Midterm Project – Chord
Part 3: File System Layer Implementation
Practices for Distributed Systems and Cloud Application Development

Part 2: Automatic scaling on AWS Cloud

```
# startup-script.py

my_node_ip = ...
existing_node_ip = ...

# Start file server
python3 -m uploadserver 5058 --directory /home/ec2-user/files

# Start Chord node
os.system("./chord {} 5057 &".format(my_node_ip))

# Join existing Chord system
my_chord_client = new_client(my_node_ip, 5057)
existing_chord_client = new_client(existing_node_ip, 5057)
my_chord_client.call("join", existing_chord_client.call("get_info"))
```

Part 2: Automatic scaling on AWS Cloud

- RPCs exposed by the provided chord binary
  - `get_info / create / join / find_successor / kill`
  - `Node get_predecessor()`
    - returns the predecessor of the node
- `Node get_successor(int i)`
  - `get_successor(0)` returns the immediate successor of the node
  - `get_successor(1)` returns the successor of `get_successor(0)`
  - the parameter `i` can only be 0, 1, or 2

Goal

- Migrate the data when new nodes are added
- Balance the load on each node by dividing files into smaller file chunks
- Implement file replication to ensure fault tolerance

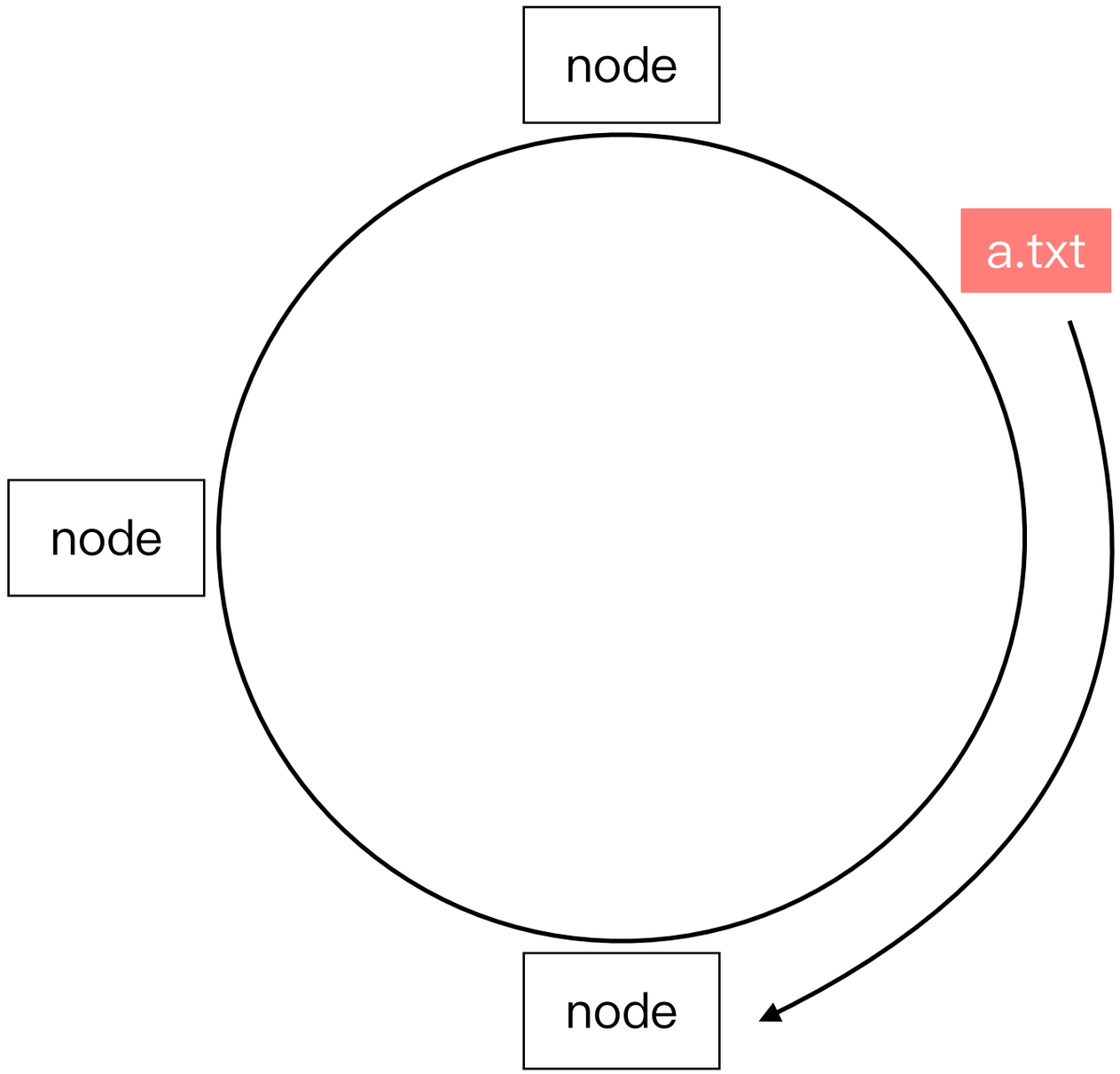
File System Layer

- You are free to use any method to implement your functionalities
    - You can use any programming language and existing tools
- You are suggested to implement your functionalities by adding new components

