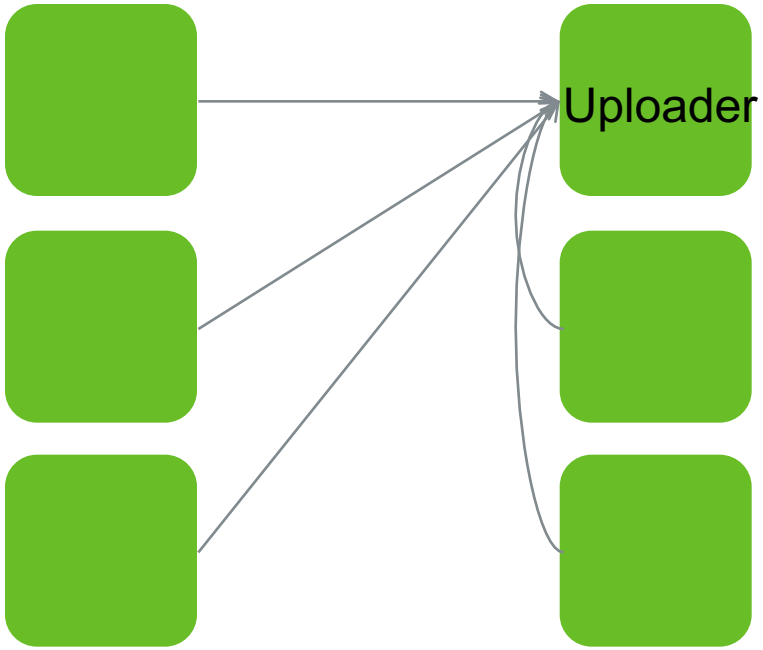
YARN file broadcast service

Zhiyuan Yang

Mentor: Bikas Saha, Li Lu

Problem

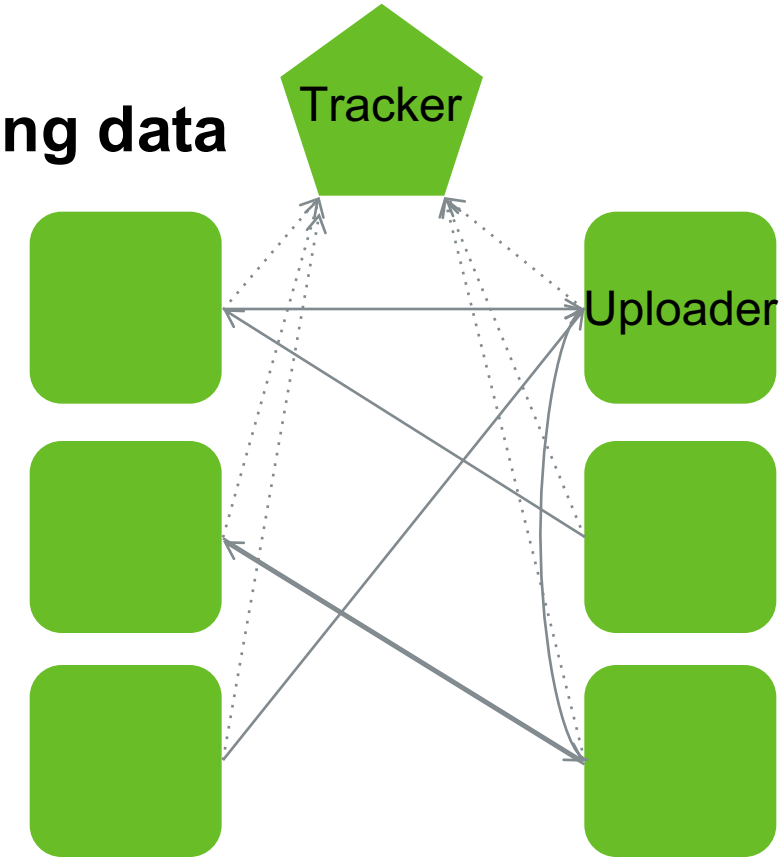
- **YARN application need to broadcast the data**
  - YARN: resource localization service
  - MapReduce: distributed cache
  - Hive: replication-based join
  - Tez: large file broadcast
  - Docker deployment: distribute large image
- **Current HDFS-based centralized solution is not good!**
- **Bottleneck of source nodes' bandwidth**
- **Bottleneck of cross-rack bandwidth**



