CS 744: OWL

Shivaram Venkataraman

Fall 2022

ADMINISTRIVIA

Project checkin – feedback

GPU availability

Midterm 2: Dec 6th in class

Poster presentation: Dec 13th

Final report: Dec 20th

NEW DATA, HARDWARE MODELS

CONTENT DISTRIBUTION

What is content?

- Docker containers
- AI models
- Search indexes

How is this workload different?

CHALLENGES / GOALS

Goals

- Minimize requests to external storage
- Latency of content fetch
- Availability

Challenges

- Load spikes hot content
- Different policies for contents
- Manageability

PRIOR SOLUTIONS

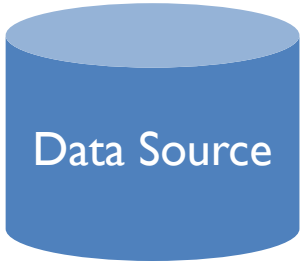
Hierarchical caching

- Need a lot of resources
- Quotas, Traffic bursts

BitTorrent

- Scalable, decentralized
- Stale peer data,
- Lack of global picture

OWL



OWL DESIGN

Peers, Superpeers

Trackers

Ephemeral Distribution Trees

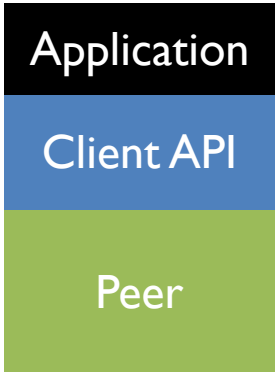
Tracker Sharding

Fault Tolerance

PEERS, SUPERPEER

What is a Peer?

- Simple API, functionality
- Ask Tracker where to fetch
- Cache in memory / local disk



SuperPeer

- Standalone process (no client)
- More resources for caching

TRACKERS

Centralized state for large number of peers

Peers register with a random tracker

What is state?

- Chunk → Peers caching it
- Peer → List of chunks cached

Soft-state (similar to GFS)

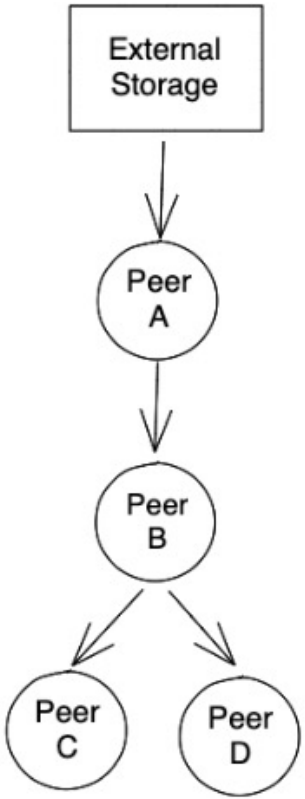
DISTRIBUTION TREES

To fetch data

- Peer sends get_data(range) → Chunks
- For a chunk, getSource(chunk) →
  - Peer/Super Peer/External Storage

