Python Tutorial – Part II: Data

Mark A. Austin

University of Maryland

austin@umd.edu
ENCE 201, Fall Semester 2025

September 2, 2025

Overview

1. Data-Driven Decision Making
2. Tabular and Non-Tabular Data Models
   - Tabular and Non-Tabular Data Models
   - Homogeneous and Heterogeneous Data
3. Tabular Data and Dataset Transformation (Pandas)
   - Basic Operations (Data Series and Dataframes)
   - Intermediate Operations (Cleaning Data)
   - Advanced Operations (Data Filtering, Data Merge)
4. Spatial Data and Dataset Transformation (GeoPandas)
   - GeoPandas Data Model
   - Models of Geometric Objects (points, lines, polygons)
   - Applications (Urban and Global GeoDataModeling)
5. Appendix A: From Data Models to Data Structures

Data-Driven

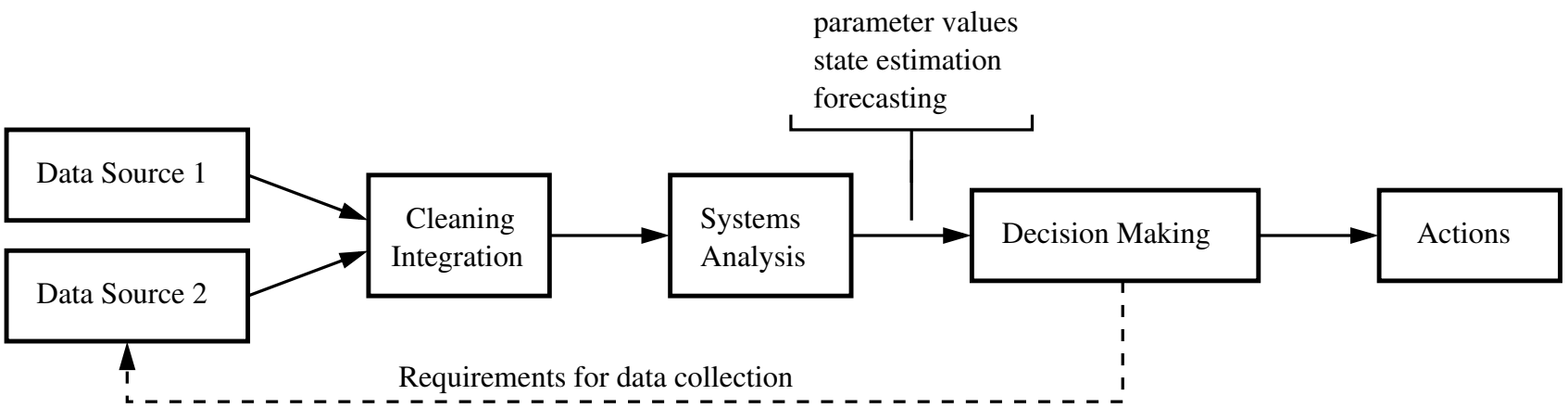
Decision Making

Data-Driven Decision Making

Data-Driven Decision Making

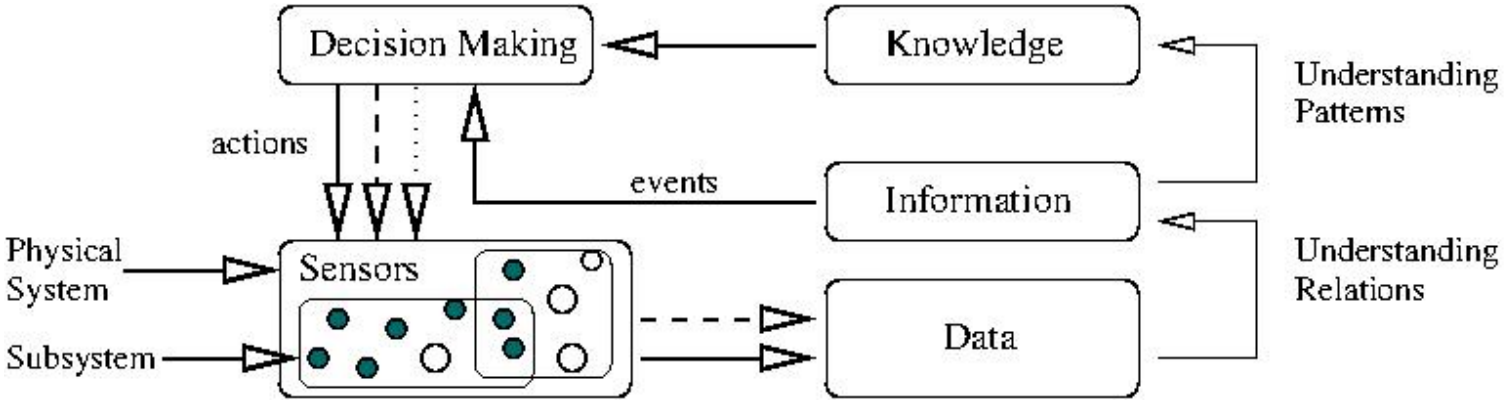
Data-driven decision making is the process of collecting data and using analysis to gain insights and inform engineering decisions, rather than rely solely on intuition and/or experience.