Prototype1 - BitTorrent

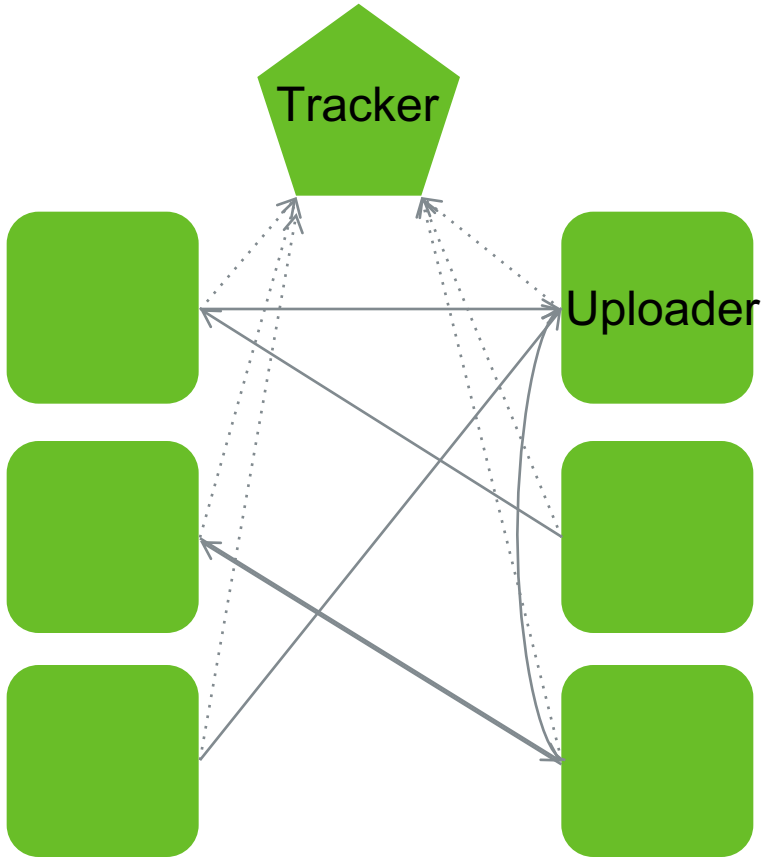
- **Idea: nodes help serving the data after getting data**

