Community-of-Interest Multicast Cache Loading

Joe Touch

Large-Scale Active Middleware Project
USC/ISI Computer Networks Division

ISI Web Research

Transport for short transactions
- Rate-based pacing (*Heidemann/Visweswaraiah*)
- TIME_WAIT avoidance (*Touch/Faber/Yue*)
- Control block sharing (*Touch/Heidemann/Eggert*)
- Support for satellite and asymmetric channels
- Support for partial order transport

Middleware for cache support
- Multicast push to client caches (*Touch/Hughes/Oswal*)
- Reducing cache hierarchy miss penalty
- Network adaptive caching
- Partial object caching
Primary Focus

Response latency is the critical parameter
- *Netscape* vs. *Word?*
- *Interactive* is much more useful than request/response

All other parameters are resources
- *Processing* (recompute)
- *Storage* (cache)
- *Bandwidth* (anticipate)

*Idle bandwidth is a wasted opportunity*

---

Cache Push Motivation

Bandwidth too high for interactive, or no interactive at all

---

Bytes sent in 100 ms

<table>
<thead>
<tr>
<th>Bandwidth per user (BW)</th>
<th>1B</th>
<th>10B</th>
<th>1KB</th>
<th>10KB</th>
<th>100KB</th>
<th>1MB</th>
<th>10MB</th>
<th>100MB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modem</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ISDN</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T-1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T-3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E1 (2.048 Mbps)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T-3/12 (44 Mbps)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OC-3 (155 Mbps)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OC-12 (622 Mbps)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Home Page images on 20-80KB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Photos 200-600KB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Distance affects latency budget, which drives up BW cost of "interactive"
Unicast Experiments

Preliminary results

- **FTP (Infocom ’94)**
  - without proaction, per-item response averages 2.1 RTTs
  - with proaction 3x lower latency, 0.7 RTT avg. response, 7x higher BW

- **HTTP**
  - without proaction, 14% hit within 100 ms
  - with proaction, 83% hit within 100ms, 5-8x higher BW

Implications

- **Benefits**
  - Faster than speed-of-light response latency
  - Efficient multicast without requiring long server queues

- **Costs**
  - Resources - BW, CPU, storage
  - Complexity - contention avoidance for BW, CPU

Multicast Vision

Hot-spots are important

- **Significant traffic (conjecture)**
- **Important traffic, opportunity for interactive response**

HS’s generate communities of interest (COI)

- **Groups of users associated with a group of data**

COI are dynamic

- **Time scale of hours-days-weeks (conjecture and goal)**
- **E.g., tell a few friends, they’ll hit, etc.**

Content dictates COI

- **Predicted by URLs for now**
COI Components

Server
- *Creates COI page groups based on popularity*
- *Creates mcast channels for each COI group*
- *Advertises channels on index channel (per-server)*

Cache components
- *Partitioned, accepts remote loads*
- *COI channel per partition to receive mcast preloads*
- *Modified cache replacement*

Transport issues
- *“Lazy” reliable mcast transport*

Tuner Protocol

Publisher / subscriber relationship

Server
- *“TV-Guide” per server*
- *Multiple channels per server dynamically ‘topic of interest’*
- *Requests mcast if in a current ‘topic of interest’*

Cache
- *Partitioned per channel*
- *Tune to TV-Guide of popular servers*
- *Allocate partition per popular ‘topic of interest’*
- *(assumes within one server, or labelled)*
- *Automatically converges at network aggregation points*
Multicast architecture

1) GET x
2) x in cache?
3) GET x
4) Get file
5) mcast send x
6) redirect to GET x'
7) GET x'
8) x' in cache?

Issues

Cache partitioning

Item selection
- Ordering / prioritization
- At server, and replacement at client

Group selection
- At server and client

Transport issues
- Lazy multicast reliable transport
- Background unicast reliable transport
- Multicast parameter tuning (TTL)
Transport

Support mcast and unicast

Unicast selective NACKs
- *NACK triggered by cache hit, idle-ness, or API event*
- *NACK suppressed by new data*

Stateless servers
- *Retain partial transfers*

File and stream mode

Source or receiver controlled
- *Tag actions as silent/loud, optional/required, sync./async.*

Other Issues

Partial object caching
- *Variable-sized objects, variable cost complicates policy*
- *Need only enough to “prime the pipeline”*

Cache hierarchy overhead
- *Store-and-forward of tests increases MISS latency*
- *Use cut-through to root in parallel*

Network-adaptive caching
- *Use unicast preload, multicast, etc.*
- *Match cache mechanism to topology*
Environment Assumptions

Response latency is important

Idle bandwidth
- Opportunistic use of ephemeral resources
- Can be used without affecting foreground traffic

COIs aggregate
- Content-based subset of pages of a single server
- Hours-days of ‘hot-spot’

Architecture supports mcast
- Efficient mcast from server to caches
- Caches nearby to clients

Management Issues

Server decides what to send
- Creates COI groups based on “popularity”

Client decides what to receive
- Tunes partitions to COI channels based on “interest”

Partitioning avoids contention
- Background vs. foreground traffic
- Server processing queues
- Cache partitions
Protocol Issues

“Lazy” reliable multicast
- Currently using MFDP
- Prefer ‘lazy-NACK’ to avoid receiver overload

Supports hierarchy
- Mcast trees determine hierarchy automatically
- Avoiding transitivity also avoids store-and-forward costs

Server, network driven
- Server, proxies at network aggregation play
- Clients avoid extra individual load

LSAM Status

http://www.isi.edu/lsam

Prototype mcast system in test
- Uses MFDP
- Single, hard-wired group

Future work
- Server group selection
- Client group selection
- Cache replacement policy development
- Enforcing ‘backgrounder’ of mcast traffic