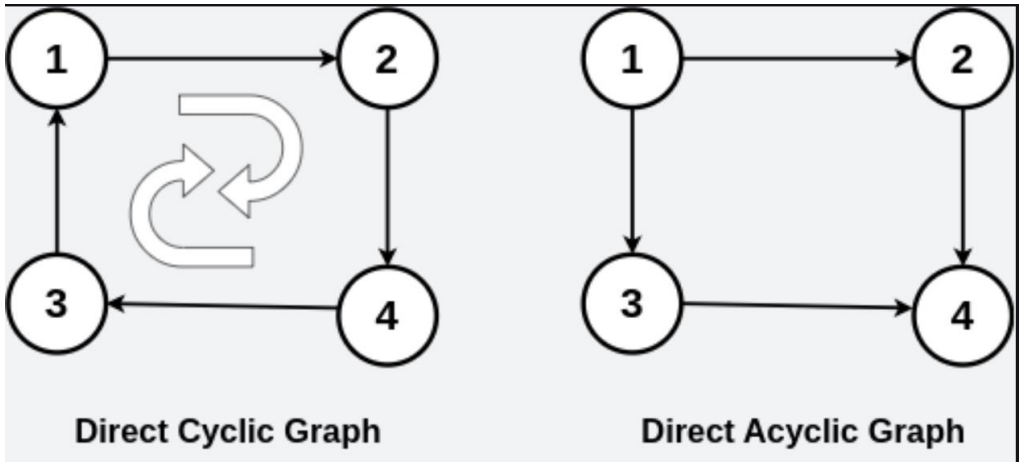
Application of Directed Acyclic Graph

PRESENT BY: BOM BAHADUR BK AND RAJARAM PAKUR
LEVEL: MASTER
NEPAL COLLEGE OF INFORMATION TECHNOLOGY(NCIT)
BALKUMARI, LALITPUR

What is Directed Acyclic Graph

- A Directed Acyclic Graph (DAG) is a directed graph that does not contain any cycles.



Properties of Directed Acyclic Graph

- **No Cycles**

  A DAG does not contain any cycles, meaning no path in the graph forms a closed loop.

- **Directed Edges**

  All edges in a DAG have a direction, from one vertex to another.

- **Topological Ordering**

  It is possible to arrange the vertices in a linear order such that for every directed edge (u → v), vertex u appears before v in the ordering.

- **Reachability**

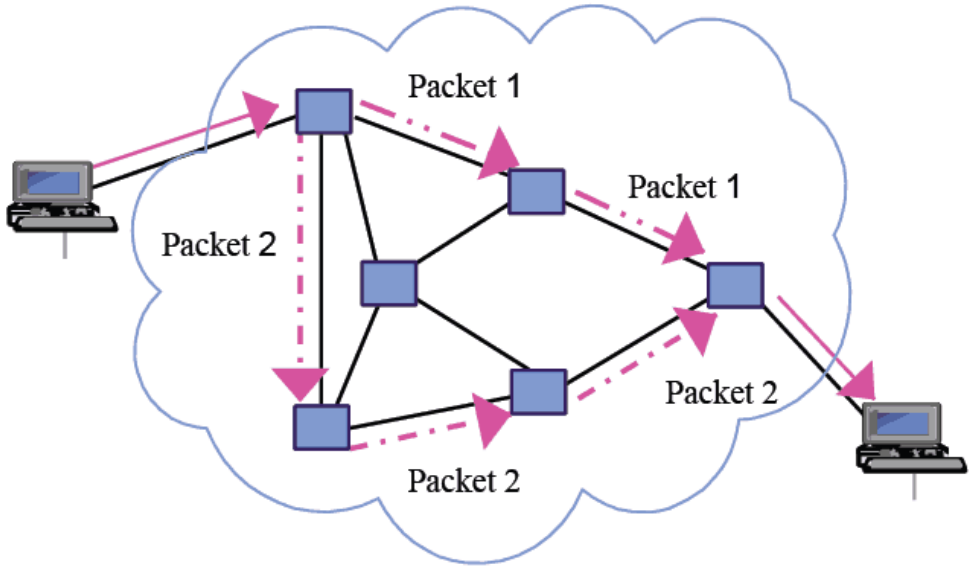
  A DAG supports reachability analysis, where you can determine whether one vertex can reach another.