Simplified Procedure:



Data-Driven Decision Making

Framework for Sensor-Enabled Decision Making:

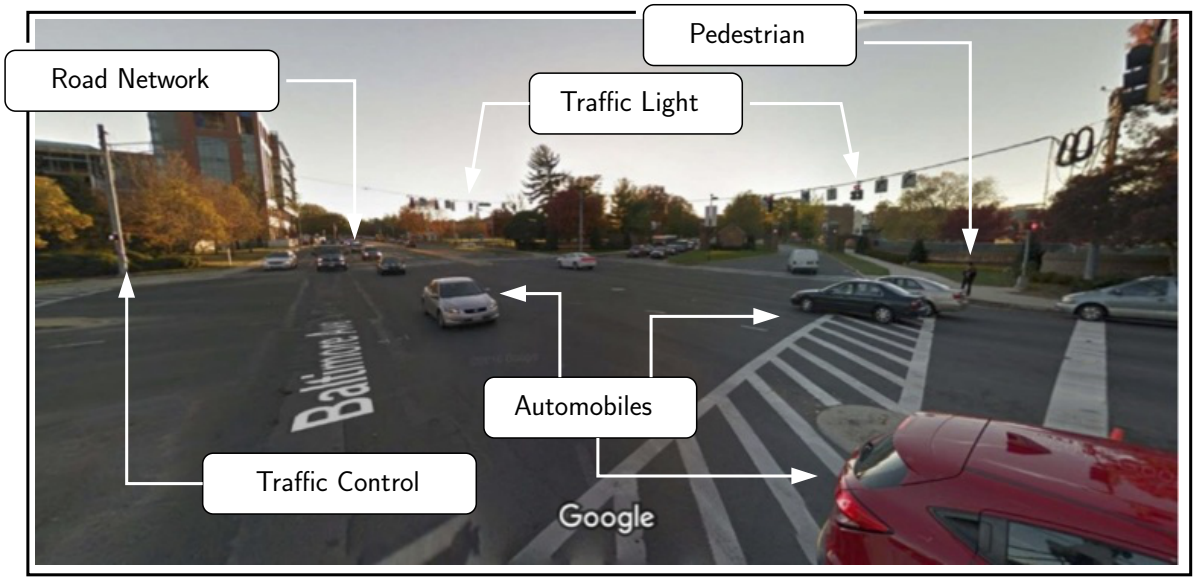


Chain of dependency relationships:

```
1.  improved performance <-- actions
2.  actions              <-- ability to identify events.
3.  identify events      <-- data processing
4.  data processing      <-- types and quality of data
5.  types and quality of data <-- sensor design and placement.
```

