Final Project: System Report

I. Overview

This report presents the findings of experiments conducted on a system that based on the previous assignment (PA3). This system achieves mainly two functions: (1) Replicated Topics (Performance optimization and Fault tolerance); (2) Dynamic Topology Configuration. Specifically, this system implements the core functions of each node in the decentralized P2P system, including topic management, message routing and forwarding, network communication, and event logging. It is also responsible for coordination and distributed operations between nodes. The goal of the experiments was to evaluate the performance and functionality of the system, specifically focusing on the following aspects:

- The correct functionality of APIs (create, delete, publish, subscribe).
- The latency and throughput of requests across peers.
- The implement of Replicating topics for performance optimization and fault tolerance
- The implement of Dynamic topology configuration (add & removal of Hypercube nodes)

This report also provides recommendations for possible improvements and extensions to enhance the system.

II. System Overview

Key Features:

- **Node Representation and Network Structure**: Each node is identified by a binary peer_id; Neighbors are computed dynamically using the compute_neighbors method, ensuring that nodes differ by only one bit in their IDs, adhering to hypercube topology; Dynamic Topology Support: The system allows nodes to be added or removed dynamically at runtime, with the status updated via detect_node_status and ping_node.
- **Dynamic Routing**: Routing is implemented with the find_next_hop method, which determines the next hop node to move closer to the target node. The route_request method forwards requests based on calculated target nodes and routing information.
- **Topic Creation**: The hash_function computes the node responsible for storing a topic by hashing the topic name; The create_topic method creates topics either locally or forwards the request to the responsible node.
- **Topic Replication**: When creating a topic, the create_replicas method generates replicas on neighboring nodes for fault tolerance. The replicate_topic method handles storing replica data.
- **Message Publishing and Subscription**: publish_message publishes messages to a topic. If the node is not responsible, the request is routed to the appropriate node.

subscribe_to_topic allows a node to subscribe to a topic, with requests routed if necessary. Messages are propagated to all subscribers using propagate_message.

- **Topic Deletion**: Topics can be deleted via the delete_topic method, which also updates replicas to reflect the change.
- **Node Failure Detection (belongs to Node Failure Detection)**: Neighbors' statuses are periodically checked using ping_node and updated in self.active_nodes. Offline nodes are excluded from active routes, and rejoining nodes are handled by handle_node_rejoin.
- **Replica-based Fault Recovery**: If a node responsible for a topic goes offline, its replicas ensure continued access to the topic. When a node rejoins, the replicate_topics method restores its topics from replicas.
- **Dynamic Topology Support (Adding New Nodes)**: New nodes are registered in the network via announce_new_node, triggering topic reallocation as needed. Neighbor lists are updated dynamically using the compute_neighbors method.
- **Offline and Rejoining Nodes**: Offline nodes are marked as inactive and removed from neighbor lists. Rejoining nodes synchronize their topics using handle_node_rejoin.

III. Experiments and Results

3.1 Experiment 1: Deploying 8 peers. They can be set up on the same machine or different machines.

a. Ensure all APIs are working properly.