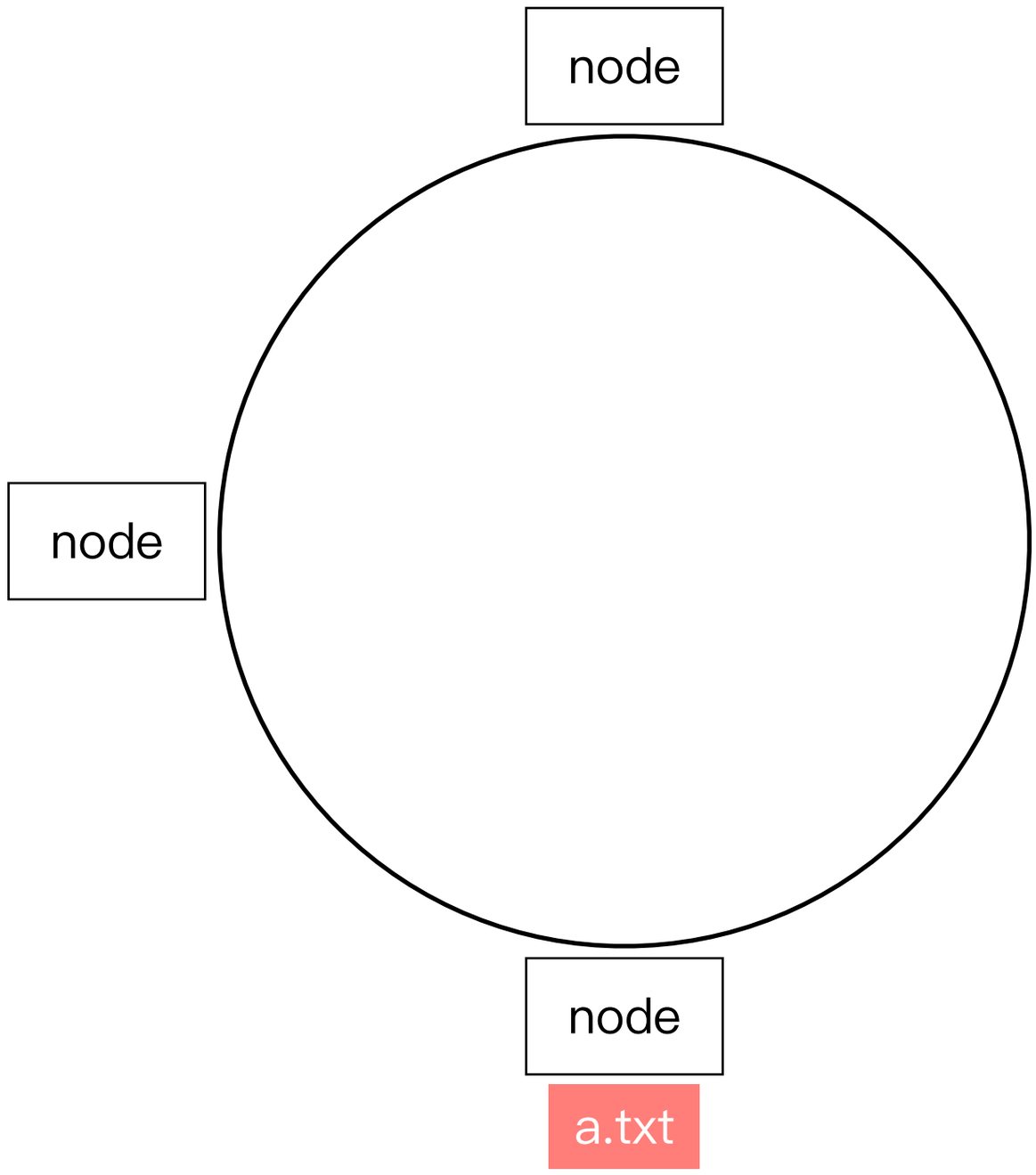
Grading Policy

- Data Migration: 33%
- File Chunks: 33%
- Replication: 33%
- Bonus: at most 33%

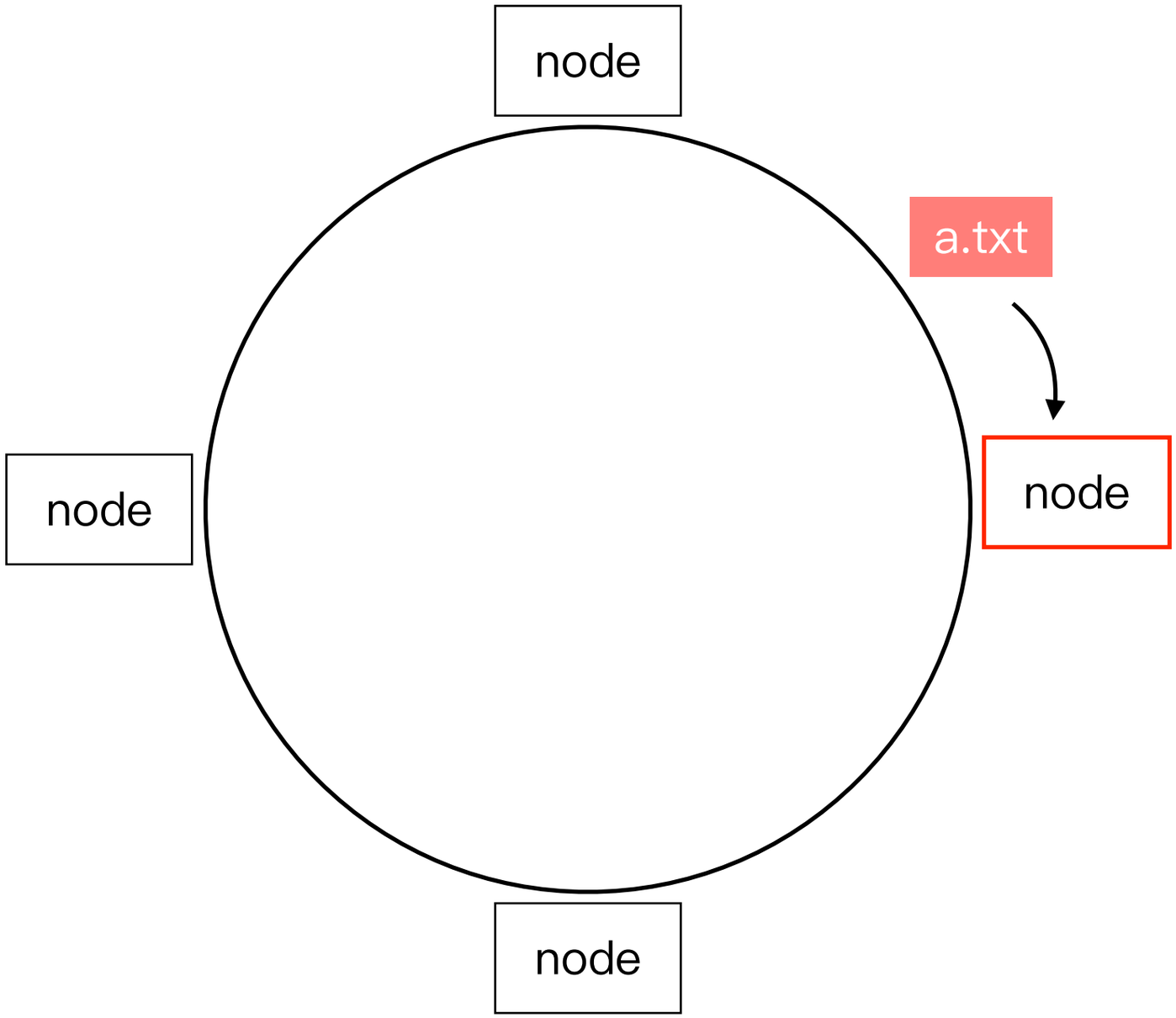
Correctness – Data Migration (33%)



