1. Introduction

This report details the simulation of four major routing protocols (RIP, OSPF, BGP, and IS-IS) in Python. The objective was to implement the core algorithms of each protocol to understand their behavior, convergence properties, and operational differences. The simulations use graph data structures to represent network topologies and implement algorithms like Bellman-Ford (for RIP) and Dijkstra (for OSPF/IS-IS) to build routing tables.

2. Network Topologies and Simulation Setup

Different network topologies were used to best demonstrate the characteristics of each protocol.

- **RIP:** A simple, 5-node line/mesh graph was used. All link costs were set to 1, as RIP's metric is **hop count**.
- **OSPF & IS-IS:** A more complex 6-node graph with varying **link costs** (metrics) was used. This demonstrates the "shortest path" calculation based on cost (bandwidth, latency, etc.) rather than just hops.
- **BGP:** A 4-node **AS-level topology** was used, where each node represents an entire Autonomous System (e.g., an ISP). The simulation focused on exchanging **AS-Path** information.

All simulations were built using standard Python libraries, with `networkx` and `matplotlib` used for graph representation and visualization.

3. Routing Table Snapshots & Observations

The Python scripts generate final, converged routing tables for each router in the simulation. The user can run the `.py` files to see the convergence process and final outputs.

Part 1: RIP (Routing Information Protocol)

- **Algorithm:** Bellman-Ford (Distance Vector)
- **Observation:** The simulation shows RIP converging in "rounds." In each round, routers exchange their entire distance vectors. The final tables show paths based purely on the minimum number of hops. For example, in the `A-B-C-D` topology, the path from A to D converges to have a cost of 3 via next-hop B. The periodic, full-table updates demonstrate a high message overhead, and the round-by-round convergence is visibly slower than a link-state protocol.

Part 2: OSPF (Open Shortest Path First)

- **Algorithm:** Dijkstra (Link-State)
- **Observation:** The simulation first assumes a converged link-state database (from LSA flooding). Each router then runs Dijkstra's algorithm to build its own Shortest Path Tree (SPT). The simulation visualizes each router's unique SPT. The resulting routing table shows the first hop along the lowest-cost path. For example, a path `A-D-C` (cost 4) might be preferred over `A-C` (cost 5). Convergence is event-driven (on link change) and much faster than RIP.

Part 3: BGP (Border Gateway Protocol)

- **Algorithm:** Path Vector
- **Observation:** The simulation models AS-to-AS communication. The key metric is the **AS-Path length**. The simulation shows routes being advertised and routers selecting the route with the shortest AS-Path. Crucially, the **loop prevention** mechanism was implemented: if an AS receives a route advertisement that contains its own AS number, it discards the route. The final tables show the chosen path and the "next-hop AS" to reach a destination prefix.

Part 4: IS-IS (Intermediate System to Intermediate System)

- **Algorithm:** Dijkstra (Link-State)
- **Observation:** As expected, the algorithmic behavior of IS-IS is identical to OSPF. Given the same network topology and link costs, it produces the exact same routing tables. The primary differences are not in the core SPF algorithm but in their transport (Layer 2 vs. IP) and deployment history (Enterprise vs. Service Provider). The simulation confirms that both are highly efficient link-state protocols.

4. Protocol Comparison

Feature	**RIP (Distance Vector)**	**OSPF / IS-IS (Link-State)**	**BGP (Path Vector)**
Algorithm	Bellman-Ford	Dijkstra (SPF)	Path Selection (Policy)
Metric	Hop Count	Link Cost (Bandwidth, etc.)	AS-Path Length & Policy
View of Network	Only knows neighbors' tables.	Has a full map of the entire area.	Knows the AS-path to prefixes.
Convergence	Slow, periodic updates. Prone to loops.	Fast, event-driven.	Slow, based on policy.
Overhead	High (broadcasts full table)	Low (sends small updates)	Moderate (sends path updates)

Scalability	Poor (max 15 hops)	Good (using Areas)	Extremely High (The Internet)
Use Case	Small, simple networks (rarely used)	Internal (IGP) for Enterprise/ISP	External (EGP) between ASes

5. Conclusions

This lab successfully demonstrated the fundamental differences between the major routing protocol classes.

- **Distance Vector (RIP)** is simple but inefficient and slow.
- **Link-State (OSPF/IS-IS)** is complex internally but provides very fast convergence and optimal paths, making it ideal for large internal networks.
- **Path Vector (BGP)** is not concerned with link-level optimal paths but with *policy* and *reachability* at a global, inter-domain scale.

The simulations in Python provided a clear, hands-on understanding of how routing tables are built and maintained, and why different protocols are needed for different network scales and purposes.