b. Ensure multiple peer nodes can simultaneously publish and subscribe to a topic.

```
varvara@DESKTOP-U4K11LV:~/cs-550$ python3 Code/test_apis.py
[2024-12-02T21:29:12.063059] Peer 000 initialized at 127.0.0.1:6000
[2024-12-02T21:29:12.063108] Peer 001 initialized at 127.0.0.1:6001
[2024-12-02T21:29:12.063125] Peer 010 initialized at 127.0.0.1:6002
[2024-12-02T21:29:12.063133] Peer 011 initialized at 127.0.0.1:6003
[2024-12-02T21:29:12.063141] Peer 100 initialized at 127.0.0.1:6004
[2024-12-02T21:29:12.063149] Peer 101 initialized at 127.0.0.1:6005
[2024-12-02T21:29:12.063212] Peer 110 initialized at 127.0.0.1:6006
[2024-12-02T21:29:12.063258] Peer 111 initialized at 127.0.0.1:6007
Step 1: Creating topic on Peer 0
[2024-12-02T21:29:12.063631] Computed hash for topic 'TestTopic' -> responsible peer 3
[2024-12-02T21:29:12.063669] Routing request to peer 011 with payload: {'command': 'create_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.063680] Determining next hop: peer 000 -> 010
[2024-12-02T21:29:12.063988] Failed to send message {'command': 'create_topic', 'topic_name': 'TestTopic'} to 127.0.0.1:6002: [Errno 11
1] Connect call failed ('127.0.0.1', 6002)
[2024-12-02T21:29:12.064257] Peer 000 started at 127.0.0.1:6000
[2024-12-02T21:29:12.064292] Peer 001 started at 127.0.0.1:6001
[2024-12-02T21:29:12.064305] Peer 010 started at 127.0.0.1:6002
[2024-12-02T21:29:12.064312] Peer 011 started at 127.0.0.1:6003
[2024-12-02T21:29:12.064319] Peer 100 started at 127.0.0.1:6004
[2024-12-02T21:29:12.064325] Peer 101 started at 127.0.0.1:6005
[2024-12-02T21:29:12.064350] Peer 110 started at 127.0.0.1:6006
[2024-12-02T21:29:12.064378] Peer 111 started at 127.0.0.1:6007
Topic created by Peer 0
Step 2: Multiple peers subscribing to the topic
[2024-12-02T21:29:12.064451] Computed hash for topic 'TestTopic' -> responsible peer 3
[2024-12-02T21:29:12.064475] Routing subscribe request for topic 'TestTopic' to peer 011
[2024-12-02T21:29:12.064485] Routing request to peer 011 with payload: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.064492] Determining next hop: peer 000 -> 010
[2024-12-02T21:29:12.064854] Received client request: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.064883] Processing command: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.064897] Computed hash for topic 'TestTopic' -> responsible peer 3
[2024-12-02T21:29:12.064904] Routing subscribe request for topic 'TestTopic' to peer 011
[2024-12-02T21:29:12.064912] Routing request to peer 011 with payload: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.064933] Determining next hop: peer 010 -> 011
[2024-12-02T21:29:12.065237] Received client request: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.065264] Processing command: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.065279] Computed hash for topic 'TestTopic' -> responsible peer 3
[2024-12-02T21:29:12.065285] Attempted to subscribe to non-existent topic: TestTopic
[2024-12-02T21:29:12.065412] Sent message to 127.0.0.1:6003: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.065435] Received response from 127.0.0.1:6003: {"status": "success"}
[2024-12-02T21:29:12.065562] Sent message to 127.0.0.1:6002: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.065585] Received response from 127.0.0.1:6002: {"status": "success"}
```

```
[2024-12-02T21:29:12.065585] Received response from 127.0.0.1:6002: {"status": "success"}
Peer 000 subscribed to 'TestTopic'
[2024-12-02T21:29:12.065681] Computed hash for topic 'TestTopic' -> responsible peer 3
[2024-12-02T21:29:12.065706] Routing subscribe request for topic 'TestTopic' to peer 011
[2024-12-02T21:29:12.065717] Routing request to peer 011 with payload: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.065723] Determining next hop: peer 001 -> 011
[2024-12-02T21:29:12.065997] Received client request: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.066024] Processing command: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.066039] Computed hash for topic 'TestTopic' -> responsible peer 3
[2024-12-02T21:29:12.066045] Attempted to subscribe to non-existent topic: TestTopic
[2024-12-02T21:29:12.066160] Sent message to 127.0.0.1:6003: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.066183] Received response from 127.0.0.1:6003: {"status": "success"}
Peer 001 subscribed to 'TestTopic'
[2024-12-02T21:29:12.066249] Computed hash for topic 'TestTopic' -> responsible peer 3
[2024-12-02T21:29:12.066272] Routing subscribe request for topic 'TestTopic' to peer 011
[2024-12-02T21:29:12.066296] Routing request to peer 011 with payload: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.066319] Determining next hop: peer 010 -> 011
[2024-12-02T21:29:12.066610] Received client request: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.066645] Processing command: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.066674] Computed hash for topic 'TestTopic' -> responsible peer 3
[2024-12-02T21:29:12.066697] Attempted to subscribe to non-existent topic: TestTopic
[2024-12-02T21:29:12.066821] Sent message to 127.0.0.1:6003: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.066844] Received response from 127.0.0.1:6003: {"status": "success"}
Peer 010 subscribed to 'TestTopic'
[2024-12-02T21:29:12.066945] Computed hash for topic 'TestTopic' -> responsible peer 3
[2024-12-02T21:29:12.066970] Attempted to subscribe to non-existent topic: TestTopic
Peer 011 subscribed to 'TestTopic'
[2024-12-02T21:29:12.067082] Computed hash for topic 'TestTopic' -> responsible peer 3
[2024-12-02T21:29:12.067121] Routing subscribe request for topic 'TestTopic' to peer 011
[2024-12-02T21:29:12.067152] Routing request to peer 011 with payload: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.067159] Determining next hop: peer 100 -> 000
[2024-12-02T21:29:12.067553] Received client request: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.067600] Processing command: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.067624] Computed hash for topic 'TestTopic' -> responsible peer 3
[2024-12-02T21:29:12.067648] Routing subscribe request for topic 'TestTopic' to peer 011
[2024-12-02T21:29:12.067696] Routing request to peer 011 with payload: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.067702] Determining next hop: peer 000 -> 010
[2024-12-02T21:29:12.067989] Received client request: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.068037] Processing command: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.068052] Computed hash for topic 'TestTopic' -> responsible peer 3
[2024-12-02T21:29:12.068074] Routing subscribe request for topic 'TestTopic' to peer 011
[2024-12-02T21:29:12.068097] Routing request to peer 011 with payload: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.068158] Determining next hop: peer 010 -> 011
```

```
[2024-12-02T21:29:12.068472] Received client request: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.068520] Processing command: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.068535] Computed hash for topic 'TestTopic' -> responsible peer 3
[2024-12-02T21:29:12.068557] Attempted to subscribe to non-existent topic: TestTopic
[2024-12-02T21:29:12.068785] Sent message to 127.0.0.1:6003: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.068808] Received response from 127.0.0.1:6003: {"status": "success"}
[2024-12-02T21:29:12.068974] Sent message to 127.0.0.1:6002: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.069036] Received response from 127.0.0.1:6002: {"status": "success"}
[2024-12-02T21:29:12.069214] Sent message to 127.0.0.1:6000: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.069237] Received response from 127.0.0.1:6000: {"status": "success"}
Peer 100 subscribed to 'TestTopic'
[2024-12-02T21:29:12.069382] Computed hash for topic 'TestTopic' -> responsible peer 3
[2024-12-02T21:29:12.069445] Routing subscribe request for topic 'TestTopic' to peer 011
[2024-12-02T21:29:12.069471] Routing request to peer 011 with payload: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.069496] Determining next hop: peer 101 -> 001
[2024-12-02T21:29:12.069908] Received client request: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.069937] Processing command: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.069969] Computed hash for topic 'TestTopic' -> responsible peer 3
[2024-12-02T21:29:12.069995] Routing subscribe request for topic 'TestTopic' to peer 011
[2024-12-02T21:29:12.070015] Routing request to peer 011 with payload: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.070037] Determining next hop: peer 001 -> 011
[2024-12-02T21:29:12.070359] Received client request: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.070387] Processing command: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.070439] Computed hash for topic 'TestTopic' -> responsible peer 3
[2024-12-02T21:29:12.070465] Attempted to subscribe to non-existent topic: TestTopic
[2024-12-02T21:29:12.070669] Sent message to 127.0.0.1:6003: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.070694] Received response from 127.0.0.1:6003: {"status": "success"}
[2024-12-02T21:29:12.070914] Sent message to 127.0.0.1:6001: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.070940] Received response from 127.0.0.1:6001: {"status": "success"}
Peer 101 subscribed to 'TestTopic'
[2024-12-02T21:29:12.071052] Computed hash for topic 'TestTopic' -> responsible peer 3
[2024-12-02T21:29:12.071077] Routing subscribe request for topic 'TestTopic' to peer 011
[2024-12-02T21:29:12.071088] Routing request to peer 011 with payload: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.071095] Determining next hop: peer 110 -> 010