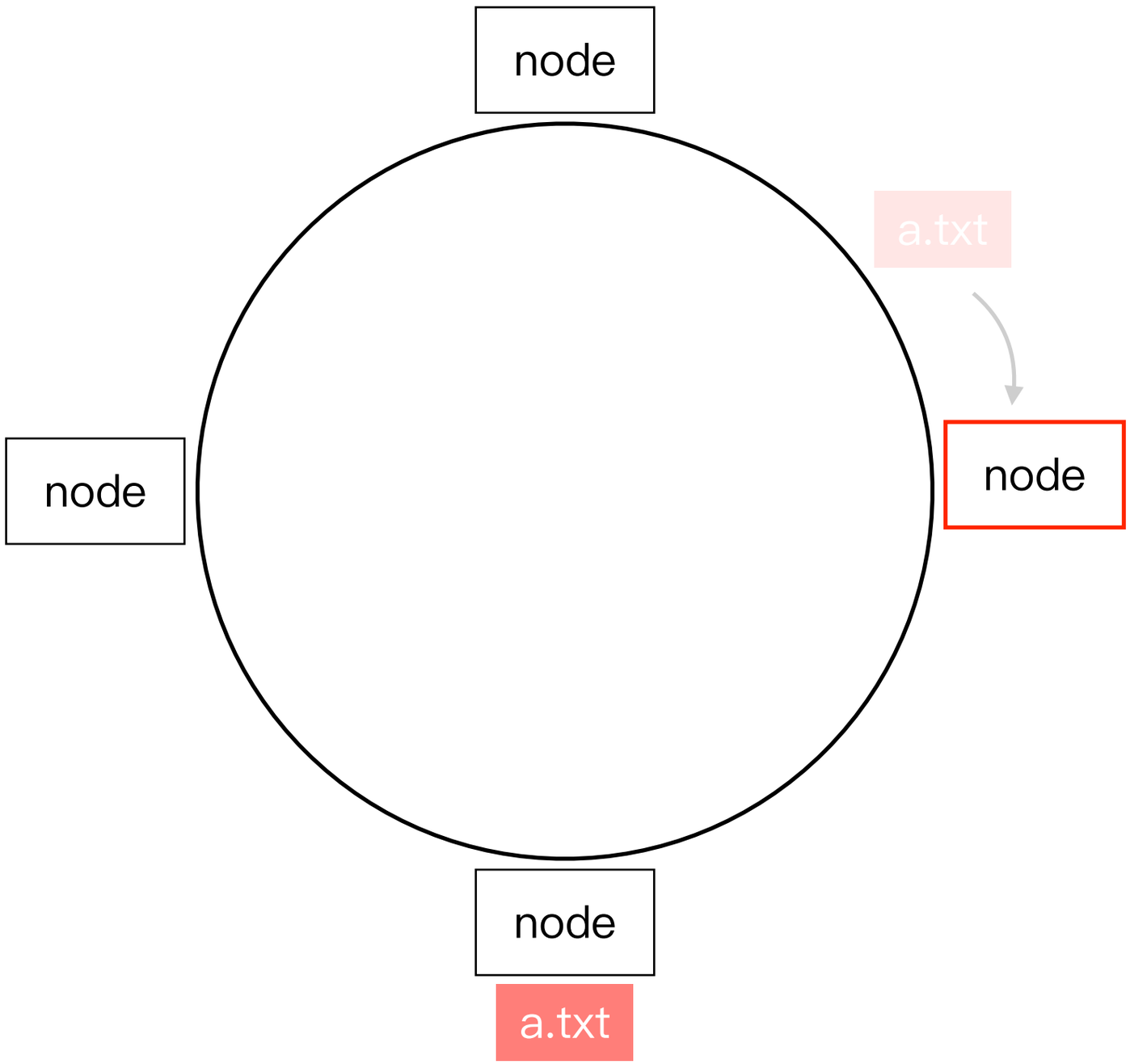
Correctness – Data Migration (33%)



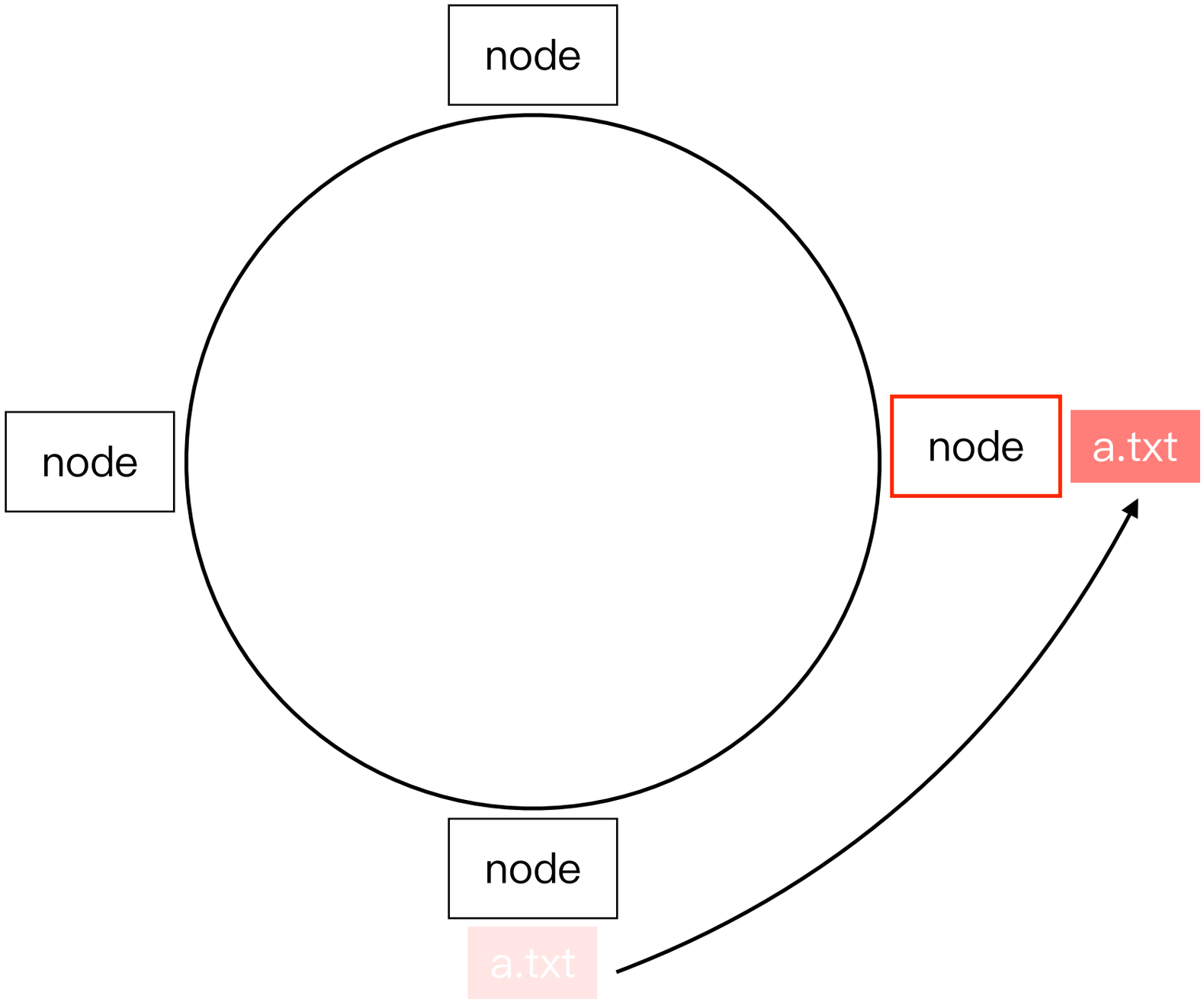
Correctness – Data Migration (33%)