Data-Driven Decision Making

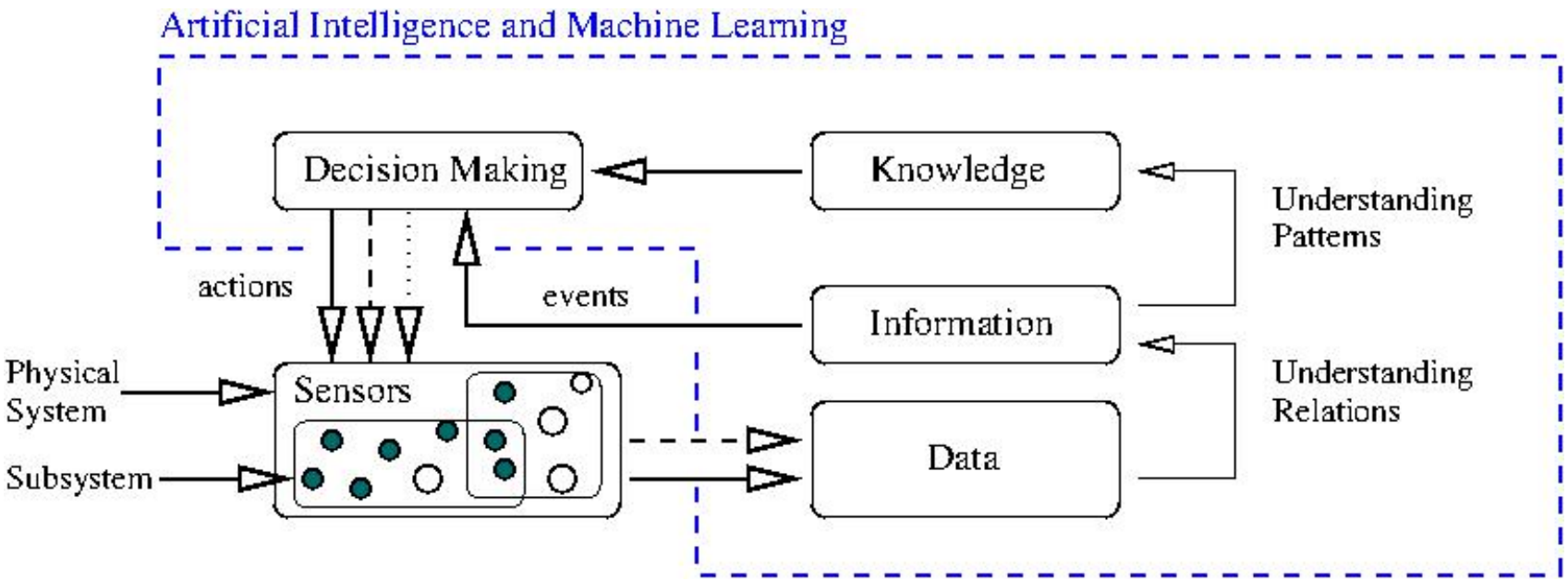
Example A: Traversal of a traffic intersection ...



Required Capability. Observe, evaluate, reason, take actions.
Challenges. Multiple domains, multiple streams of heterogeneous data, event-driven behavior, dynamic, time critical.

Data-Driven Decision Making

Solution Procedure: Pathway from sensing and data collection to AI/ML-enabled decision making ...



Challenges: Stringent requirements for real-time processing $\longrightarrow$ drive custom approaches to collection and fusion of data.

Data-Driven Decision Making

No sensor type works well for all tasks and in all conditions, so sensor fusion will be necessary to provide redundancy for autonomous functions

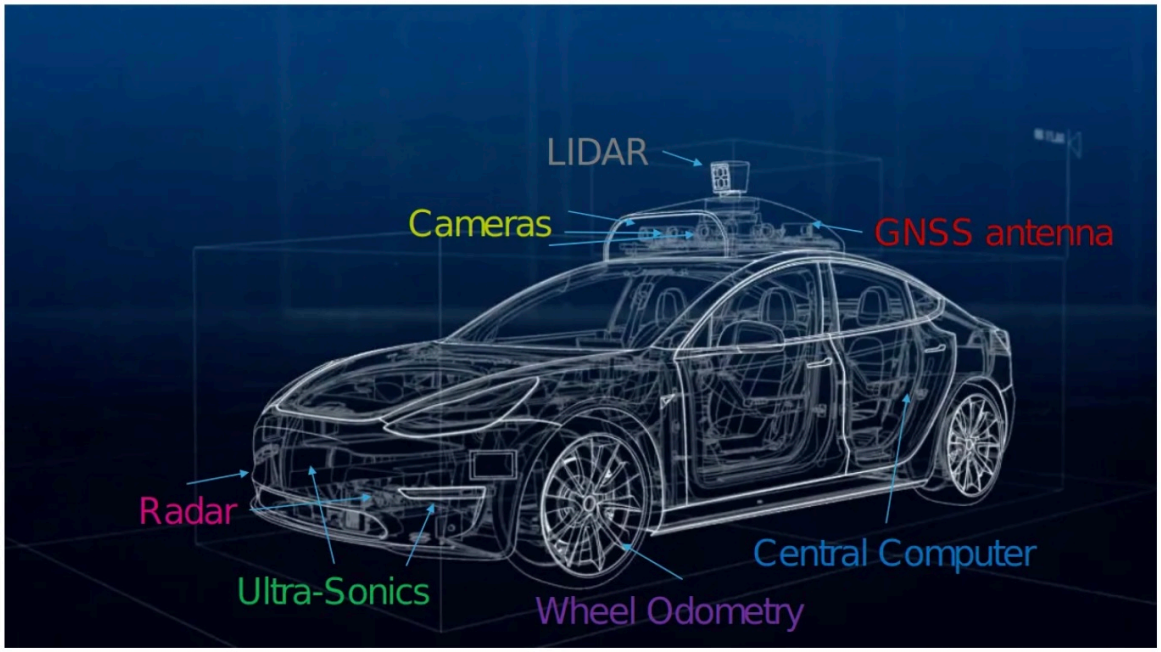
Most likely used fusion solution in future | Good | Fair | Poor

