web caching

Γιάννος Γρηγορίου
Agenda

- Introduction-Why web caching?
- The Expected gains
- Cache Types
- Cache Architectures
- Cache Replacement policy
- Cache Consistency
- Prefetching
- Caching dynamic contents and multimedia streaming
- Conclusion and Future works
Introduction

What is Web Caching?
- Introducing proxy servers at certain points in the network that serve in caching Web documents for faster client access.
- Comparable to the cache memory in a computer system.

Why is it needed?
- Rapid growth in HTTP traffic (largest part of the Internet traffic) which causes more network congestion and server unavailability.
- The number of static web pages almost doubles every year.
How Web Cache Works

1. Client sends HTTP request
2. Web Cache responds immediately if cached object is available
3. If object is not in cache, Web Cache requests object from Application Server
4. Application Server generates response (may include Database queries)
5. Application Server responds to Web Cache
6. If response is cacheable, Web Cache retains a copy for subsequent requests
7. Web Cache compresses page and responds to Client
The Expected gains

- **Bandwidth saving** - it reduces the amount of bandwidth used by a client and of course this saves money.

- **Improving content availability.**

- **Improving web server availability.**

- **Reducing network latency** - the request is satisfied from the cache which is closer to the client.

- **Server load balancing.**

- **Improving user’s perception about network’s performance** - it takes less time for a representation to be displayed. This makes the Web seem more responsive.
Cache Types

Types of caches

- **Client Caches** - Web browsers like Internet Explorer etc. Cache stored on the local hard drive - serves one user.

- **Proxy Caches** - Placed in between (LAN network and ISP network) the client and origin server - serves many users

- **Reverse Proxy** – Placed before the web server for load balancing – serves the web server.
Terminologies

- **Cache miss**

![Diagram of Cache miss](image)

- **Cache hit.**

![Diagram of Cache hit](image)
Cache Types – Cache hit
Cache Types – Cache miss

Cache miss

ISP

ISP Proxy server

Reverse proxy

Web server

Web browser cache

client

router
Caching Architectures

- Proxy Caching
  - Deployed at the edges of the network
  - Unavailable cache → Unavailable network
  - Single point of failure
  - User browser manual reconfiguration in times of failure
  - Browser auto-reconfiguration is a recent trend

(a). standalone

![Diagram of caching architectures with client, cache, router, and Web components]
Caching Architectures

- **Reverse Proxy Caching**
  - Placing proxies near the content provider

- **Transparent Caching**
  - Eliminates the needs to manually configure web browsers
  - Router-based transparent proxy caching
  - Switch-based transparent proxy caching

![Diagram of caching architectures](image)

- (b) router-transparent caching
- (c) switch-transparent proxy caching
Caching Architectures

- A Standalone Proxy in a network.
  - Single point of failure
  - Bottleneck
  - Lack of robustness
  - Lack of scalability

**Solution:** Cooperative caching, where caching proxies collaborate with one another in serving request.
Caching Architectures

**Hierarchical caching**
- Caches are connected in a parent-child relationship.
- Tree structure where every node points to its parent.
- Child caches are polled first.
- Higher level caches are polled next.
- Parent cache responsible for retrieving a fresh copy.
- Nodes higher in the hierarchy have large user population and higher hit rates.
Caching Architectures

■ Hierarchical caching
  □ Benefits:
    ■ Scaleable architecture
    ■ Reduces bandwidth demand on origin servers.
  □ Drawbacks:
    ■ Each Hierarchy level introduces additional delays
    ■ A top level server located far away from the uses may not prove to be beneficial
    ■ Documents are replicated on each level
    ■ Higher level caches may become bottlenecks with long queuing delays
Caching Architectures

- Co-operative (Transversal) Caching (Mesh)
  - Mesh Topology
  - Uses Internet cache protocol (ICP) to share information, to balance loads, and to provide resistance to cache failures.
  - When caches misses, it sends an ICP query to neighbor caches, the neighbors respond with ICP reply indicating a hit or a miss.
Caching Architectures

- Co-operative Caching (Mesh)
  - Benefits
    - Finds out which neighboring cache has the object
    - Load balancing
    - Prevents top level choking
    - Can be configured so that neighboring caches are not queried for nearby origin servers
  - Drawbacks
    - ICP message overhead
    - Replication of objects
    - No security (Uses UDP)
Caching Architectures

- **Hybrid**
  - A combination of Hierarchical and Mesh architecture.
  - When caches misses, it sends a query to neighbor caches at the same level (siblings) using multicast before sending a cache request to a parent.
    - Combines the advantages of the 2 previous architectures and minimized their disadvantages.
A Typical Caching Architecture

WEB server

memory

L2 cache

ISP proxy server

L1 cache

LAN proxy server

client

CPU

Web browser cache

A Typical Caching Architecture

WEB server

memory

L2 cache

ISP proxy server

L1 cache

LAN proxy server

client

CPU

Web browser cache
Caching Architectures

![Diagram of multi-processor system with memory, cache, and processor blocks connected by a bus.]