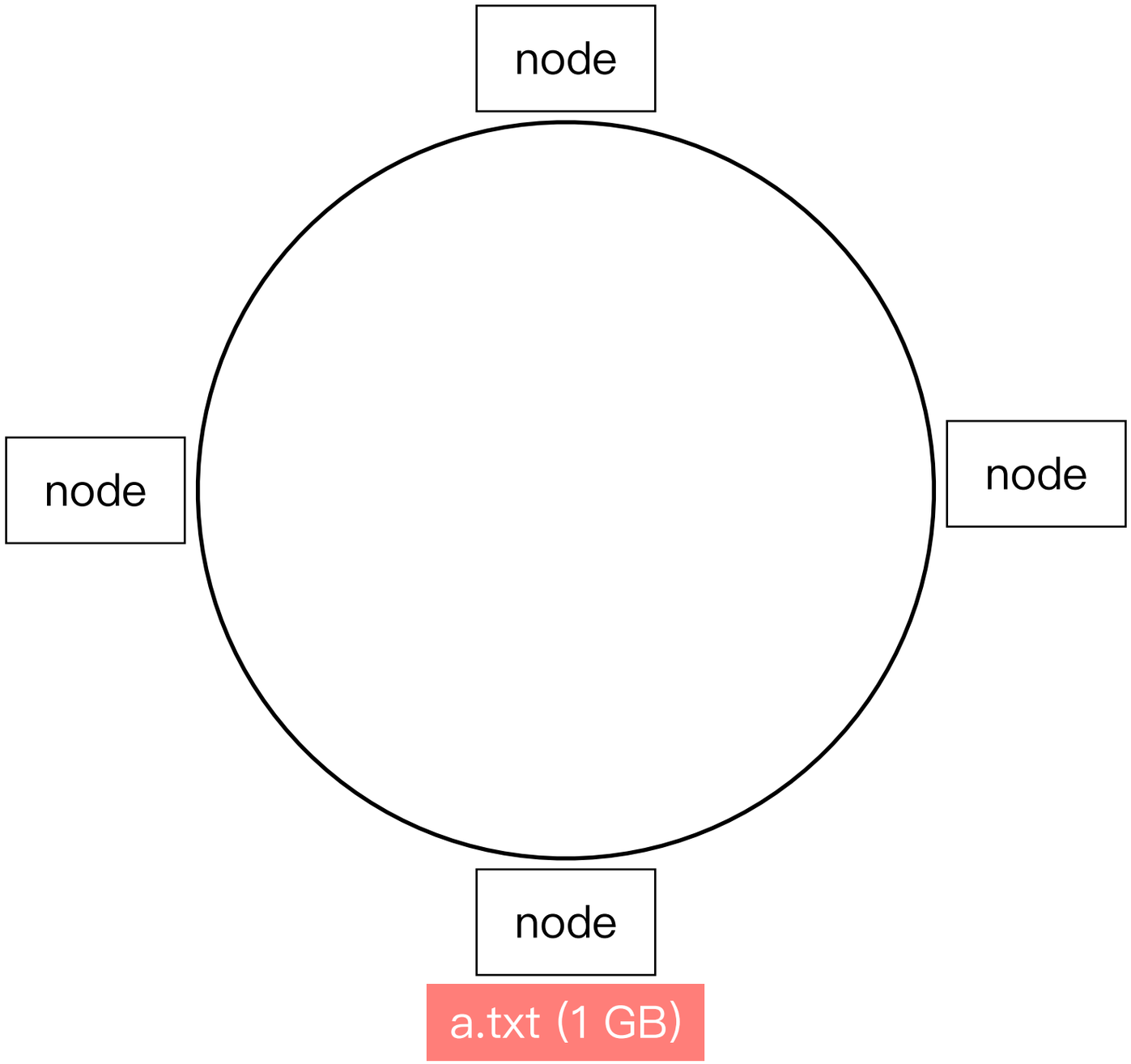
Correctness – Data Migration (33%)



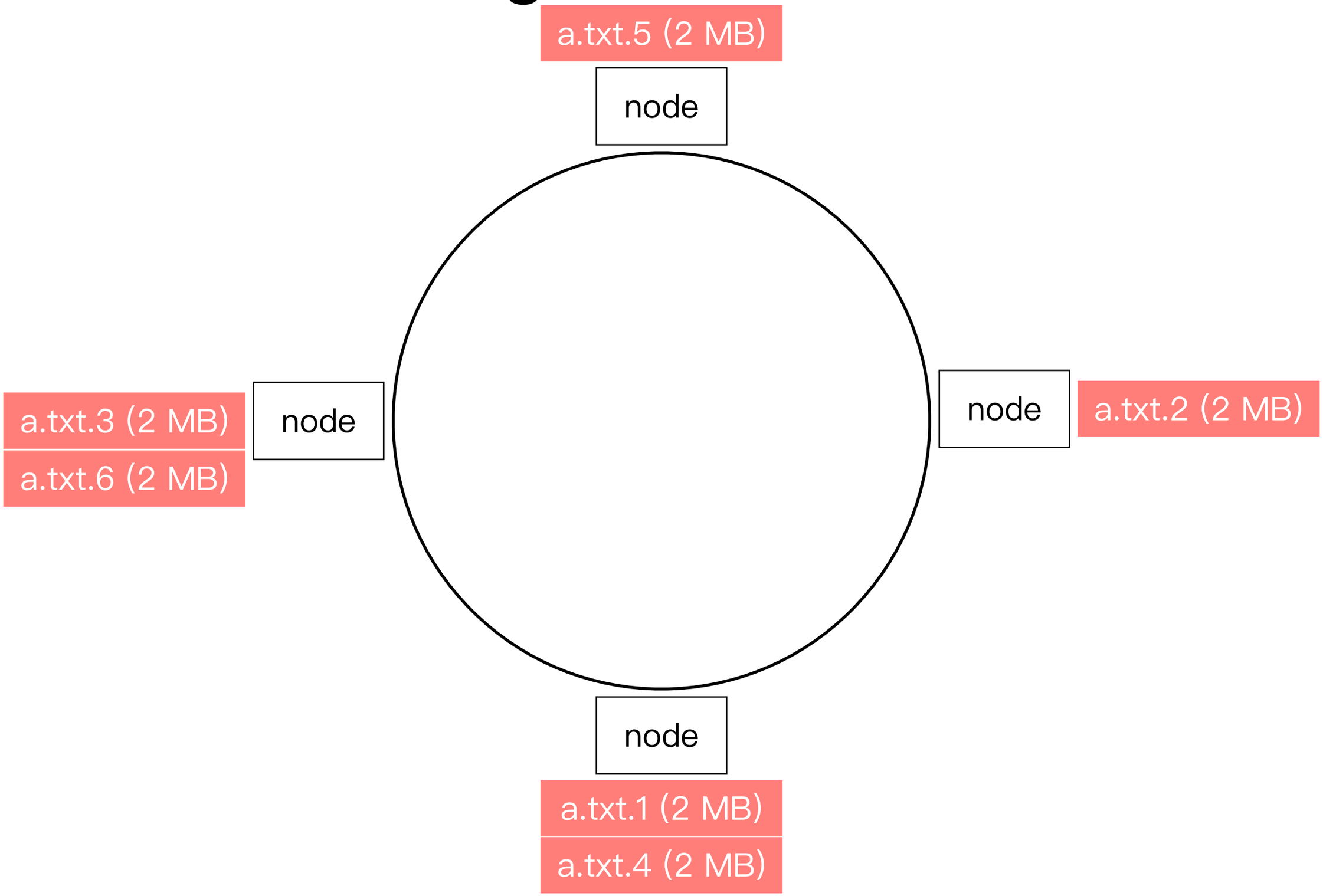
Correctness – Data Migration (33%)