- **Naive BitTorrent solution**
  - data can be downloaded from any peer that



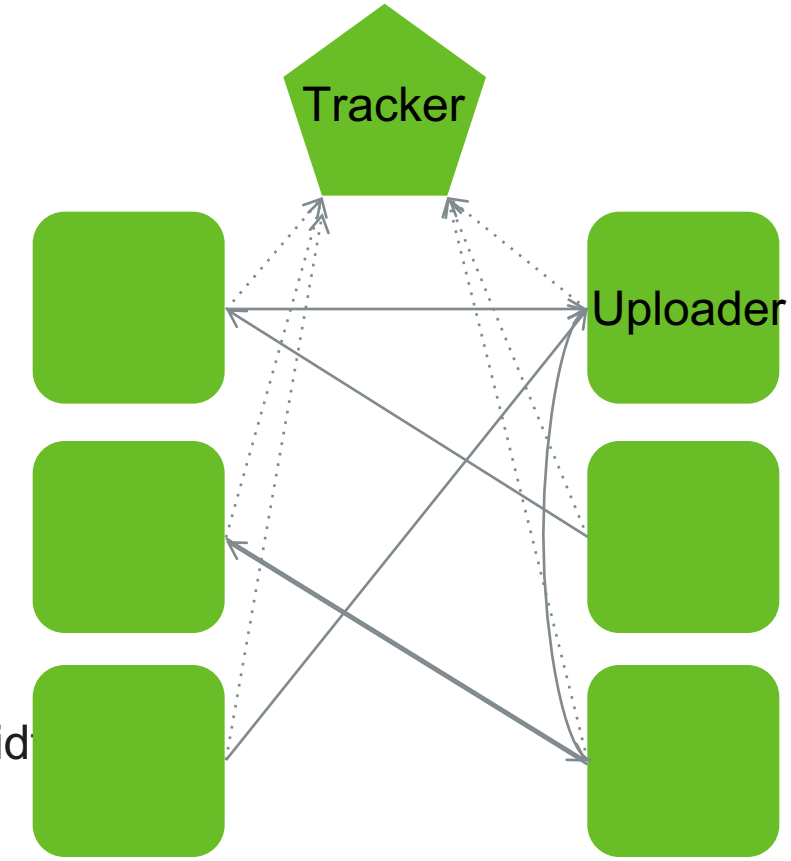
Benefits

- **High performance**
  - No longer abuse few source nodes



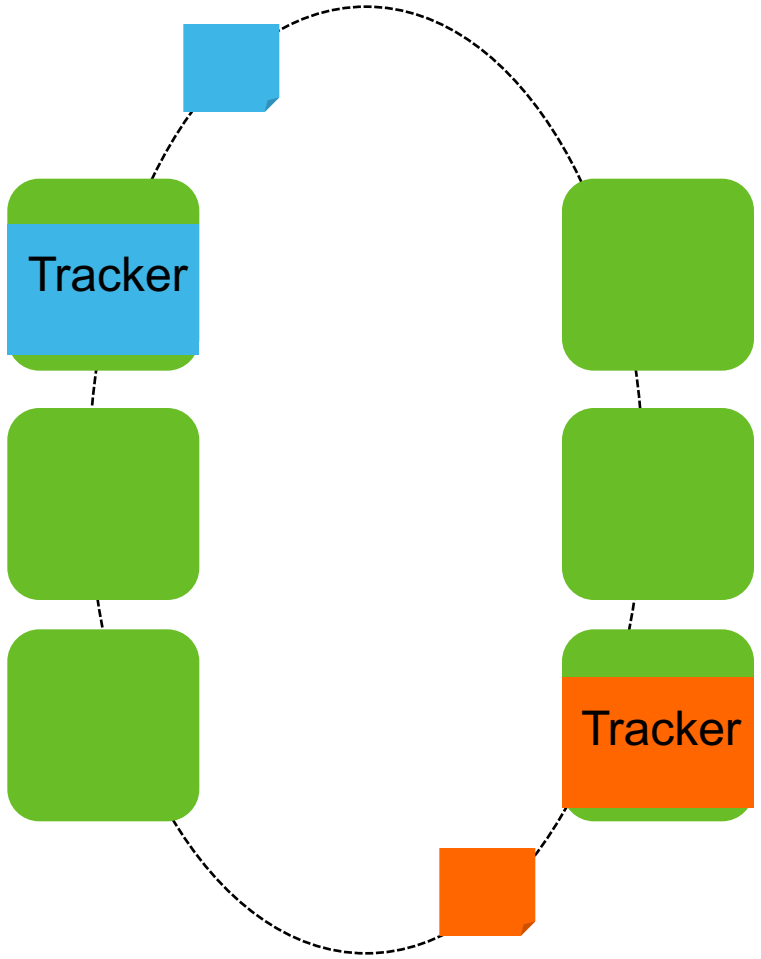
Issues

- **No fault tolerance**
  - single point of failure (tracker daemon)
- **No load balance**
  - single tracker for all broadcasts
- **Limited scalability**
  - bound by tracker's ability
- **No topology awareness**
  - any peer can contact other other, abuse on cross-rack bandwid
- **Operational cost**
  - need restart failed tracker daemon