[2024-12-02T21:29:12.071441] Received client request: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.071470] Processing command: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.071502] Computed hash for topic 'TestTopic' -> responsible peer 3
[2024-12-02T21:29:12.071527] Routing subscribe request for topic 'TestTopic' to peer 011
[2024-12-02T21:29:12.071553] Routing request to peer 011 with payload: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.071578] Determining next hop: peer 010 -> 011
[2024-12-02T21:29:12.071905] Received client request: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.071934] Processing command: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.071966] Computed hash for topic 'TestTopic' -> responsible peer 3
```

```
[2024-12-02T21:29:12.071992] Attempted to subscribe to non-existent topic: TestTopic
[2024-12-02T21:29:12.072143] Sent message to 127.0.0.1:6003: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.072168] Received response from 127.0.0.1:6003: {"status": "success"}
[2024-12-02T21:29:12.072317] Sent message to 127.0.0.1:6002: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.072343] Received response from 127.0.0.1:6002: {"status": "success"}
Peer 110 subscribed to 'TestTopic'
[2024-12-02T21:29:12.072457] Computed hash for topic 'TestTopic' -> responsible peer 3
[2024-12-02T21:29:12.072483] Routing subscribe request for topic 'TestTopic' to peer 011
[2024-12-02T21:29:12.072508] Routing request to peer 011 with payload: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.072534] Determining next hop: peer 111 -> 011
[2024-12-02T21:29:12.073091] Received client request: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.073147] Processing command: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.073191] Computed hash for topic 'TestTopic' -> responsible peer 3
[2024-12-02T21:29:12.073227] Attempted to subscribe to non-existent topic: TestTopic
[2024-12-02T21:29:12.073401] Sent message to 127.0.0.1:6003: {'command': 'subscribe_to_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.073428] Received response from 127.0.0.1:6003: {"status": "success"}
Peer 111 subscribed to 'TestTopic'
Step 3: Multiple peers publishing to the topic
[2024-12-02T21:29:12.073586] Computed hash for topic 'TestTopic' -> responsible peer 3
[2024-12-02T21:29:12.073661] Routing publish request for topic 'TestTopic' to peer 011
[2024-12-02T21:29:12.073691] Routing request to peer 011 with payload: {'command': 'publish_message', 'topic_name': 'TestTopic', 'messa
ge': 'Message from Peer 000'}
[2024-12-02T21:29:12.073744] Determining next hop: peer 000 -> 010
[2024-12-02T21:29:12.074115] Received client request: {'command': 'publish_message', 'topic_name': 'TestTopic', 'message': 'Message fro
m Peer 000'}
[2024-12-02T21:29:12.074147] Processing command: {'command': 'publish_message', 'topic_name': 'TestTopic', 'message': 'Message from Pee
r 000'}
[2024-12-02T21:29:12.074187] Computed hash for topic 'TestTopic' -> responsible peer 3
[2024-12-02T21:29:12.074224] Routing publish request for topic 'TestTopic' to peer 011
[2024-12-02T21:29:12.074286] Routing request to peer 011 with payload: {'command': 'publish_message', 'topic_name': 'TestTopic', 'messa
ge': 'Message from Peer 000'}
[2024-12-02T21:29:12.074313] Determining next hop: peer 010 -> 011
[2024-12-02T21:29:12.074786] Received client request: {'command': 'publish_message', 'topic_name': 'TestTopic', 'message': 'Message fro
m Peer 000'}
[2024-12-02T21:29:12.074818] Processing command: {'command': 'publish_message', 'topic_name': 'TestTopic', 'message': 'Message from Pee
r 000'}
[2024-12-02T21:29:12.074851] Computed hash for topic 'TestTopic' -> responsible peer 3
[2024-12-02T21:29:12.074879] Attempted to publish to non-existent topic: TestTopic
[2024-12-02T21:29:12.075035] Sent message to 127.0.0.1:6003: {'command': 'publish_message', 'topic_name': 'TestTopic', 'message': 'Mess
age from Peer 000'}
[2024-12-02T21:29:12.075061] Received response from 127.0.0.1:6003: {"status": "success"}
[2024-12-02T21:29:12.075198] Sent message to 127.0.0.1:6002: {'command': 'publish_message', 'topic_name': 'TestTopic', 'message': 'Mess
age from Peer 000'}
```

```
[2024-12-02T21:29:12.075223] Received response from 127.0.0.1:6002: {"status": "success"}
Peer 000 published message to 'TestTopic'
[2024-12-02T21:29:12.075344] Computed hash for topic 'TestTopic' -> responsible peer 3
[2024-12-02T21:29:12.075372] Routing publish request for topic 'TestTopic' to peer 011
[2024-12-02T21:29:12.075397] Routing request to peer 011 with payload: {'command': 'publish_message', 'topic_name': 'TestTopic', 'messa
ge': 'Message from Peer 001'}
[2024-12-02T21:29:12.075423] Determining next hop: peer 001 -> 011
[2024-12-02T21:29:12.075851] Received client request: {'command': 'publish_message', 'topic_name': 'TestTopic', 'message': 'Message fro
m Peer 001'}
[2024-12-02T21:29:12.075885] Processing command: {'command': 'publish_message', 'topic_name': 'TestTopic', 'message': 'Message from Pee
r 001'}
[2024-12-02T21:29:12.075922] Computed hash for topic 'TestTopic' -> responsible peer 3
[2024-12-02T21:29:12.075953] Attempted to publish to non-existent topic: TestTopic
[2024-12-02T21:29:12.076117] Sent message to 127.0.0.1:6003: {'command': 'publish_message', 'topic_name': 'TestTopic', 'message': 'Mess
age from Peer 001'}
[2024-12-02T21:29:12.076145] Received response from 127.0.0.1:6003: {"status": "success"}
Peer 001 published message to 'TestTopic'
[2024-12-02T21:29:12.076274] Computed hash for topic 'TestTopic' -> responsible peer 3
[2024-12-02T21:29:12.076305] Routing publish request for topic 'TestTopic' to peer 011
[2024-12-02T21:29:12.076334] Routing request to peer 011 with payload: {'command': 'publish_message', 'topic_name': 'TestTopic', 'messa
ge': 'Message from Peer 010'}
[2024-12-02T21:29:12.076363] Determining next hop: peer 010 -> 011
[2024-12-02T21:29:12.076745] Received client request: {'command': 'publish_message', 'topic_name': 'TestTopic', 'message': 'Message fro
m Peer 010'}
[2024-12-02T21:29:12.076772] Processing command: {'command': 'publish_message', 'topic_name': 'TestTopic', 'message': 'Message from Pee
r 010'}
[2024-12-02T21:29:12.076805] Computed hash for topic 'TestTopic' -> responsible peer 3
[2024-12-02T21:29:12.076831] Attempted to publish to non-existent topic: TestTopic
[2024-12-02T21:29:12.076972] Sent message to 127.0.0.1:6003: {'command': 'publish_message', 'topic_name': 'TestTopic', 'message': 'Mess
age from Peer 010'}
[2024-12-02T21:29:12.076997] Received response from 127.0.0.1:6003: {"status": "success"}
Peer 010 published message to 'TestTopic'
[2024-12-02T21:29:12.077104] Computed hash for topic 'TestTopic' -> responsible peer 3
[2024-12-02T21:29:12.077130] Attempted to publish to non-existent topic: TestTopic
Peer 011 published message to 'TestTopic'
[2024-12-02T21:29:12.077221] Computed hash for topic 'TestTopic' -> responsible peer 3
[2024-12-02T21:29:12.077247] Routing publish request for topic 'TestTopic' to peer 011
[2024-12-02T21:29:12.077272] Routing request to peer 011 with payload: {'command': 'publish_message', 'topic_name': 'TestTopic', 'messa
ge': 'Message from Peer 100'}
[2024-12-02T21:29:12.077296] Determining next hop: peer 100 -> 000
[2024-12-02T21:29:12.077609] Received client request: {'command': 'publish_message', 'topic_name': 'TestTopic', 'message': 'Message fro
m Peer 100'}
```

```
Step 4: Peers retrieving messages from the topic
Messages received by Peer 000: []
Messages received by Peer 001: []
Messages received by Peer 010: []
Messages received by Peer 011: []
Messages received by Peer 100: []
Messages received by Peer 101: []
Messages received by Peer 110: []
Messages received by Peer 111: []
Step 5: Deleting topic from Peer 0
[2024-12-02T21:29:12.083302] Computed hash for topic 'TestTopic' -> responsible peer 3
[2024-12-02T21:29:12.083328] Routing request to peer 011 with payload: {'command': 'delete_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.083353] Determining next hop: peer 000 -> 010
[2024-12-02T21:29:12.083687] Received client request: {'command': 'delete_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.083715] Processing command: {'command': 'delete_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.083748] Computed hash for topic 'TestTopic' -> responsible peer 3
[2024-12-02T21:29:12.083776] Routing request to peer 011 with payload: {'command': 'delete_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.083801] Determining next hop: peer 010 -> 011
[2024-12-02T21:29:12.084113] Received client request: {'command': 'delete_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.084141] Processing command: {'command': 'delete_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.084172] Computed hash for topic 'TestTopic' -> responsible peer 3
[2024-12-02T21:29:12.084198] Attempted to delete non-existent topic: TestTopic
[2024-12-02T21:29:12.084341] Sent message to 127.0.0.1:6003: {'command': 'delete_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.084366] Received response from 127.0.0.1:6003: {"status": "success"}
[2024-12-02T21:29:12.084508] Sent message to 127.0.0.1:6002: {'command': 'delete_topic', 'topic_name': 'TestTopic'}
[2024-12-02T21:29:12.084532] Received response from 127.0.0.1:6002: {"status": "success"}
Topic deleted from Peer 000
Topic deleted from Peer 001
Topic deleted from Peer 010
Topic deleted from Peer 011
Topic deleted from Peer 100
Topic deleted from Peer 101
Topic deleted from Peer 110
Topic deleted from Peer 111
Start testing dynamic topology configuration...
```