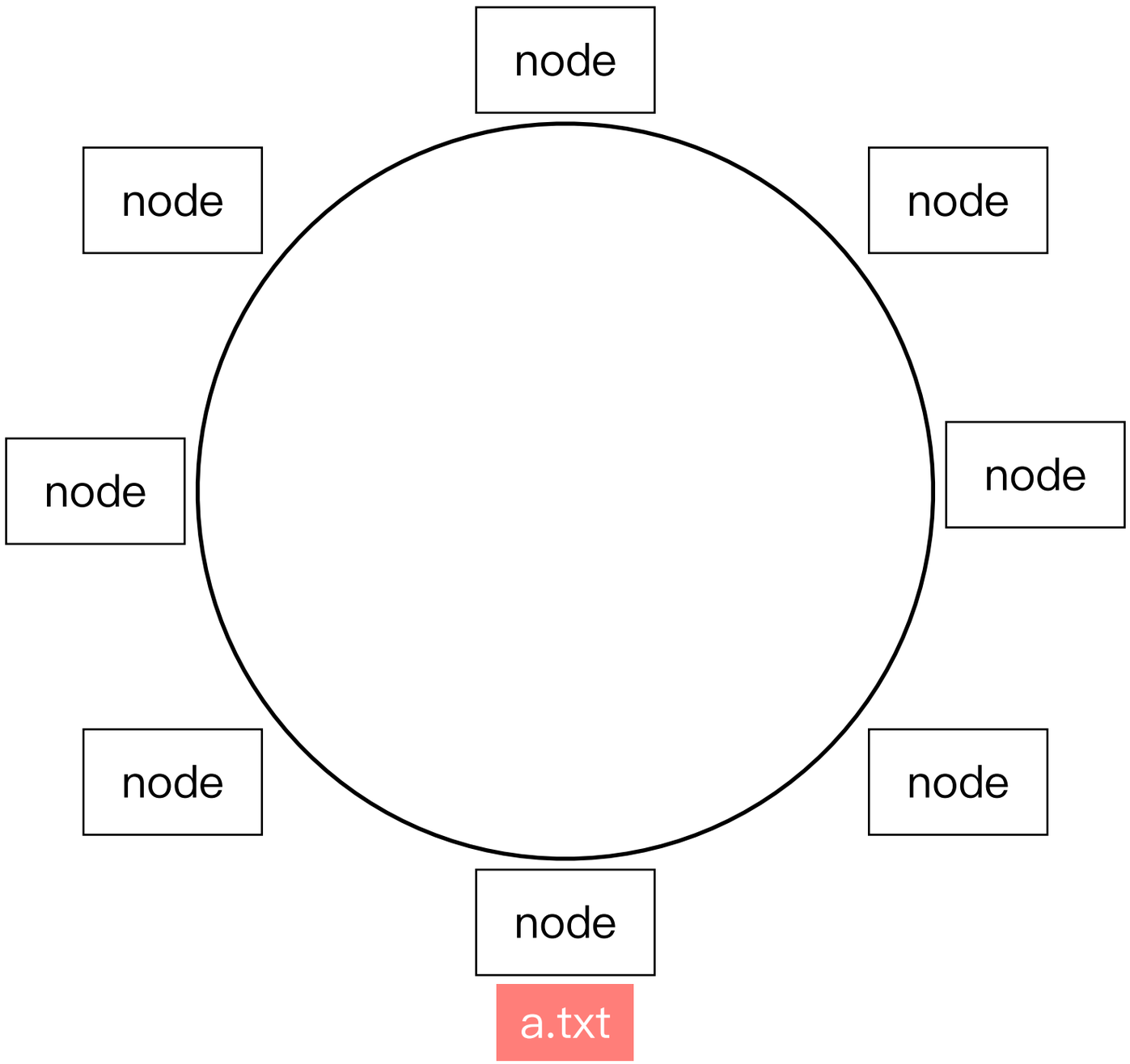
Load Balancing – File Chunks (33%)



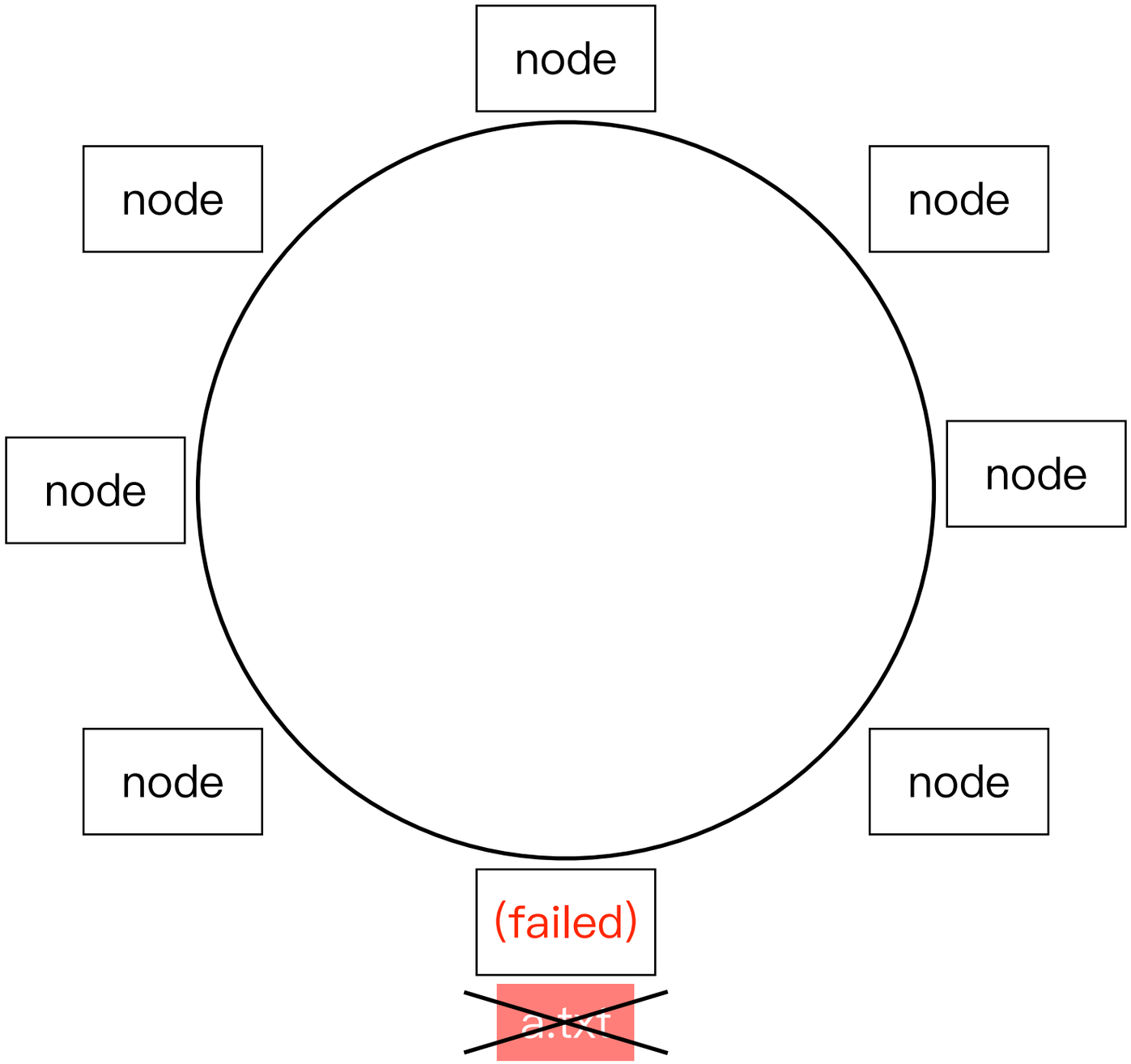
Load Balancing – File Chunks (33%)