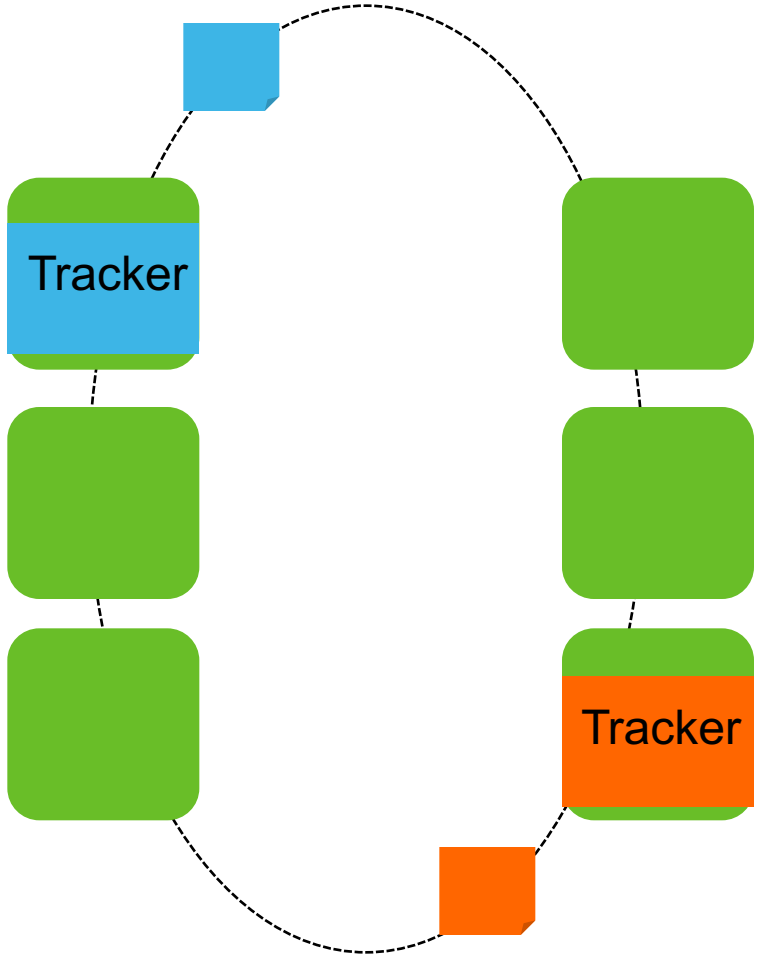
Prototype 2 – Trackerless BitTorrent

- **Every node is not only peer, but also tracker**
- **Every node uses the same consistent hashing to determine who is the tracker(rack master)**
- **There is different tracker for different broadcast**



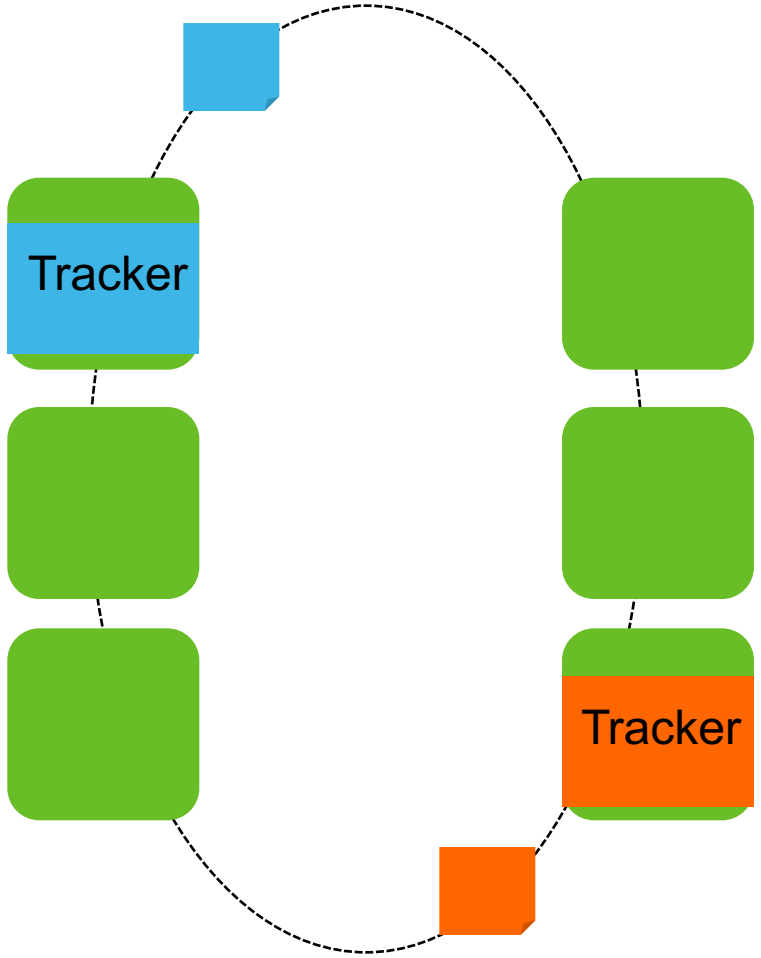
Benefits

- High performance
- **High scalability**
  - no longer bound by single tracker's ability
- **Load balancing**
  - different rack masters for different torrents
- **Fault tolerance**
  - if a node fails, the next one on the ring will take over its responsibility
  - if it recovers from failure, it become the tracker again
- **No operational cost**