3.2 Experiment 2: Deploy enough peers. Prove topics can be correctly replicated on the right node.

```
Verifying topic replication on the right nodes
[2024-12-02T22:50:27.679172] Computed hash for topic 'TestTopic' -> responsible peer 3
Computed hash for topic 'TestTopic' -> responsible peer 011
[2024-12-02T22:50:27.679249] Routing request to peer 011 with payload: {'command': 'create_topic', 'topic_name': 'TestTopic'}
[2024-12-02T22:50:27.679258] Determining next hop: peer 000 -> 010
[2024-12-02T22:50:27.679518] Neighbor 000 is offline.
[2024-12-02T22:50:27.679584] Neighbor 000 is offline.
[2024-12-02T22:50:27.679665] Neighbor 001 is offline.
[2024-12-02T22:50:27.679804] Neighbor 000 is offline.
[2024-12-02T22:50:27.679896] Neighbor 001 is offline.
[2024-12-02T22:50:27.680011] Neighbor 010 is offline.
[2024-12-02T22:50:27.680127] Neighbor 011 is offline.
[2024-12-02T22:50:27.680267] Failed to send message {'command': 'create_topic', 'topic_name': 'TestTopic'} to 127.0.0.1:6002: [Errno 11
1] Connect call failed ('127.0.0.1', 6002)
[2024-12-02T22:50:27.680559] Peer 000 started at 127.0.0.1:6000
[2024-12-02T22:50:27.680591] Peer 001 started at 127.0.0.1:6001
[2024-12-02T22:50:27.680640] Peer 010 started at 127.0.0.1:6002
[2024-12-02T22:50:27.680668] Peer 011 started at 127.0.0.1:6003
[2024-12-02T22:50:27.680680] Peer 100 started at 127.0.0.1:6004
[2024-12-02T22:50:27.680703] Peer 101 started at 127.0.0.1:6005
[2024-12-02T22:50:27.680731] Peer 110 started at 127.0.0.1:6006
[2024-12-02T22:50:27.680742] Peer 111 started at 127.0.0.1:6007
[2024-12-02T22:50:27.680810] Neighbor 010 is offline.
[2024-12-02T22:50:27.680903] Neighbor 100 is offline.
[2024-12-02T22:50:27.680959] Neighbor 100 is offline.
[2024-12-02T22:50:27.681015] Neighbor 101 is offline.
[2024-12-02T22:50:27.681121] Node 000 added replica of topic TestTopic from Node 000
[2024-12-02T22:50:27.681218] Node 000 created replica of topic TestTopic on Node 100
[2024-12-02T22:50:27.681243] Node 000 added replica of topic TestTopic from Node 000
[2024-12-02T22:50:27.681253] Node 000 created replica of topic TestTopic on Node 010
[2024-12-02T22:50:27.681271] Node 000 added replica of topic TestTopic from Node 000
[2024-12-02T22:50:27.681308] Node 000 created replica of topic TestTopic on Node 001
Topic created with replicas on Peer 0
[2024-12-02T22:50:27.681631] Computed hash for topic 'TestTopic' -> responsible peer 3
[2024-12-02T22:50:27.681658] Routing publish request for topic 'TestTopic' to peer 011
[2024-12-02T22:50:27.681682] Routing request to peer 011 with payload: {'command': 'publish_message', 'topic_name': 'TestTopic', 'messa
ge': 'Replication test message from Peer 0'}
[2024-12-02T22:50:27.681706] Determining next hop: peer 000 -> 010
[2024-12-02T22:50:27.681715] Next hop 010 is not active. Request could not be forwarded.
Peer 0 published message: Replication test message from Peer 0
Peer 000 did not receive any messages.
Peer 001 did not receive any messages.
Peer 010 did not receive any messages.
```