- **Clients**: ISP proxy servers, WEB servers, CYTA proxy server, UCY proxy server, EAC proxy server, COOP proxy server
- **Processor Blocks**: Proc 1, Proc 2, Proc 3, Proc 4
- **Cache Blocks**: cache, cache, cache, cache
- **Shared Resources**: Bus, Shared cache

Diagram represents a multi-processor system with caching architectures.
Web Caching Software

- **CERN httpd** is the original Web server and proxy cache. It is not used much these days.
- **Squid** is freely available, open source software that runs on most Unix platforms. Squid is likely the most widely deployed caching solution and full of many useful (and some not-so-useful) features.
- **Microsoft** has a high-end product called ISA server. The older **Microsoft Proxy Server** is a software product that runs on Windows NT.
- **Cisco** has an appliance called the Cache Engine. The product, which is small and modular, is designed for clustering with WCCP.
- **Oops!** is a proxy server; the main aims of its development being stable operation, service speed, main protocols support, modularity, ease at use.
- **Oracle9iAS Web Cache** combines caching, compression and assembly technologies to accelerate the delivery of both static and dynamically generated Web content. It also implements **Edge Side Include** (ESI) protocol advanced by Oracle, Akamai, and others.
- And many others…
Cache Replacement Policy

- Purpose is to decide which items to evict when there is no more space in the cache. → an old item to be replaced by a new one

- The effectiveness of a cache primarily depend on the accuracy of the replacement decisions that are taken upon the cache’s saturation.
Cache Replacement Policy

Well known cache replacement algorithms

- Least recently used (LRU) - evicts the item that was requested least recently.
- Least frequently used (LFU) - evicts the item that was least frequently used.
- Size - evicts the largest (MAX) or the smallest (MIN) object.
- Latency Access Time (LAT) - evicts the document with the lowest download latency first.
- Greedy Dual Size - evicts objects with the least cost/size.
- LRU-MIN - minimizes the number of objects to be removed upon saturation.
- HYB (Hybrid) – an optimal combination of LAT-LFU-SIZE.
Cache Consistency

- Purpose is to avoid serving an outdated cached copy of an item to the users.

- Items kept in cache may not be consistent with the original copy, if the original copy changes faster than the cache refresh frequency.
Cache Consistency

- Approaches to maintain cache consistency
  - Polling-Every-Time (PET)
  - Time-To-Live (TTL) based validation
  - Server Invalidation
Cache Consistency

- Polling-Every-Time (PET)
  - Clients send an “If modified since” request to the server on every cache hit. If the document is modified since time specified, new data is sent otherwise no data is sent.
  - **Drawback**: waste the network bandwidth if the cached copy has not yet updated on the original server.
Cache Consistency

- Time-To-Live (TTL) based validation. The proxy assigns a TTL value for each object
  - **Drawback:** an object can expire but has not yet updated on the original server, the proxy still needs to verify with the server.

- Proactive Polling
  - a TTL based validation where the proxy proactively poll the server to check the validity of the cache object either at a fixed interval or an adaptive interval.
  - **Drawback:** waste the network bandwidth if the cached copy is rarely accessed after validation.
  - To overcome this problem one may batch the validation request and responses over normal HTTP traffic.
Cache Consistency

**Server Invalidation**
- Web Server detects an object change and sends invalidation messages to all caches that have recently accessed it.
- Upon receiving the invalidation message the caches delete the object and do not retrieve a new copy.

**Benefits:**
- Strong cache consistency
- Reduced network transactions
- Improved cache utilization (by deleting stale copies)

**Drawbacks:**
- Storage needed on the web server to keep track of caches
- CPU Overhead - search and update lists
Cache Consistency

- Weak Consistency
  - TTL-based validation
  - Proactive polling

- Strong Consistency
  - Polling Every Time (PET)
  - Server Invalidation
What to cache?

- Objects with
  - long freshness time
  - small in size
  - They have high access latency

are best suited for caching
What not to cache?

- Objects that cannot be normally cached are:

  - Objects that are password protected
  - Any file exceeding a predefined limit
  - SSL requests which are tunneled
  - Any URL with /cgi-bin/ → Dynamic web contents
  - If the origin server wants to record the exact number of times their page is viewed, they may decide not to allow their documents to be cacheable (known as cache busting)
Advantages / Disadvantages of web caching

- Advantages:
  - Decreases network congestion
  - Reduces access latency
  - Reduces work load on the origin server
  - In case of origin server crash, a client can obtain a cached copy
  - Information distributed more widely at a lower cost

- Disadvantages:
  - Stale data due to lack of proper proxy updating
  - Access latency may increase in case of a cache miss
  - A single proxy server is a single point of failure
  - Proxy bottleneck
Prefetching

- Proactively preloads data from the web server into cache to facilitate near-future accesses.