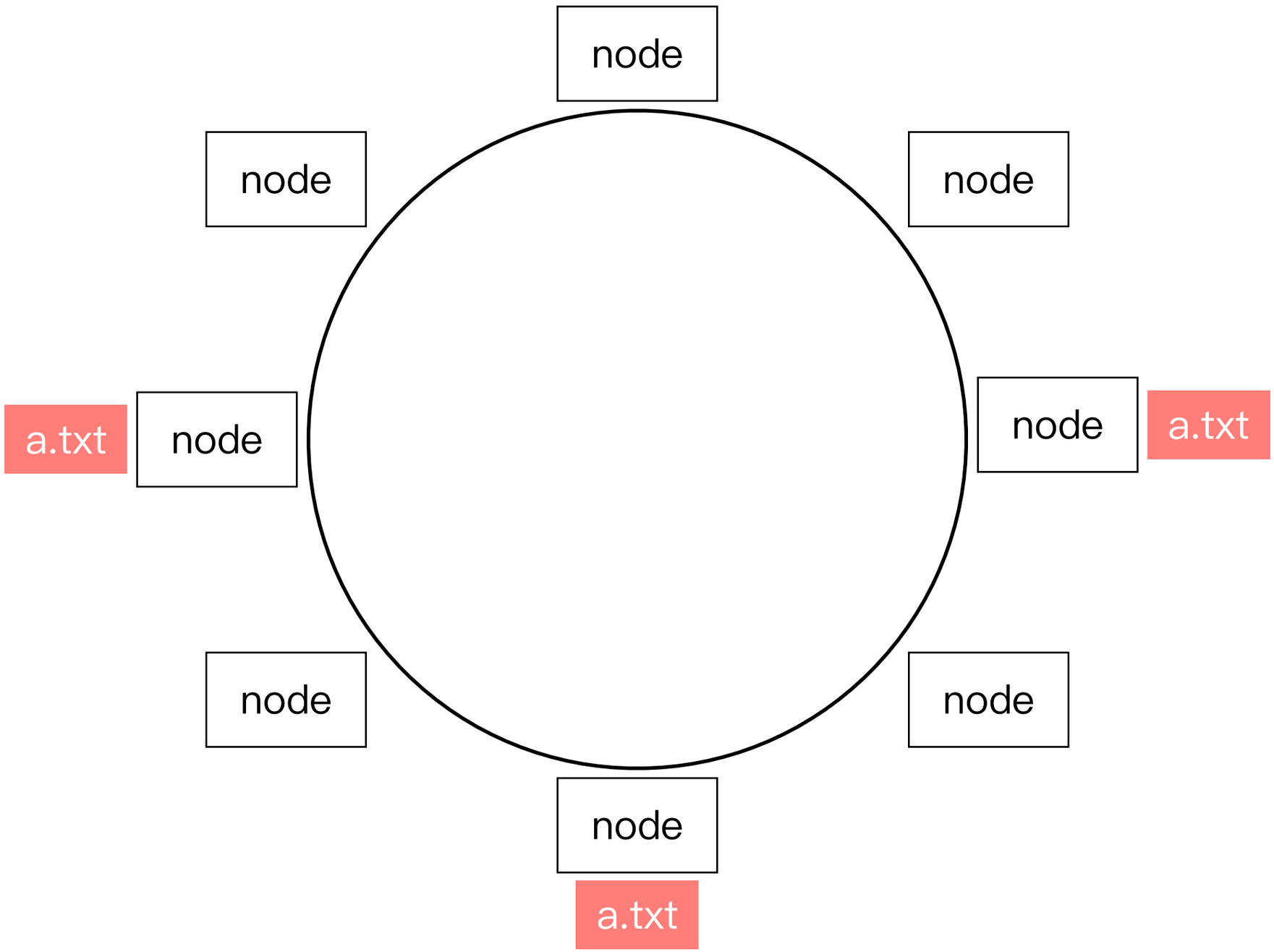
Fault Tolerance – Replication (33%)