	Camera	Radar	LiDAR	Ultrasonic	LiDAR+Radar+ Camera
Object detection	Fair	Good	Good	Good	Good
Object classification	Good	Poor	Fair	Poor	Good
Distance estimation	Fair	Good	Good	Good	Good
Object edge precision	Good	Poor	Good	Good	Good
Lane tracking	Good	Poor	Poor	Poor	Good
Range of visibility	Fair	Good	Fair	Poor	Good
Functionality in bad weather	Poor	Good	Fair	Good	Good
Functionality in poor lighting	Fair	Good	Good	Good	Good

Source: ...

Data-Driven Decision Making

Solution Procedure: Self-driving car sensors ...



Diversity of Sensors: GPS/position, radar, orientation, video, microphones.

Source: ...

Data-Driven Decision Making

Video Cameras: (corners of car and roof) ...

- Help to recognize objects around the vehicle.

Position Sensors: (located in wheels) ...

- Helps to determine car's location from wheel rotations.

LIDAR/Radar: (located on car corners + roof) ...

- Measures speed of cars ahead.

Ultrasonic: (located on car bumpers) ...

- Near-range sensing to support operations such as parking.

Orientation Sensor: (located in car interior) ...

- Acts like the car's inner ear, sensing motion and balance.

Microphone: (located on roof) ...

- Detect sirens of approaching emergency vehicles.

Data-Driven Decision Making

Example B: Buildings and Cities.

Use sensing and software to:

- Improve situational awareness – to understand what is actually happening in a building or city?
- Connect sensor measurements to short- and long-term urban needs (e.g., decisions on a bus stop; longer term urban planning).
- Capture the spatial, temporal, and intensity aspects of environmental phenomena (e.g., fires, flooding) and their impact on natural (e.g., air quality) and man-made systems (e.g., transportation networks, food chains).
- Look ahead and forecast future states of the system?

Data-Driven Decision Making

Historical Problems:

- Manual (ad-hoc) data acquisition procedures are slow and error prone:
  - Missing data,
  - Inconsistent formatting (e.g., of dates),
  - Poorly written (ambiguous) textual descriptions,
  - Spelling mistakes,
  - Duplicate data, etc ...
- Data is poorly organized.
- Analysis procedures are computationally inefficient.

Too often:

- Data is collected without any real thought for end usage.
- Open source data is outdated.

Present-Day Challenges

Large Data Volume:

- Large volume (size) of data collected and analyzed.
- Steadily increases with time: gigabytes, ... petabytes, ...

Diverse Data Type and Format:

- Diverse range of data types (numeric, textual, spatial, temporal, audio, visual) and formats (structured, non-structured, semi-structured).
- Data can be noisy ...