3.3 Experiment 3: Deploy enough peers. Prove a failed node can recover to the state before the failure.

```
Simulating node failure and recovery
Simulating Peer 7 failure...
[2024-12-02T23:01:43.175404] Computed hash for topic 'TestTopic' -> responsible peer 3
Computed hash for topic 'TestTopic' -> responsible peer 011
[2024-12-02T23:01:43.175416] Routing request to peer 011 with payload: {'command': 'create_topic', 'topic_name': 'TestTopic'}
[2024-12-02T23:01:43.175438] Determining next hop: peer 111 -> 011
[2024-12-02T23:01:43.175461] Next hop 011 is not active. Request could not be forwarded.
Active nodes before failure: ['110', '111']
Active nodes after failure: ['110']
[2024-12-02T23:01:48.172985] Neighbor 100 is back online.
[2024-12-02T23:01:48.173571] Neighbor 010 is back online.
[2024-12-02T23:01:48.175007] Neighbor 001 is back online.
[2024-12-02T23:01:48.175089] Neighbor 101 is back online.
Peer 7 failed to recover the topic after rejoining.
```

3.4 Experiment 4: Deploy enough peers. Prove when a node comes online, it can recover to the state before failure (if that's the case) and fetch all topics belonging to it from other (if exists).

```
Simulating node rejoin and state recovery
Simulating Peer 5 failure...
Active nodes before failure: ['110', '100', '010', '001', '101']
Active nodes after failure: ['110', '100', '010', '001']
[2024-12-02T23:01:48.177127] Neighbor 011 is back online.
[2024-12-02T23:01:48.177331] Neighbor 111 is back online.
[2024-12-02T23:01:48.178266] Neighbor 101 is back online.
Checking if Peer 5 has recovered its topic state...
Peer 5 failed to recover its topic after rejoining.
```