Application of DAG

1. Routing in computer networks
2. Version control system
3. Compiler Design
4. Job scheduling
5. Data processing frameworks
6. Genealogy and family tree
7. Citation graphs

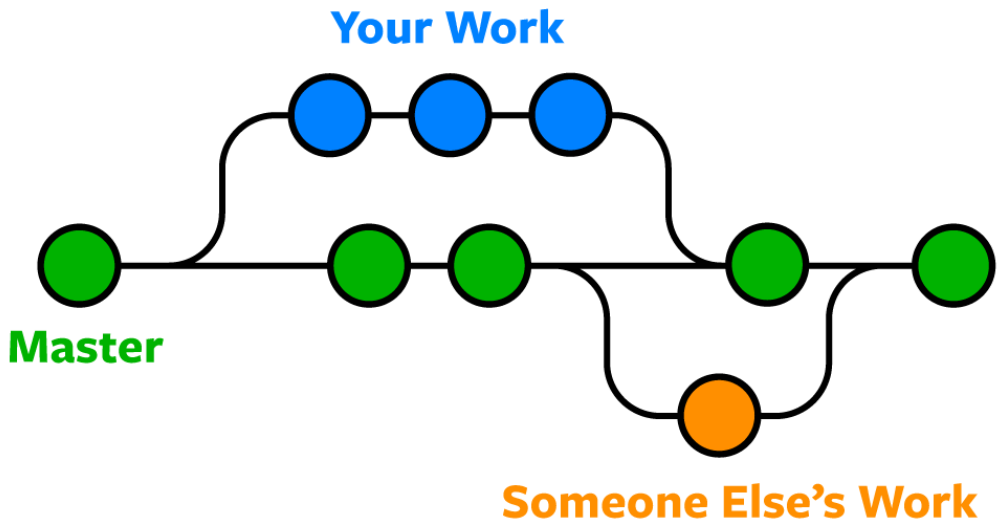
1. Routing in Computer Networks

- DAGs are used in routing algorithms to represent dependencies and avoid cyclic routes.
- They ensure efficient data packet traversal without looping indefinitely.
- Example: Link-state routing protocols like OSPF (Open Shortest Path First) use DAGs to compute the shortest paths to all nodes in the network.



2. Version Control Systems (VCS)

- DAGs manage commits and track the history of changes.
- Each commit is a node, and edges represent parent-child relationships between commits.
- Efficient history tracking.
- Easy conflict resolution during merges.
- Example: GIt



3. Compiler Design

- DAGs are used in compilers to represent expressions or instructions in intermediate code. This structure helps optimize redundant calculations.
- Example:
  - In syntax-directed translation, DAGs represent expressions to eliminate duplicate subexpressions.
  - In dependency graphs, they ensure that instructions are executed in the correct order without circular dependencies.

4. Job Scheduling

- DAGs represent tasks and their dependencies. Each node is a task, and edges indicate that one task depends on another.
- Example: In task scheduling for parallel computing or project management (e.g., critical path method), DAGs ensure that tasks are executed in a valid sequence without violating dependencies.

5. Data Processing framework

- DAGs model data workflows, showing how data flows between operations or transformations. They help in defining the sequence and dependencies between operations.
- Example:

  Apache Spark uses DAGs for processing large-scale data workflows.

  Pipelines in ETL (Extract, Transform, Load) systems are structured as DAGs to ensure proper data transformation order.

6. Genealogy and family tree

- Genealogy charts often use DAGs to represent family relationships. Nodes represent individuals, and directed edges denote parent-child relationships.
- DAGs are suitable because, in most genealogical systems, cycles (e.g., one person being their own ancestor) are not allowed.
- Example: Software like GEDCOM uses DAGs to model family trees.

7. Citation Graphs

- Citation graphs represent research papers as nodes, and directed edges indicate that one paper cites another.
- Since a paper cannot cite itself or any of its future descendants, the graph remains acyclic.
- Example: Google Scholar and other academic databases use DAGs to analyze citation patterns and determine influential papers.

Any Queries?

Thank You So Much!