Consistency and Replication of Distributed Web-Based Systems

Presented by: Korn Sooksatra
Introduction

- There are so many users requesting web servers
Challenges

• Low latency
  – To let clients experience lower latency

• Congestion
  – To reduce traffic around web servers

• Consistency
  – To make data in caches and replicas up-to-date

• Availability
  – To make web servers available all the time
Caching

• Goal
  – Reduce incoming packets at servers' sides
  – Reduce latency
  – Reduce traffic

• Able to place caches in 2 locations
  – Web browser caching
  – Web proxy caching
    • Hierarchical caching
    • Cooperative caching
Web Browser Caching

- Get documents from a cache of browser instead
Web Proxy Caching

- Since clients usually run web proxies, placing caches to them could improve performance

- There are 2 improvements
  - Hierarchical caching
  - Cooperative caching
Hierarchical Caching

- If a proxy's cache encounters a cache miss, ask its parent and siblings:

https://www.cse.wustl.edu/~jain/cis788-99/ftp/web_caching/fig5.gif
Cooperative Caching

- If encounter a cache miss, ask its neighbors

Hierarchical vs Cooperative

- Since cooperative caching are connected with high-speed links, transmission time is lower.
- However, hierarchical caching achieves better connection time.
- Expected latencies (transmission + connection) for hierarchical ones are better.

Consistency of Caching

- Could use If-Modified-Since
  - Contact a server too much

- Squid Web Proxy
  - Assign an expiration time for each document
    \[ T_{\text{expire}} = \alpha (T_{\text{cached}} - T_{\text{last modified}}) + T_{\text{cached}} \]
  - From practical experience, \( \alpha = 0.2 \)
  - Drawback
    - Bad for documents which have not been modified for a long time
Consistency of Caching

- A server sends invalidations to proxies
  - Tell proxies when a document is modified
  - Hardly used in practice
  - Drawback
    - Keep track of a large number of proxies
    - Flood the traffic
  - In term of bandwidth and client latency, this is the best

Replication for Web Hosting Systems

• Goal
  – Make the site easily and continuously accessible

• Content delivery networks (CDNs)
  – Web hosting service which replicates the site over the internet
  – There is a feedback-control loop
    • Metric estimation
    • Adaptation triggering
    • Taking appropriate measures
Metric Estimation

- Tradeoff between (access time) and (financial cost and bandwidth for updates)

- Example
  - Latency metrics
  - Bandwidth metrics
  - Spatial metrics
  - Consistency metrics
  - Financial metrics
Adaptation Triggering

- When does it need to replicate documents?
  - Periodic evaluation is fine, but cannot handle a flash crowd (a sudden burst)
  - Flash-crowd predictor is needed
  - Baryshikov (2005) proposed a flash-crowd predictor which is quite successful

https://ars.els-cdn.com/content/image/1-s2.0-S0167739X13001672-gr9.jpg
Adjustment Measure

- Embedded documents (e.g. images) are rarely changed, so good to cache or replicate them
  - Use a modified URL to go to the CDN DNS
  - In the modified URL, there is a unique identifier for consistency
Replication for Web Applications

- Websites nowadays are often dynamic
  - Contents of documents depend on queries from clients
Database Copy

- **Full replication**
  - Good for complex queries (join operations)
  - Bad when there are many updates
    - An origin server needs to update its replicas to keep consistent

- **Partial replication**
  - Sufficient for simple queries (one table involved)
  - Difficult to decide which data should be replicated
    - The origin server might apply access traces to check which data should be replicated in a certain edge server
Partial Replication

- Utilize a content-aware cache and a content-blind cache
  - Content-aware cache
    - Check queries if can answer with local data
    - Difficulty to keep consistent
      - An origin server needs to know which records associated with which templates
  - Content-blind cache
    - Hash queries and check them
    - Reduce computation of edge servers
    - Wasteful of storage
Example of CDNs

- Akamai
  - 200,000 servers over 130 countries
  - Used by
    - 20 of the top global e-commerce sites
    - 30 of the top media and entertainment companies
    - 9 of the largest newspapers
    - Etc.
Future Work

• So far, we considered only caching and replication.

• Hence, the next work is about how to choose replica for each request when the closest replica is congested without network monitoring.

• There could be an heuristic function for this decision.
Thank you