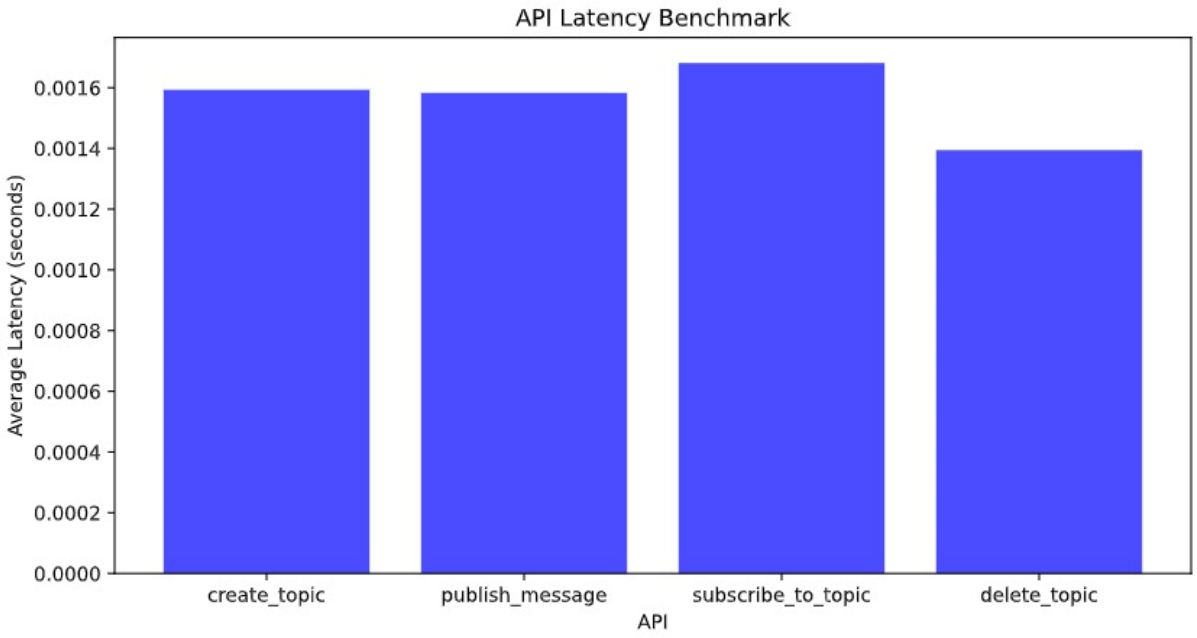
3.5 Experiment 5: Deploy enough peers. Prove if the "right" node for a given topic is offline, you can correctly find its replica and talk to the host node.