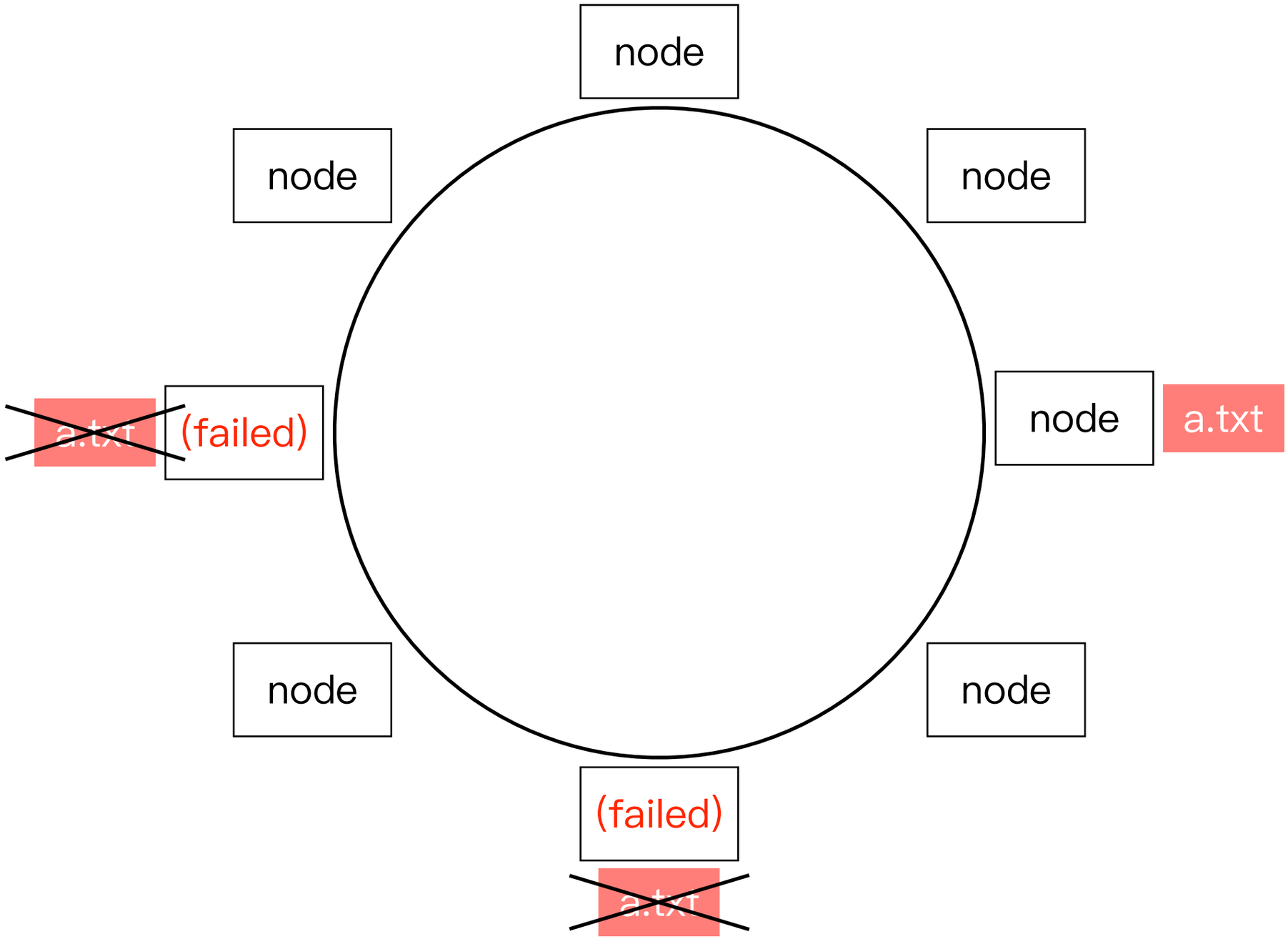
Fault Tolerance – Replication (33%)



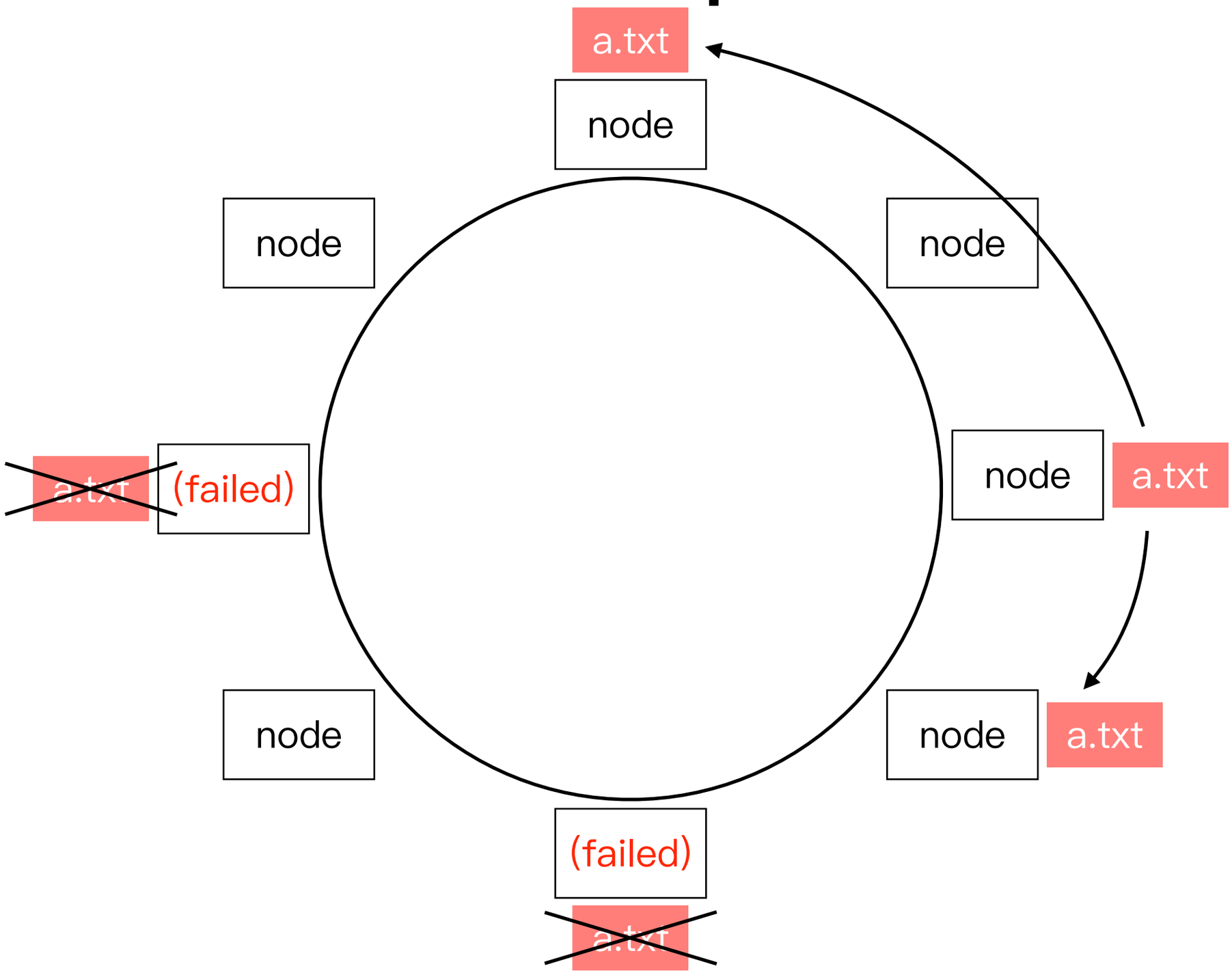
Fault Tolerance – Replication (33%)



Fault Tolerance – Replication (33%)



