = 2` Remainder `1`
  - `2 / 2 = 1` Remainder `0`
  - `1 / 2 = 0` Remainder `1`
- **Result (read remainders from bottom to top):** `101101`
- **Pad to 8 bits (on the left):** `00101101`

2. Fractional Part (`0.375`)

- **Method:** Repeatedly multiply by 2 & note the integer part.
- **Calculation:**
  - `0.375 * 2 = 0.75 -> 0`
  - `0.75 * 2 = 1.5 -> 1`
  - `0.5 * 2 = 1.0 -> 1`
- **Result (read digits from top to bottom):** `011`
- **Pad to 4 bits (on the right):** `0110`

3. Combine

- **Integer:** `00101101`
- **Fraction:** `0110`
- **Final (12-Bit):** `001011010110`

Binary to Decimal Conversion

Example: `11001.1101`

1. Integer Part (`11001`)

- **Calculation:**
  - `1 * 2^4 -> 16`
  - `1 * 2^3 -> 8`
  - `0 * 2^2 -> 0`
  - `0 * 2^1 -> 0 1 * 2^0 -> 1`

Total = 25

2. Fractional Part (`.1101`)

- **Calculation:**
  - `1 * 2^-1` -> 0.5
  - `1 * 2^-2` -> 0.25
  - `0 * 2^-3` -> 0
  - `1 * 2^-4` -> 0.0625

Total = 0.8125

3. Combine

- **Integer:** `25`
- **Fraction:** `0.8125`
- **Final (12-Bit):** `25.8125`

Special Case: Loss of Precision (Ex: `2.3`)

- Not every fraction can be represented exactly.
- **Integer** `2` **->** `00000010`
- **Fraction** `0.3` **->** `0.010011...` **(repeating)**
- **Truncated to 4 bits:** `0100` (which represents `0.25`)
- **Final result for 2.3:** `000000100100` (represents `2.25`)

Bytes

- A kilobyte, or KB, is $1,024$ bytes
- A megabyte, or MB, is $1,024^2$ bytes
- A gigabyte, or GB, is $1,024^3$ bytes
- A terabyte, or TB, is $1,024^4$ bytes
- A petabyte, or PB, is $1,024^5$ bytes

Com Net

Key Networking Components

- **LAN:** A private, local network (e.g., home or office).
- **WAN:** A broad network connecting cities or countries (e.g., the Internet).
- **IP Address:** A unique identifier that ensures data packets reach the correct destination.
- **Router:** A gateway that connects different networks (LAN to WAN) and directs traffic between them.
- **Switch:** An intelligent hub within a LAN that learns device locations to distribute data efficiently.
- **Packet Switching:** Breaking data into small chunks (packets) and transmitting them at a specific rate (R).
- **Modem:** Converts signals from your ISP (e.g., fiber or cable) into a digital format your router understands.
- **Access Point (AP):** A device that creates a wireless signal (Wi-Fi) so devices can connect to the LAN without cables.
- **Firewall:** A security system that monitors and filters incoming/outgoing traffic based on predefined rules.
- **Network Interface Card (NIC):** The hardware component (in a PC or laptop) that allows a device to connect to a network.
- **DNS (Domain Name System):** The "phonebook" of the internet; it translates human-readable names (google.com) into IP addresses.

Protocol Layers

Protocols define format, order of msgs sent and received among network entities, and actions taken on msg transmission, receipt.

- Applications: FTP, SMTP, HTTP.
- Transport: TCP, UDP.
- Network: IP, routing protocols.
- Link: Ethernet, WiFi.
- Physical: Ethernet, WiFi.

Advantages and disadvantages of Protocol Layers

Advantage

- Flexibility: Individual parts are replaceable.
- Simplicity: Focus on a specific task.
- Interoperability: All devices understand each other.

Disadvantage

- Overhead: Data packets become larger due to multiple headers.
- Performance: "Packaging" and "unpackaging" takes time.
- Duplication: Some functions are performed multiple times.

Domain Name System

Translates domain names to numerical IP addresses.

- DNS Recursor (Client Side): Initiates the query. It's the first stop and acts as a librarian trying to find the IP address.
- Root Nameserver (The Dot '.'): The starting point. It directs the Recursor to the correct Top-Level Domain (TLD) Nameserver.
- TLD Nameserver (e.g., .com, .de, .org): Manages all domains under its specific extension. It points the Recursor to the Authoritative Nameserver for the specific domain requested.
- Authoritative Nameserver: Holds the definitive DNS records (A, CNAME, MX, etc.) for the requested domain. It provides the actual IP address back to the Recursor.
- DNS Recursor / Client: Receives the IP address and finally connects to the web server to load the website.

Packaet Transmission Delay

This is the time required by the transmitter to push all bits of a data packet onto the cable (or medium).

- L: Packet length (bits).
- R: Link bandwidth (dps).

$$d_{\text{trans}} = \frac{L \quad \text{(bits)}}{R \quad \text{(bits/sec)}}$$

In a network, the **slowest** point always determines the overall speed. This is known as the bottleneck principle.

$$R_{av} = \min\{R_c, R_s, \dots, R_n\}$$

Propagation delay

This is the time it takes for a single bit to physically travel from point A to point B.