Prefetching policy issues:

- Avoid overloading the network
- Avoid overflowing the cache’s storage space
- Prefetching should only apply to user request and not to requests from neighbors
- Prefetched document should remain in cache long enough to have a chance to be accessed
- When a document is removed from the cache, should also removed all the documents prefetched for that document.
Prefetching

- Prefetching methods and approaches
  - Statistics-based approaches (user behavior)
  - Prefetching documents near the top of the requested document
Prefetching Policies

- **Mixed-access pattern**
  - Prediction makes use of aggregate access from different clients but does not explore which client makes the request

- **Per-client access pattern**
  - Analyze access patterns on per-client basis and use the aggregate patterns for predictions

- **Object structural information based**
  - Prefetched the hyperlinks for the accessed document.
Caching dynamic contents

- Dynamically Generated Pages
  - Increased due to
    - Database centric e-commerce application
    - Frequently update contents
    - Personalization

- Proxy caching is ineffective for such pages
Caching dynamic contents

How to cache Dynamic Contents?

Policy based caching – Cache the generation process of the dynamic pages and not the pages themselves.

- Caching query results
  - Scalable and transparent architecture possibility

- Caching databases contents
  - Limited scalability and transparency

- Caching the application (caching applets from web servers)
- Partial-Page caching (split web page into templates and fragments)
- Combination of the above
  - i.e. DOM Proxy: a combination of the application and query results caching.
Caching multimedia streaming

- Caching streaming objects (audio and video)
  - Multimedia objects have distinct features such as,
    - Large size
    - Long duration,
    - Intensive use of bandwidth
    - Interactivity

- Properties to be study for the design of an effective caching mechanism:
  - Streams do not require to be delivered at once. The server usually pipelines the data to the client through the network
  - Multimedia streams are able to change their size by adjusting the quality
Caching multimedia streaming

- What we want to succeed by caching the multimedia streaming objects?
  - Maximize quality of the delivery stream to the clients
  - Minimize load on the server
  - Minimize the startup latency
  - Provide low-latency recording functionality for the clients

- The main challenge for a proxy caching of Internet multimedia streams is the need for congestion control.
  - Quality Adaptation: Change the compression ratio according to available network bandwidth
    - Problem: there is no correlation between the variation on quality of the cache stream and the required quality for the new session.
Caching multimedia streaming

- Conventional Proxy caching do not perform effectively and in some cases are inapplicable.

- Approaches of caching multimedia objects
  - Partial Caching Algorithm: a small portion of the video is stored on the proxy and as a result the network resources consumed could be significantly reduced.
  
  - Problems:
    - How to optimally choose the portion to cache?
    - How for the proxy and the origin server to jointly deliver streaming objects to clients?
    - Streaming objects often have Variable bit rate and have high demands on transmission delays or delay jitter.
  
  - Layered Encoding: compressed data is split into a base layer which contains the most essential low quality information. The other layers provides optional enhancement information.
Conclusion
Conclusion

- Benefits of web caching and prefetching
  - Reduce latency
  - Improved performance of WWW
  - Balance workload of web servers
  - Decrease effects of network bottleneck

- Side effects of web caching and prefetching
  - Deploy proxy servers may cost more
  - Increase complexity of maintenance
  - Caching and Prefetching themselves will consume bandwidth
New Era

1. Database driven web content
2. E-commerce
3. Anytime and Everywhere
4. Voice over Internet-Real Time
5. Video over Internet- Real Time
6. Security sensitive content
The future for web caching

- **Content Security**
  - Proxy and Firewall
  - Access control
  - Virus scanning
  - Content filtering

- **Handling more complex objects and real-time data**
  - Dynamic content – Databases
  - Mobile Proxy caching
  - Content Distribution Networks (CDN)
  - Real time engine: captures, caches and queries data at speeds greater than 12000 event/s

- **Web Caching based on Ontology**
  - User access pattern prediction
  - Prefetching
  - Cache replacement algorithms
References

3. “Improving the Effectiveness of Web Caching” Jean-Marc, Valerie Issarny, Michel Banatre, INRIA
4. “Caching and Prefetching for Web Content Distribution” Jianliang Xu, Jiangchuan Liu, Bo Li, Xiaohua Jia
5. “Web Caching Prefetching as an Aspect: Towards a Dynamic-Weaving Based Solution”, Marc Segura-Devillechaise, Jean-Marc Menaud, Gilles Muller, Obasco Group
6. “DOMProxy: Enabling Dynamic-Content Front-end Web Caching” Manolis Valiskakis, John Roussos, Panos Georgantas, Timos Sellis, National Technical University of Athens
7. “Proxy Caching for Multimedia Playback Streams in the Internet”, Reza Rejaie, Mark Handley, Haobo Yu, Deborah Estrin, University of Southern California
Thank you!