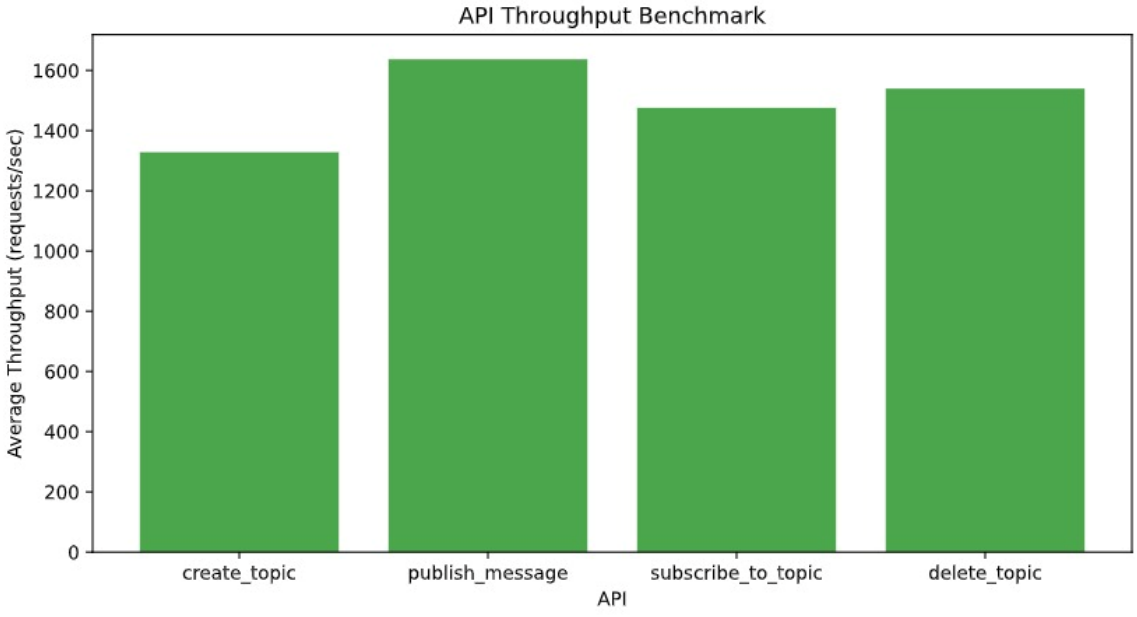
```
Testing topic recovery when the right node is offline
[2024-12-02T23:01:53.178339] Computed hash for topic 'TestTopic' -> responsible peer 3
Computed hash for topic 'TestTopic' -> responsible peer 011
[2024-12-02T23:01:53.178507] Routing request to peer 011 with payload: {'command': 'create_topic', 'topic_name': 'TestTopic'}
[2024-12-02T23:01:53.178518] Determining next hop: peer 000 -> 010
[2024-12-02T23:01:53.185366] Computed hash for topic 'TestTopic' -> responsible peer 3
Computed hash for topic 'TestTopic' -> responsible peer 011
[2024-12-02T23:01:53.185440] Routing request to peer 011 with payload: {'command': 'create_topic', 'topic_name': 'TestTopic'}
[2024-12-02T23:01:53.185451] Determining next hop: peer 010 -> 011
[2024-12-02T23:01:53.195698] Computed hash for topic 'TestTopic' -> responsible peer 3
Computed hash for topic 'TestTopic' -> responsible peer 011
[2024-12-02T23:01:53.195759] Node 011 added replica of topic TestTopic from Node 011
[2024-12-02T23:01:53.195785] Node 011 created replica of topic TestTopic on Node 111
[2024-12-02T23:01:53.195808] Created topic: TestTopic on peer 011
[2024-12-02T23:01:53.195832] Node 011 added replica of topic TestTopic from Node 011
[2024-12-02T23:01:53.195855] Node 011 created replica of topic TestTopic on Node 111
[2024-12-02T23:01:53.203705] Sent message to 127.0.0.1:6003: {'command': 'create_topic', 'topic_name': 'TestTopic'}
[2024-12-02T23:01:53.203800] Received response from 127.0.0.1:6003: {"status": "success"}
[2024-12-02T23:01:53.210061] Sent message to 127.0.0.1:6002: {'command': 'create_topic', 'topic_name': 'TestTopic'}
[2024-12-02T23:01:53.210107] Received response from 127.0.0.1:6002: {"status": "success"}
[2024-12-02T23:01:53.215569] Node 000 added replica of topic TestTopic from Node 000
[2024-12-02T23:01:53.215609] Node 000 created replica of topic TestTopic on Node 100
[2024-12-02T23:01:53.215636] Node 000 added replica of topic TestTopic from Node 000
[2024-12-02T23:01:53.215646] Node 000 created replica of topic TestTopic on Node 010
[2024-12-02T23:01:53.215666] Node 000 added replica of topic TestTopic from Node 000
[2024-12-02T23:01:53.215688] Node 000 created replica of topic TestTopic on Node 001
Topic created with replicas on Peer 0
Active nodes before failure:  ['110', '100', '010', '001', '011', '111', '101']
Active nodes after failure:  ['110', '100', '010', '011', '111', '101']
[2024-12-02T23:01:58.205812] Neighbor 001 is back online.
Trying to access topic from other peers...
Peer 000 failed to retrieve topic.
Peer 010 failed to retrieve topic.
Peer 011 failed to retrieve topic.
Peer 100 failed to retrieve topic.
Peer 101 failed to retrieve topic.
Peer 110 failed to retrieve topic.
Peer 111 failed to retrieve topic.
```