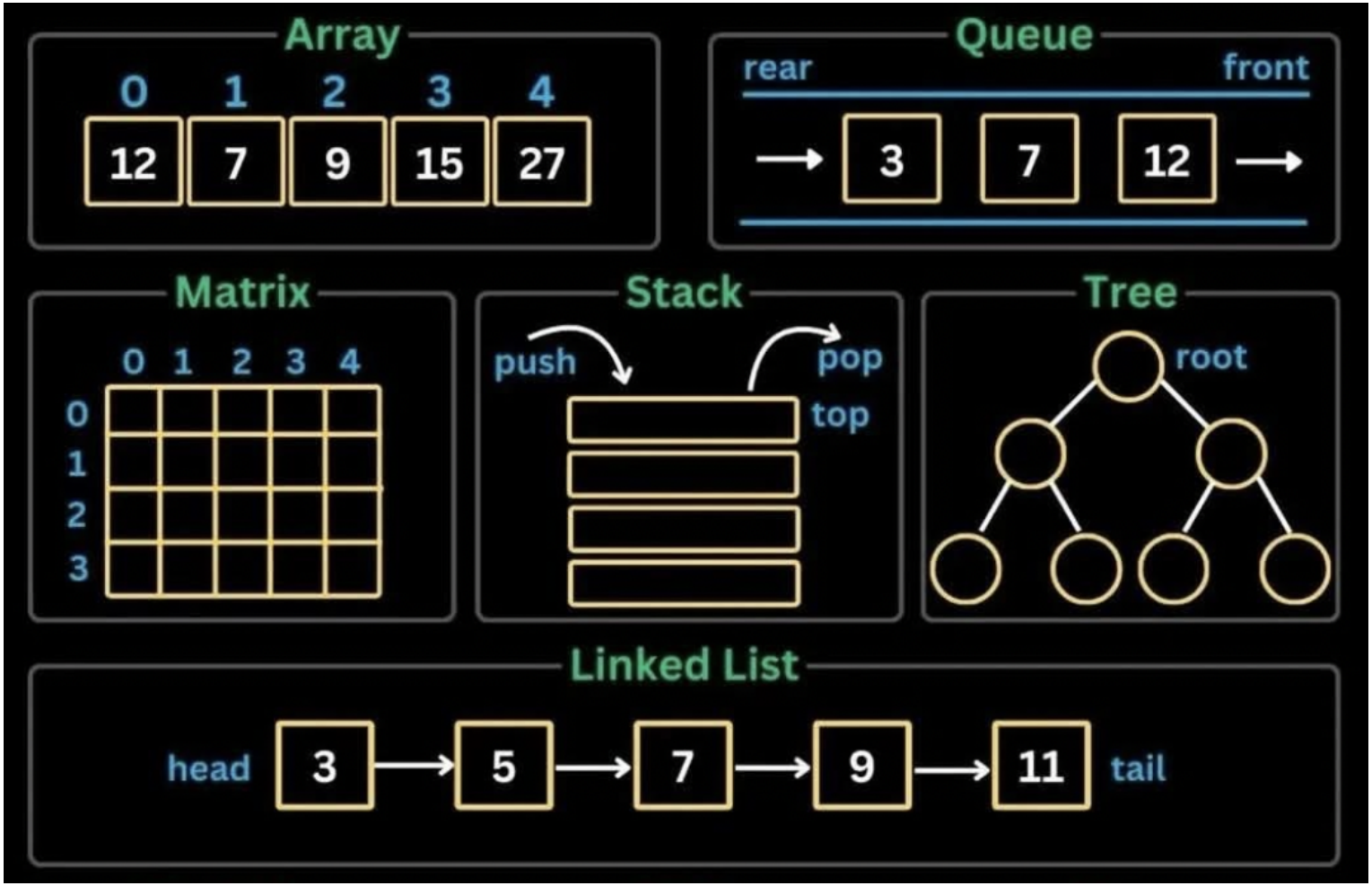
High Data Velocity:

- Refers to the speed (static vs highly dynamic) at which data is generated, collected, processed, and analyzed.
- Provides systems/organizations with insight in real time.