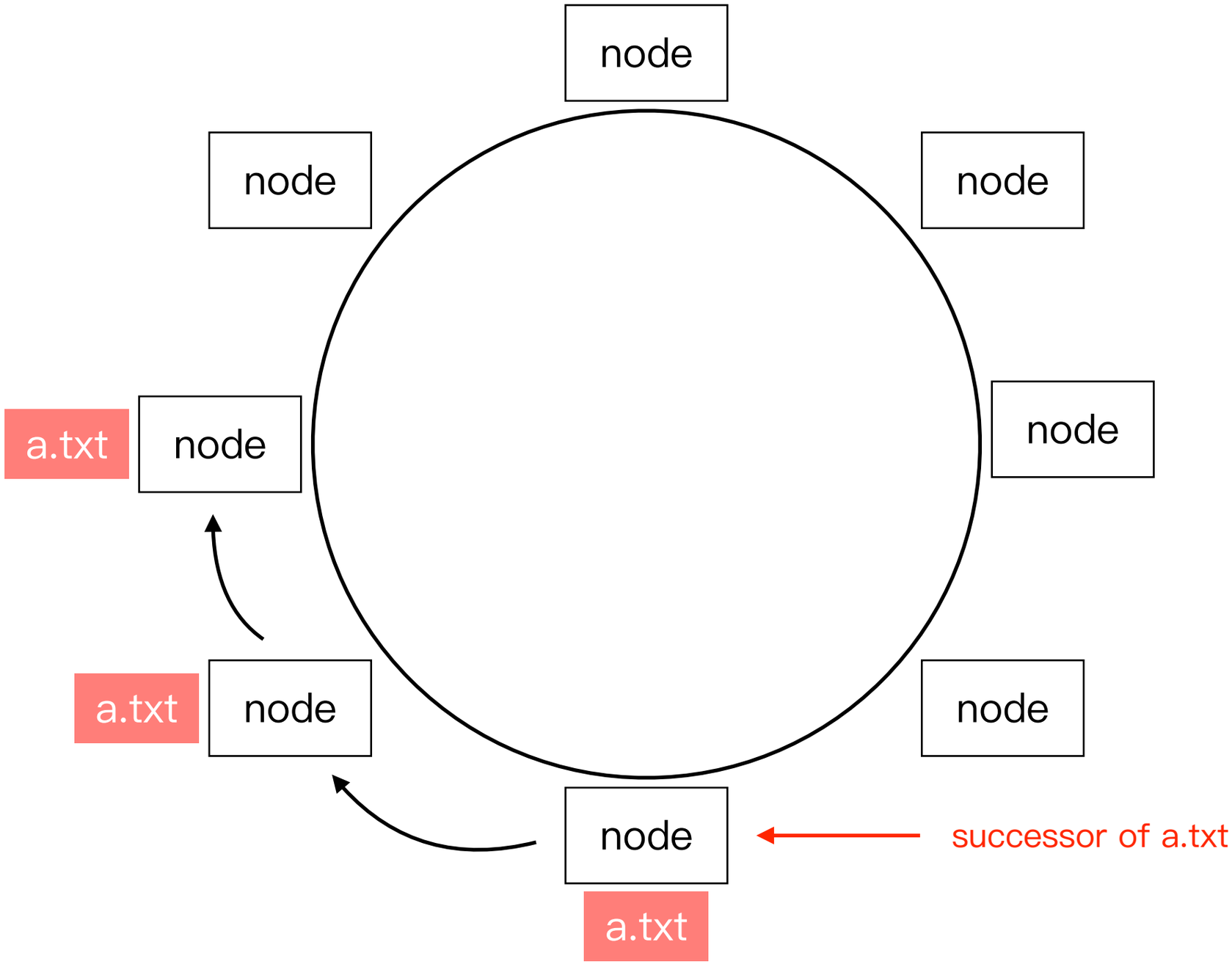
Fault Tolerance – Replication (33%)



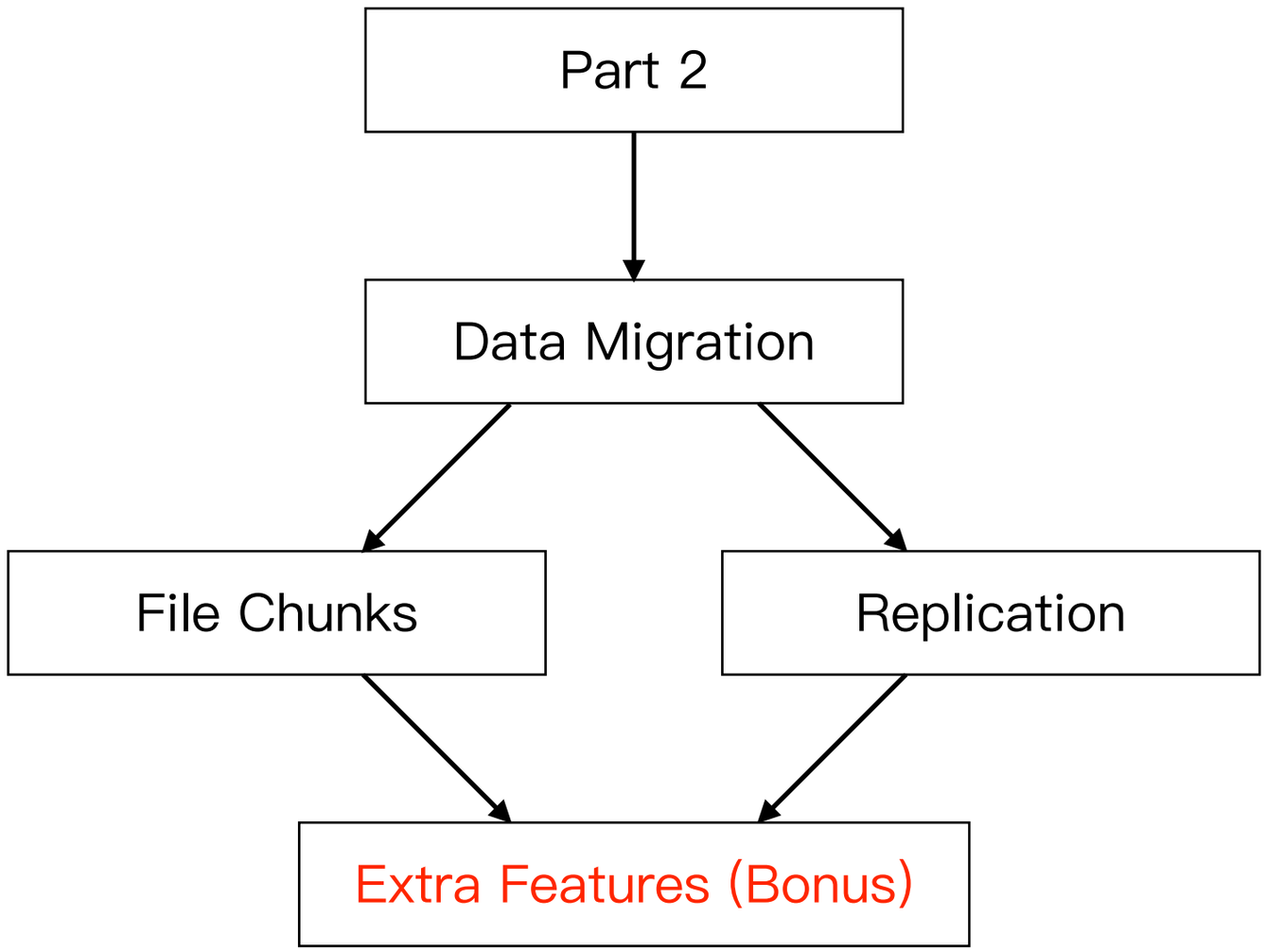
Fault Tolerance – Replication (33%)

- You may refer to the system design of CFS to implement the replication functionality

Fault Tolerance – Replication (33%)



Grading Policy



Grading Policy

- Demo (5/1 – 5/3)
  - 線上測驗 → Midterm Project Demo 時間 (before 4/13 23:59)