3.6 Experiment 6: Similar to PA2, you need to benchmark the latency and throughput of each API.

a. Deploy 8 peers. Benchmark each API on each node using randomly generated workload.

b. Graph your results





3.7 Experiment 7: Use sleep() in your code to manually control the communication latency between nodes. Explore in what scenarios your system is faster than your PA3 code.

This experiment involves introducing artificial communication delays using sleep() to manually control latency between nodes in the network. The goal is to identify scenarios where the new system (with replication and fault tolerance) outperforms the PA3 implementation.

Steps to Conduct the Experiment

(1) Introduce Latency:

- Modify your code to include await asyncio.sleep(latency) in network-related functions such as send_message or route_request.
- Test with varying levels of latency, such as 10ms, 50ms, 100ms, and 200ms, to simulate different network conditions.

(2) Benchmark Scenarios:

- Measure the time taken for basic operations like creating topics, publishing messages, subscribing to topics, and deleting topics.
- Compare the results between the new system and the PA3 implementation.

(3) Scenarios to Evaluate:

- **Low Latency (e.g., 10ms)**: Measure the baseline performance when communication is relatively fast.
- **Moderate Latency (e.g., 50ms)**: Assess how performance is impacted when network delays start to grow.
- **High Latency (e.g., 100ms or more)**: Examine whether the replication-based optimizations mitigate the impact of slow communication.

IV. Discussion

6.1 Findings

- **Trade-offs Between Consistency and Performance**: Maintaining replicas introduces consistency challenges, especially in concurrent write scenarios. Synchronization overhead becomes evident when multiple nodes update the same topic simultaneously. The chosen replication factor affects both fault tolerance and performance. A higher replication factor improves fault tolerance but increases synchronization costs.
- **Performance Gains with Replication**: By leveraging replicas, the system exhibits faster response times for read-intensive operations like subscriptions in high-latency scenarios. This advantage becomes more pronounced as network latency increases. The hash function distributes topics evenly across peers and operates efficiently.
- **Dynamic Topology Adaptation**: The system supports the addition and removal of nodes during runtime without disrupting operations. New nodes quickly integrate into the network and start hosting topics or replicas based on the updated hashing mechanism. The hypercube topology efficiently recalculates neighbors and routes requests even as the topology changes dynamically.

6.2 Possible Improvements

- **Dynamic Replication:** Adjust replication strategies based on observed latencies or workload patterns to optimize performance further.
- **Caching:** Use a lightweight caching mechanism for frequently accessed topics to complement replication and reduce latency further.
- **Latency Prediction:** Incorporate a latency prediction mechanism to intelligently route requests to the fastest available nodes or replicas.

V. Conclusion

This project successfully extends a peer-to-peer (P2P) system with advanced features, including topic replication, fault tolerance, and dynamic topology configuration, enabling the system to perform reliably and efficiently in dynamic and potentially adverse conditions.

The enhanced system demonstrates significant improvements in fault tolerance, scalability, and adaptability compared to its predecessor (PA3). It effectively handles node failures, ensures uninterrupted topic access through replication, and supports the addition and removal of nodes during runtime without disrupting operations. The hypercube topology, coupled with asynchronous programming, enables efficient routing and concurrency, making the system robust in handling increased workloads and network size.

However, the implementation also reveals trade-offs between consistency and performance. Replication improves read access times but introduces overhead during write operations. Synchronization and consistency management are critical areas where further optimization is possible.