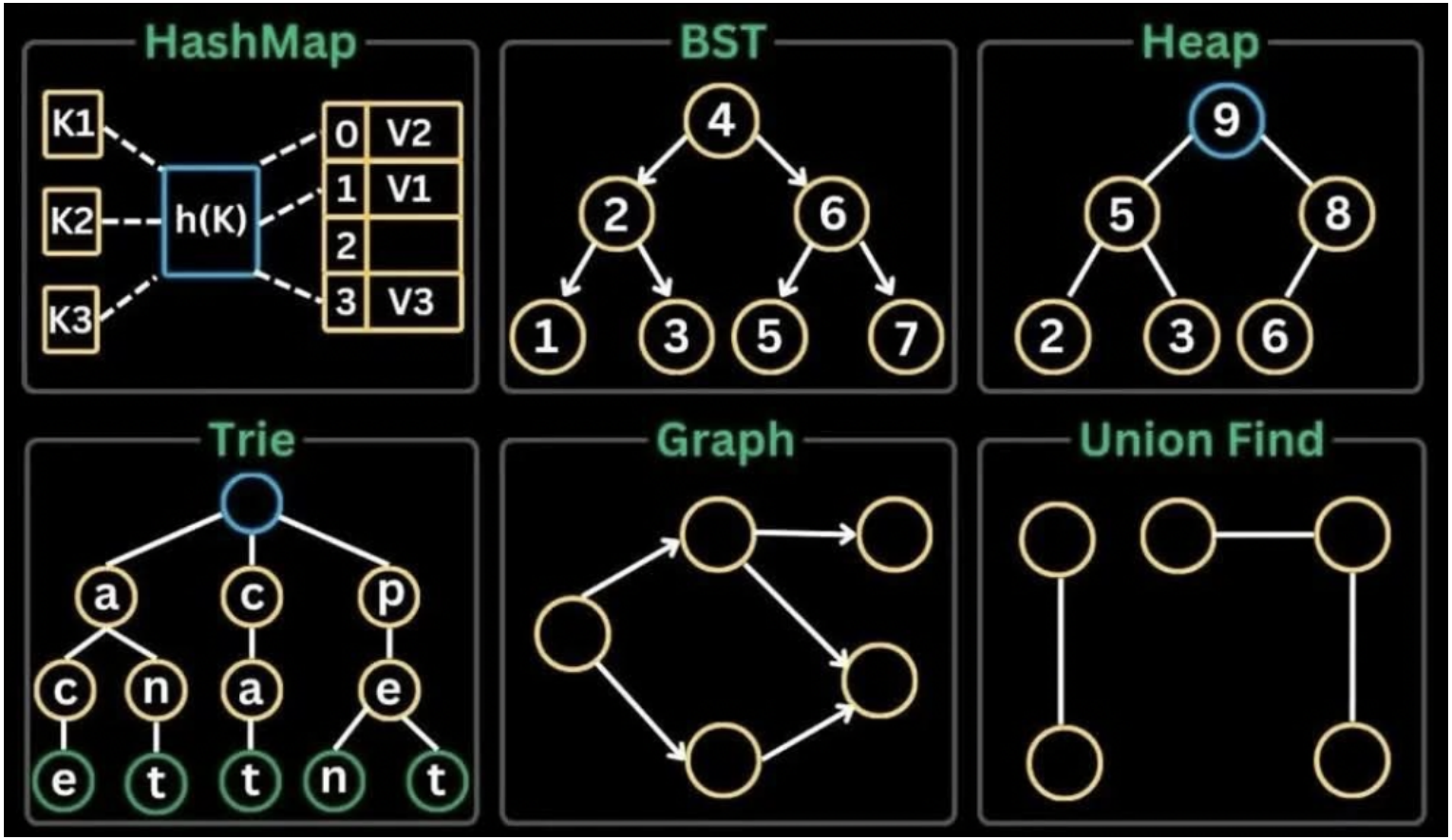
Appendix A

(From Data Models to Data Structures)

From Data Models to Data Structures



From Data Models to Data Structures



From Data Models to Data Structures

Arrays:

- Store multiple elements (of the same type) in a block of contiguous memory.
- Elements are accessed by index.
- Arrays have a fixed or resizeable size.

Matrices:

- Two-dimensional array of elements, organized into rows and columns.
- Elements are accessed using a combination of row index and column index.

From Data Models to Data Structures

Queue:

- Queues follow the rule: first-in first-out order.
- Efficient insertion at the rear. Efficient removal (deletion) at the front.

Stack:

- Stacks follow the rule: last-in first-out order.
- Efficient insertion and deletion of items at the top.

From Data Models to Data Structures

Linked List:

- Stores elements with next (and sometimes previous) node references.
- Supports dynamic size adjustments.
- Very efficient operations for node insertion and deletion.

HashMap:

- Employ (key, value) pairs for quick lookup.
- Uses a hash function for indexing.
- Efficient retrieval, insertion, and deletion operations.

From Data Models to Data Structures

Tree:

- Organizes data into a hierarchical structure.
- Elements have parent and child relationships.
- Used for hierarchy, search and sort.

Binary Search Tree (BST):

- Each node in the tree structure has a most two children, a left child and a right child.
- Left child's value is always less than the parent's node value.
- Right child's value is always greater than parent's node value.
- Hierarchy structure makes searching

From Data Models to Data Structures

Heap:

- A binary tree-based data structure that satisfies the max-heap (min-heap) property:
- Max-heap property: for every node, its value is greater than or equal (less than or equal) to the values of its children.
- The root stores the max value (max-heap) within the heap.

Trie:

- Tree-like structure for efficient storage/retrieval of strings.
- Strings are organized based on their prefixes, allowing for fast searches and auto-completion.
- Each node represents a character of a string.
- Pathway from root –> node: prefix of a stored string.

From Data Models to Data Structures

Graph:

- Represent relationships between vertex and edge entities.
- Used for modeling (physical, transportation, social) networks.

Key Takeaway:

When you have a chance (not right now) take a course on data structures and algorithms – they are the foundation of models and processes for good software development.