- d: Length of physical link.
- s: Propagation speed in medium.

$$d_{\text{prop}} = \frac{d}{s}$$

Rate

Rate (bits/time unit) at which bits transferred between sender/receiver. We disquinsh between:

- Instantaneous: Rate at given point in time.
- Average: Rate over longer period of time.

Delay & Loss in bash

```
# Trace
traceroute www.google.ch

# Total way
ping www.google.ch
```

Domain Name System

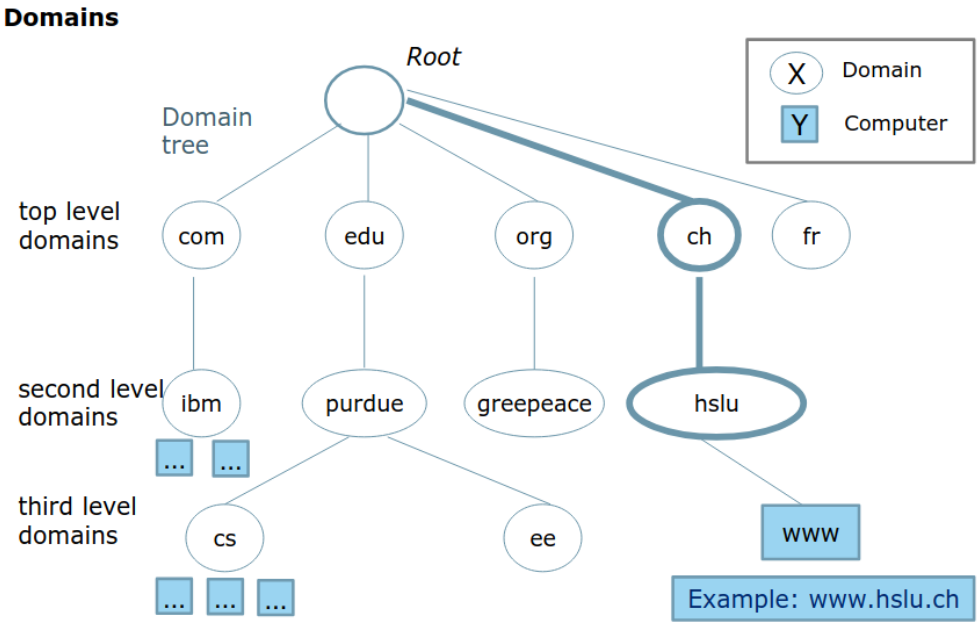

Figure 1: Domain Tree

HTML

Uniform Resource Locator (URL)

Reference to a web resource that specifies its location in a computer network (e.g. web page, video, image, etc.)

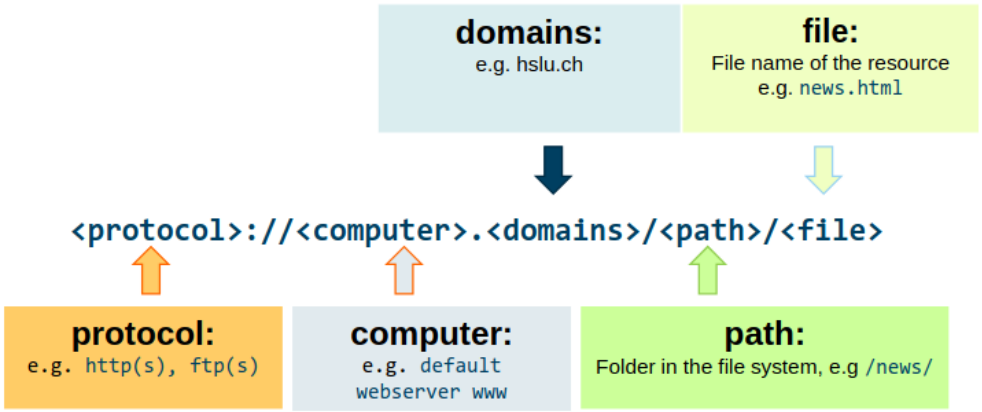

Figure 2: URL Structure

HTML Quick Summary

Core Structure

- `<!DOCTYPE html>`: HTML5 declaration.
- `<html>`: Document root.
- `<head>`: Metadata/Title.
- `<body>`: Visible content.

Basic Elements

- **Headings**: `<h1>` to `<h6>` (Text size presets).
- **Paragraph**: `<p>content</p>` (Text blocks).
- **Formatting**: `<b>` (Bold), `<i>` (Italic), `<u>` (Underline).
- **Links**: `<a href="URL">Text</a>`.
- **Images**: `<img src="path">`.

Nested Elements

Elements inside others: `<p>Star<b>Wars</b></p>`.