Trackers

- Build ephemeral distribution tree
- Stream data from peers
- Locality based



POLICIES IN TRACKERS

Selection Policy

Which peer should we fetch from

Caching Policy

Which blocks should be stored in memory

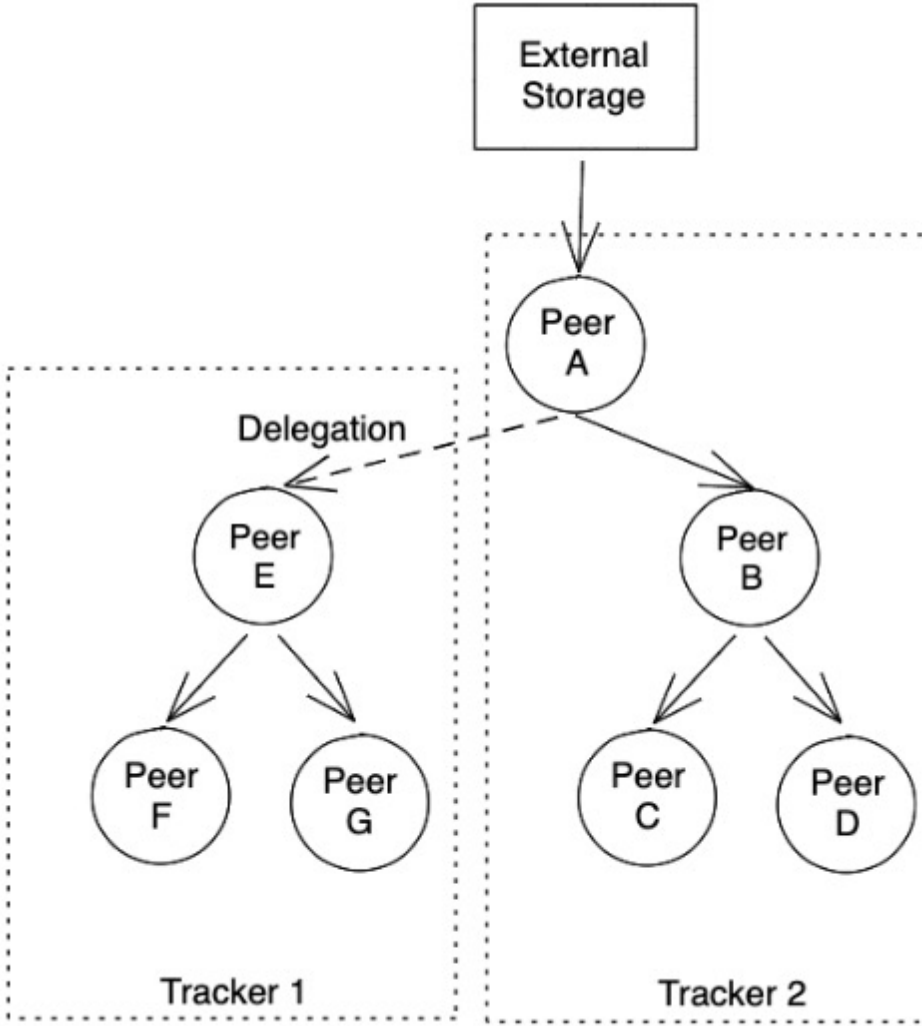
Buckets to control configuration across applications

TRACKER SHARDING

Millions of peers, tracker bottlenecks

Partition peers across trackers

Challenge?



VIRTUAL SUPERPEERS

What data should a peer store?

- So far: Data already fetched a peer
- Can we use a peer for caching other data?

Partition cache space into peer / virtual superpeer

- Use spare memory on the machine

Tracker-only concept!

SUMMARY

Problem: Content distribution is challenging

Approach: Decentralized data-plane, centralized control plane

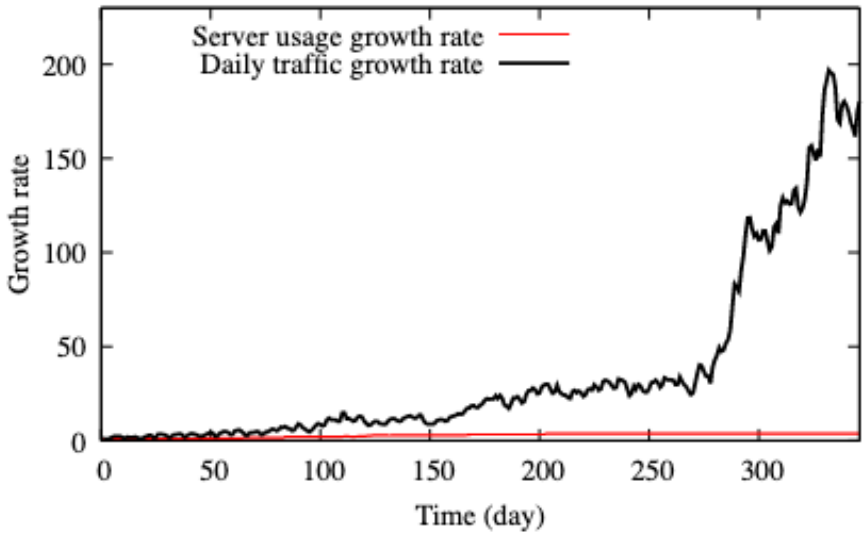
Features

- Ephemeral data distribution trees
- Policies on tracker for selection, caching
- Sharding trackers for scalability, fault tolerance

DISCUSSION

https://forms.gle/cbAyPYsVGqdcaZyx9

What is one disadvantages of the design used in Owl? Construct one scenario to highlight how this disadvantages might affect a client.


Server usage growth rate
Daily traffic growth rate
Growth rate
200
150
100
50
0
0
50
100
150
200
250
300
Time (day)

NEXT UP

Next steps:

- TPU Paper
- Midterm 2, Dec 6th
- Poster session, Dec 13th