3.7 USE CASES IN BIOMEDICAL IMAGING, POWER GRID MODELING, CIRCUIT ROUTING

Data structures are **essential in complex applications** where large amounts of data need to be stored, accessed, and processed efficiently. The following are some key domains:

1. Biomedical Imaging

Overview

Biomedical imaging includes technologies like MRI, CT scans, and ultrasound, which produce large amounts of digital image data. Efficient processing and storage of this data require **appropriate data structures**.

Data Structures Used

1. **Arrays / Matrices**
    - Images are stored as **2D or 3D arrays** of pixels/voxels.
    - Each element represents a **pixel intensity or color value**.
2. **Linked Lists**
    - Store **dynamic sequences of image slices** in 3D imaging (like CT scans).
3. **Trees (Quadtrees / Octrees)**
    - Used for **image compression** or **region segmentation**.
    - Quadtrees divide images into **smaller squares** recursively for faster processing.
4. **Graphs**
    - Used in **image segmentation**, where pixels are nodes and edges represent similarity between pixels.

Example Applications

- Fast rendering of MRI slices using arrays
- Image compression using **quadtrees**
- Detecting tumors using graph-based image segmentation

Benefits

- Efficient memory usage for large images
- Faster access and processing
- Supports complex operations like filtering and segmentation

2. Power Grid Modeling

Overview

Power grids are complex networks with **generators, substations, and loads connected through transmission lines**. Modeling these systems efficiently is critical for **simulation, optimization, and fault detection**.

Data Structures Used

1. **Graphs**
    - Nodes = generators, substations, consumers
    - Edges = transmission lines
    - Supports **power flow calculations** and network analysis
2. **Matrices (Adjacency / Incidence Matrices)**
    - Store **connections between nodes** in a structured way
    - Enables **matrix-based simulations** of grid behavior
3. **Queues / Priority Queues**
    - Used in **load balancing algorithms** and fault handling
    - Priority queues help in **dispatching power efficiently**

Example Applications

- Shortest path algorithms for **power routing**
- Load flow analysis using adjacency matrices
- Detecting and isolating faults in real-time

Benefits

- Realistic simulation of power grids
- Fast computation of flow and optimization
- Easy modeling of large-scale grids

3. Circuit Routing (VLSI / PCB Design)

Overview

Circuit routing involves connecting **components on a chip or PCB** using wires or traces. This requires **efficient path finding and layout optimization**.

Data Structures Used

1. **Graphs**
    - Nodes = circuit pins
    - Edges = possible wiring paths
    - Used for **routing algorithms** like maze routing
2. **Trees (Spanning Trees)**
    - **Minimum Spanning Tree (MST)** used for **signal routing with minimal wire length**
3. **Grids / Arrays**
    - Represent the **layout space of the chip or board**
    - Each cell indicates **free or occupied space**
4. **Stacks / Queues**
    - Manage **routing paths during exploration** (DFS/BFS traversal)

Example Applications

- Maze routing algorithm for PCB traces
- Steiner tree algorithms for chip interconnects
- Layer assignment using arrays to avoid overlaps

Benefits

- Reduces wire length and cross-talk
- Optimizes layout for manufacturing
- Enables automated design tools to handle complex circuits

Domain	Data Structures Used	Purpose / Benefit
Biomedical Imaging	Arrays, Linked Lists, Trees, Graphs	Store images, compression, segmentation, rendering
Power Grid Modeling	Graphs, Matrices, Queues	Model network, simulate power flow, optimize dispatch
Circuit Routing	Graphs, Trees, Arrays, Stacks/Queues	Connect components efficiently, optimize PCB layout

- **Graphs** are heavily used in networked systems (power grids, circuit routing, segmentation).
- **Arrays and matrices** are best for structured data like images and grid simulations.
- **Trees** are useful for optimization and compression.
- **Queues and stacks** help in traversal and scheduling tasks.