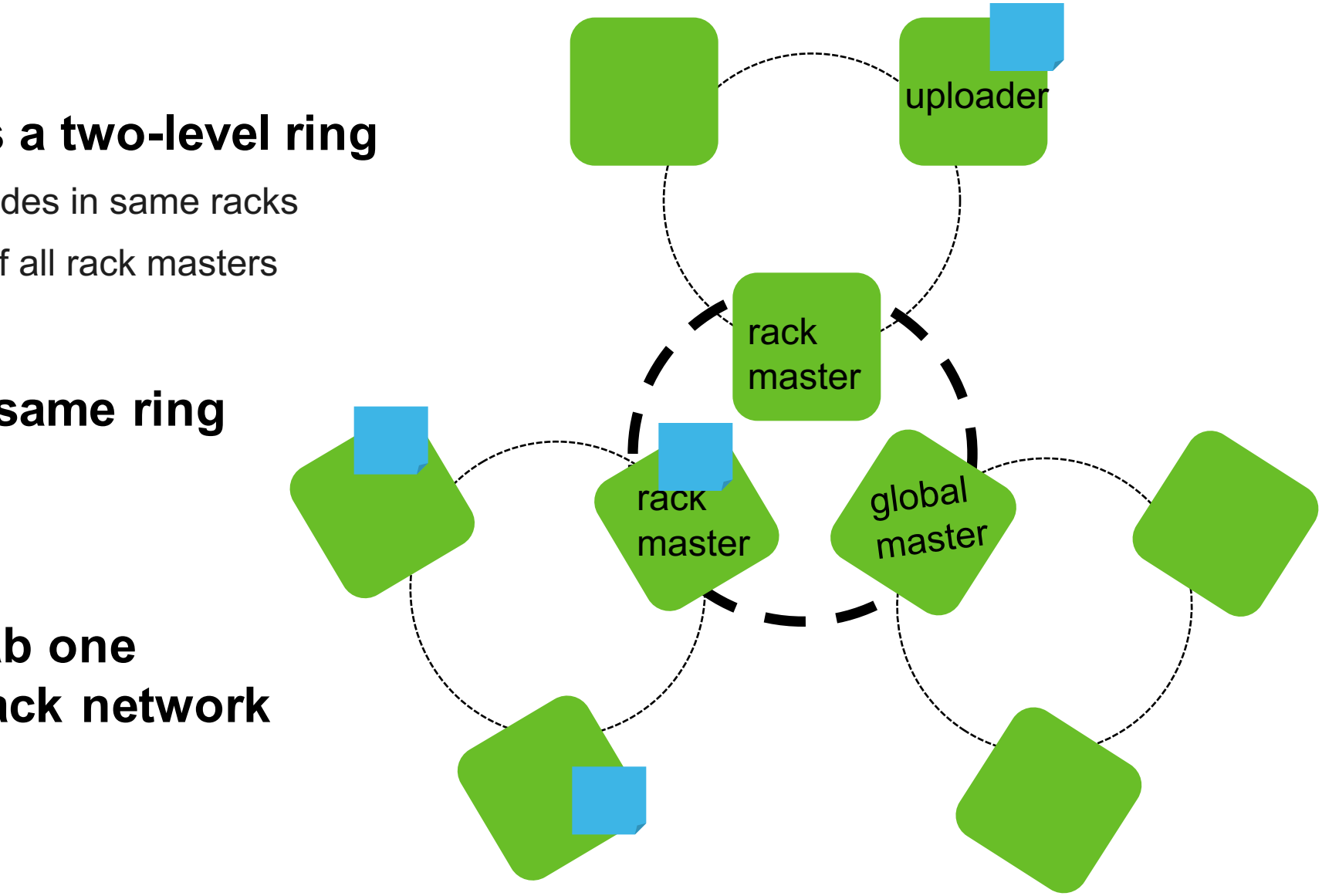
Issues

- **No topology awareness**
  - any peer can contact other other, abuse on cross-rack bandwidth