Example HTML

```
<!DOCTYPE html>
<html>
<head>
    <link rel="stylesheet" href="style.css">
</head>
<body>
  <h1>Hello</h1>
  <p>Check this <a href="#">Link</a>.</p>
</body>
</html>
```

Example CSS

```css
/* Selects the h1 element */
h1 {
  color: blue;
  font-family: Arial, sans-serif;
}

/* Selects all paragraphs */
p {
  font-size: 16px;
  line-height: 1.5;
}
```

Visualisation

Visual Variables

- Mark / Shape: Used shapes to distinguish between different types in a plot
- Position: Position of an object Used to encode data
- Size (Length, Area and Volume): Sizes to encode data
- Brightness: Mapping values to the plot
- Color: Colormap that can be used to encode a data variable
- Orientation: Used to encode data
- Texture: Mapping values to the plot
- Motion: Associated with any of the other visual variables

Recursion Essentials

Recursion involves a function calling itself to solve a smaller sub-problem.

3-Step Pattern

1. **Base Case:** The exit condition to stop recursion (prevents Stack Overflow).
2. **Recursive Step:** The function calls itself with reduced input.
3. **Convergence:** Ensure the input always moves toward the Base Case.

Code Example (Python)

```python
def factorial(n):
    if n <= 1: return 1     # 1. Base Case
    return n * factorial(n - 1) # 2. Recursive Step
```

Cloud Comp

Hardware Essentials

Central Processing Unit (CPU)

The CPU is the “brain” of the computer, processing commands via the **Instruction Cycle**: * **Fetch**: Retrieves an instruction from the RAM. * **Decode**: Translates the instruction into internal signals. * **Execute**: Carries out the operation and stores the result.

Random Access Memory (RAM)

- **Function**: High-speed storage for data and instructions currently in use by the CPU.
- **Property**: **Volatile**—all data is lost when the power is turned off.

Cloud Computing Fundamentals

Business Model

- **Client**: Pays based on actual usage rather than provisioning for peak capacity.
- **Provider**: Leverages shared capabilities to serve multiple users efficiently.

Core Cloud Properties

- **Self-managing Services**: Consumers can provision resources automatically as needed without requiring human interaction with the provider.
- **High Accessibility**: Resources are available over the network anytime and from any location.

Infra

Feature	IaaS	PaaS	SaaS
You Manage	OS, Apps, Data	Applications, Data	Nothing (just usage)
Provider Manages	Hardware, Network	Hardware, OS, Stack	Everything
Example	AWS (EC2), Azure	Google App Engine	Gmail, Salesforce

Figure 3: Comparison Summary of Cloud Services

Virtualization

Virtualization is essentially a way to trick a computer into thinking it is running multiple separate machines, all on the same physical hardware. The piece of software that makes this possible is called a Hypervisor.

Virtualization: Hypervisors

A Hypervisor is software that creates and runs virtual machines.

- **Type 1 (Bare-Metal)**:
  - **Setup**: Installed directly on the physical hardware.
  - **Pros**: High performance, efficient, and secure due to direct hardware access.
  - **Usage**: Enterprise servers and data centers (e.g., VMware ESXi).
- **Type 2 (Hosted)**:
  - **Setup**: Runs as an application on top of an existing Operating System (Windows/macOS).
  - **Pros**: Easy to set up for development and testing.
  - **Usage**: Personal computers (e.g., VirtualBox).

  ### Virtualization: Pros & Cons

Advantages

- **Efficiency**: Maximizes hardware use by running multiple VMs on one physical server.
- **Cost Savings**: Lowers expenses for hardware, power, and cooling.
- **Isolation**: Enhances security; a compromised VM does not affect others on the same host.
- **Disaster Recovery**: Enables instant movement or restoration of entire VMs.

Disadvantages

- **Single Point of Failure**: If the physical host fails, all hosted VMs crash.
- **Performance Overhead**: The hypervisor consumes resources, causing a slight speed loss.
- **Complexity**: Requires specialized expertise and software to manage.
- **Resource Contention**: Multiple VMs competing for the same CPU/RAM can cause bottlenecks.

Processing Data

`grep`

Take a stream of text or data, perform operations on it, and produce a modified version of that stream as output.

```
grep -e <REGEX_PATTERN> <FILE>
```

`cut`

Selects portions of each line (as specified by list) from each file and writes them to the output.

```
cut -d <DELIMITER> -f <FIELD_SELECTION> <FILE>
```

`wget`

Wget is a free utility for non-interactive download of files and whole websites (follows links) from the Internet.

Pipes & Filters Pattern

A pipe is used in Linux to send the output of one command/process to another command/process for further processing. Decompose a task that performs complex processing into a series of separate processing steps that can be reused.

```
grep -e <REGEX_PATTERN> <FILE> | cut -d <DELIMITER> -f <FIELD_SELECTION> <FILE>
```

5V-Model

The 5 V's of big data are Volume, Velocity, Variety, Veracity, and Value. These characteristics help define the challenges and opportunities associated with managing and analyzing large data sets.

Regular Expressions

A regular expression, often shortened to regex, is a sequence of characters that defines a search pattern. It uses a specific syntax combining literal characters (e.g., a, 1) and metacharacters (e.g., . , *, []) which have special meaning.