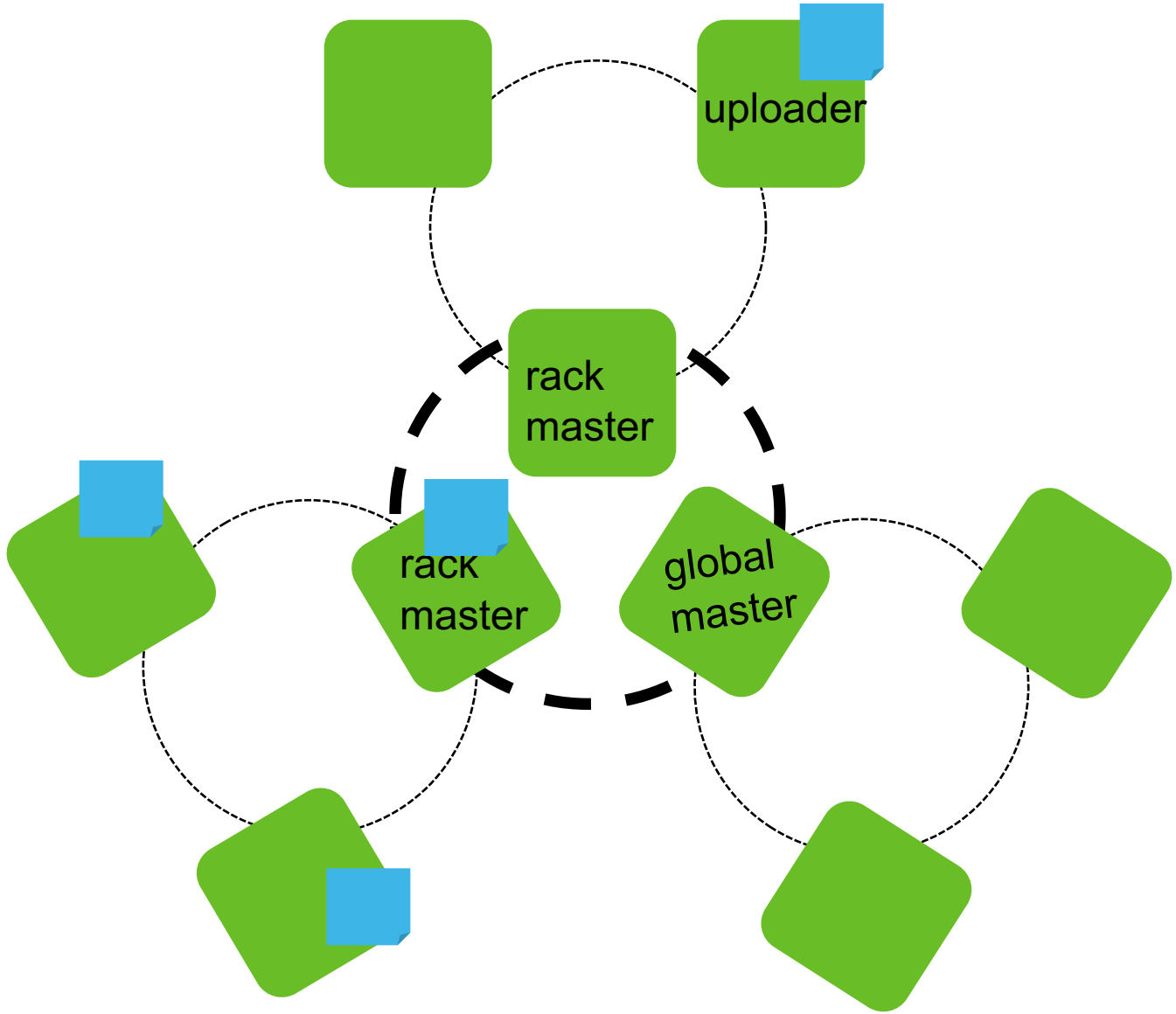
Prototype 3 – Two-level Trackerless BitTorrent

- **Take the cluster as a two-level ring**
  - the first level is rings of nodes in same racks
  - the second level is rings of all rack masters
- **Only nodes in the same ring can communicate**
- **Each rack only grab one copy from cross-rack network**



Benefits

- High performance
- High scalability
- Load balancing
- Fault tolerance
- No operational cost
- **Topology awareness**



Use Case

- **YARN: resource localization service**
- **MapReduce: distributed cache**
- **Hive: replication-based join**
- **Tez: large file broadcast**
- **Docker integration: distribute large image**

Summary

- **Problem: large file broadcast on large cluster**
- **Solution: two-level trackerless BitTorrent**
- **Result:**
  - high performance
  - high scalability
  - fault tolerance
  - topology awareness
  - load balancing
  - no operational cost